



HC32F460_F45x_A460 Series

32-bit ARM® Cortex®-M4 Microcontroller

Reference Manual

Rev1.5 September 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 List.....	30
Figure List.....	33
Overview.....	41
1 Memory Mapping	42
1.1 Memory Mapping.....	42
1.2 External Space Mapping.....	46
1.3 Bit Band Region.....	46
1.4 Address Remapping.....	46
1.5 Remap Register	48
1.5.1 Access Protection Register (MMF_REMPRT).....	48
1.5.2 Remap Control Register (MMF_REMCRx) (x=0, 1).....	49
2 Bus Architecture (BUS).....	50
2.1 Overview	50
2.2 Bus Architecture.....	51
2.3 Bus Function.....	52
3 Reset Control (RMU)	53
3.1 Introduction.....	53
3.2 Reset Mode and Reset Flag Bit	54
3.3 Reset Timing.....	56
3.3.1 Power-on Reset	56
3.3.2 Reset Pin Reset Pin Reset is a Reset caused by the Drive of the NRST Pin to a Low Level.	56
3.3.3 Brown-out Reset	57
3.3.4 Programmable Voltage Detection 1 Reset, Programmable Voltage Detection 2 Reset	58
3.3.5 Watchdog Reset, Dedicated Watchdog Reset	60
3.3.6 Power-down Wake-up Reset.....	60
3.3.7 Software Reset.....	61
3.3.8 MPU Error Reset	61
3.3.9 RAM Parity Reset.....	62
3.3.10 RAMECC Reset	62
3.3.11 Clock Frequency Exception Reset	62
3.3.12 External High Speed Oscillator Abnormal Shock Reset	63
3.3.13 Judgment of Reset Mode.....	63

3.3.14	Reset Conditions for Each Module.....	64
3.4	Register Description	65
3.4.1	Reset Flag Register0 (RMU _ RSTF0)	65
4	Clock Controller (CMU)	67
4.1	Introduction.....	67
4.2	System Block Diagram	68
4.2.1	System Block Diagram.....	68
4.2.2	Clock Frequency Measurement Block Diagram	69
4.3	Timer Specification.....	70
4.4	Working Clock Specification	71
4.5	Crystal Oscillator Circuit.....	72
4.5.1	External High-Speed Oscillator	72
4.5.2	External High-Speed Oscillator Fault Detection	74
4.5.3	External Low-speed Oscillator.....	76
4.6	Internal RC clock.....	77
4.6.1	HRC Clock	77
4.6.2	MRC Clock	77
4.6.3	LRC Clock	78
4.6.4	SWDTRC Clock	78
4.7	PLL Clock	78
4.8	Clock Switching Step	79
4.8.1	Timer Switchover	80
4.8.2	Clock Divider Switching	81
4.9	Clock Output Function	82
4.10	Clock Frequency Measurement.....	83
4.10.1	Clock Frequency Measurement.....	83
4.10.2	Digital Filtering Function	84
4.10.3	Interrupt/Reset Function	84
4.11	Register Description	85
4.11.1	CMU XTAL Configuration Register (CMU _ XTALCFGR).....	87
4.11.2	CMU XTAL Stability Configuration Register (CMU_XTALSTBCR)	87
4.11.3	CMU XTAL Control Register (CMU _ XTALCR).....	88
4.11.4	CMU XTAL Oscillatory Fault Control Register (CMU _ XTALSTDSCR)	88
4.11.5	CMU XTAL Oscillatory Fault State Register (CMU _ XTALSTDSSR).....	89
4.11.6	CMU XTAL32 Configuration Register (CMU_XTAL32CFGR).....	89
4.11.7	CMU XTAL32 Filter Register (CMU_XTAL32NFR).....	90
4.11.8	CMU XTAL32 Control Register (CMU_XTAL32CR).....	90
4.11.9	CMU HRC Calibration Register (CMU_HRCTRM).....	91

4.11.10 CMU HRC Control Register (CMU_HRCCR)	91
4.11.11 CMU MRC Calibration Register (CMU_MRCTRM)	92
4.11.12 CMU MRC Control Register (CMU_MRCCR)	92
4.11.13 CMU LRC Calibration Register (CMU_LRCTRM)	93
4.11.14 CMU LRC Control Register (CMU_LRCCR)	93
4.11.15 CMU MPLL Configuration Register (CMU_PLLCFG)	94
4.11.16 CMU MPLL Control Register (CMU_PLLCR)	95
4.11.17 CMU UPLL Configuration Register (CMU_UPLLCFG)	96
4.11.18 CMU UPLL Control Register (CMU_UPLLCR)	97
4.11.19 CMU Timer Stabilizer (CMU_OSCSTBSR)	98
4.11.20 Timer Switch Register of CMU System (CMU_CKSWR)	98
4.11.21 CMU Clock Divider Configuration Register (CMU_SCFGR)	99
4.11.22 CMU USBFS Clock Configuration Register (CMU_UFSCKCFG)	101
4.11.23 CMU AD/TRNG Clock Configuration Register (CMU_PERICKSEL)	102
4.11.24 CMU I2S Clock Configuration Register (CMU_I2SCKSEL)	103
4.11.25 CMU Debug Clock Configuration Register (CMU_TPIUCKCFG)	104
4.11.26 CMU MCO1 Configuration Register (CMU_MCO1CFG)	104
4.11.27 CMU MCO2 Configuration Register (CMU_MCO2CFG)	105
4.11.28 FCM Lower Limit Compare Value Register (FCM_LVR)	105
4.11.29 FCM Upper Limit Comparison Value Register (FCM_UVR)	106
4.11.30 FCM Counter Value Register (FCM_CNTR)	106
4.11.31 FCM Start Stop Register (FCM_STR)	106
4.11.32 FCM Measurement Object Control Register (FCM_MCCR)	107
4.11.33 FCM Measurement Reference Control Register (FCM_RCCR)	108
4.11.34 FCM Interrupt Reset Control Register (FCM_RIER)	109
4.11.35 FCM Flags Register (FCM_SR)	109
4.11.36 FCM Flag Clear Register (FCM_CLR)	110
5 Power Control (PWC).....	111
5.1 Introduction	111
5.2 Distribution of Power	111
5.3 Power Voltage Detection Unit (PVD) Description	113
5.3.1 Description of Action of Power-on/Power-off Reset.....	113
5.3.2 Brown-out Reset (BOR) Description	114
5.3.3 Programmable Voltage Detection 1 (PVD1), Programmable Voltage Detection 2 (PVD2)	115
5.3.4 PVD1, PVD2 Interrupt/Reset Block Diagram.....	116
5.3.5 Input/Output Pin.....	116
5.3.6 PVD1 Interrupt and Reset	117

5.3.7	PVD2 Interrupt and Reset	118
5.3.8	Internal Voltage Sampling and Detection Function	119
5.4	Action Mode and Low Power Mode	120
5.4.1	Operating Mode	123
5.4.2	Sleep Mode	125
5.4.3	Stop Mode.....	125
5.4.4	Power Down Mode	127
5.5	Methods of Reducing Power Consumption.....	130
5.5.1	Reduce System Clock Speed.....	131
5.5.2	Turn-off Unused Timer.....	131
5.5.3	Functional Clock Stop.....	131
5.5.4	Turn Off Unused RAM	131
5.6	Register Protection Function.....	133
5.7	Register Description	134
5.7.1	Power Mode Control Register 0 (PWC_PWRC0)	136
5.7.2	Power Mode Control Register 1 (PWC_PWRC1)	137
5.7.3	Power Mode Control Register 2 (PWC_PWRC2)	138
5.7.4	Power Mode Control Register 3 (PWC_PWRC3)	138
5.7.5	Power-down Wake-up Enable Register 0 (PWC_PDWKE0)	139
5.7.6	Power-down Wake-up Enable Register 1 (PWC_PDWKE1)	139
5.7.7	Power-down Wake-up Enable Register 2 (PWC_PDWKE2)	140
5.7.8	Power-down Wake-up Event Edge Selection Register (PWC_PDWKES).....	140
5.7.9	Power-down Wake-up Flag Register 0 (PWC_PDWKF0).....	141
5.7.10	Power-down Wake-up Flag Register 1 (PWC_PDWKF1).....	142
5.7.11	Power Monitor Control Register (PWC_PWCMR)	142
5.7.12	Mode Switch Control Register (PWC_MDSWCR)	143
5.7.13	Function Clock Control 0 (PWC_FCG0)	144
5.7.14	Function Clock Control 1 (PWC_FCG1)	146
5.7.15	Function Clock Control 2 (PWC_FCG2)	148
5.7.16	Function Clock Control 3 (PWC_FCG3)	150
5.7.17	PWC_FCG0 Protection Control (PWC_FCG0PC)	152
5.7.18	Function Protection Control Register (PWC_FPRC).....	152
5.7.19	STOP Mode Control Register (PWC_STPMCR)	153
5.7.20	RAM Power Control Register 0 (PWC_RAMPC0)	154
5.7.21	RAM Operating Condition Register (PWC_RAMOPM).....	155
5.7.22	XTAL32 is Controlled by a Current Source (PWC_XTAL32CS)	155
5.7.23	Wake-up Timer Control Register (PWC_WKTCR)	156
5.7.24	PVD Control Register 0 (PWC_PVDCR0)	156

5.7.25	PVD Control Register 1 (PWC_PVDCR1)	157
5.7.26	PVD Filter Control Register (PWC_PVDFCR).....	158
5.7.27	PVD Level Control Register (PWC_PVDLCR)	159
5.7.28	PVD Interrupt Control Register (PWC_PVDICR).....	160
5.7.29	PVD Detection Status Register (PWC_PVDDSR)	161
6	Initialization Configuration (ICG).....	162
6.1	Introduction	162
6.2	Register Description	163
6.2.1	Initialization Configuration Register0 (ICG0).....	163
6.2.2	Initialization Configuration register1 (ICG1).....	165
6.2.3	Initialization Configuration Registern (ICGn) n = 2 ~ 3.....	166
6.2.4	Initialize Configuration Register 4 (ICG4).....	166
6.2.5	Initialize Configuration Register 5 (ICG5).....	166
6.2.6	Initialization Configuration Registern (ICGn) n = 6 ~ 7.....	167
7	Embedded FLASH (EFM)	168
7.1	Introduction	168
7.2	Main Characteristics.....	168
7.3	Embedded FLASH	169
7.4	Read Interface	171
7.4.1	Relationship between CPU Clock and FLASH Read Time.....	171
7.4.2	FLASH Low Power Read.....	171
7.5	FLASH Read Acceleration Cache.....	173
7.6	FLASH Programming and Erasing Operations.....	174
7.6.1	Single Programming Read-free Mode.....	174
7.6.2	Single Programming Read-back Mode	174
7.6.3	Continuous Programming Operation.....	175
7.6.4	Erase Operation.....	175
7.6.5	Data Security Protection	176
7.6.6	Bus hold function.....	177
7.6.7	FLASH Wipe, Programming Window Protection.....	177
7.6.8	Interrupt.....	178
7.7	One Time Programmable Byte.....	179
7.8	Boot Swap	180
7.9	Register Description	181
7.9.1	Access Protection Register (EFM _ FAPRT).....	182
7.9.2	FLASH Stop Register (EFM_FSTP).....	182
7.9.3	Read Mode Register (EFM _ FRMC).....	183
7.9.4	Erase Mode Register (EFM _ FWMC).....	184

7.9.5	Status Register (EFM _ FSR).....	185
7.9.6	Status Purge Register (EFM _ FSCLR).....	186
7.9.7	Interrupt Enable Register (EFM _ FITE).....	187
7.9.8	Boot Swap Status Register (EFM_FSWP)	187
7.9.9	FLASH Window Protection Start Address register (EFM _ FPMTSW).....	188
7.9.10	FLASH Window Protection End Address register (EFM _ FPMTEW)	188
7.9.11	UNIQUE ID Register (EFM_UQID1)	188
7.9.12	UNIQUE ID Register (EFM_UQID2)	189
7.9.13	UNIQUE ID Register (EFM_UQID3)	189
7.10	Precautions.....	190
8	Built-in SRAM (SRAM).....	191
8.1	Introduction	191
8.2	Register Description	193
8.2.1	SRAM Wait Control Register (SRAM_WTCR)	193
8.2.2	SRAM Wait Protection Register (SRAM_WTPR).....	195
8.2.3	SRAM Check Control Register (SRAM_CKCR)	196
8.2.4	SRAM Check Protection Register (SRAM_CKPR).....	197
8.2.5	SRAM Check Status Register (SRAM_CKSR)	198
9	General IO (GPIO)	199
9.1	Introduction	199
9.2	Port Function Summary	200
9.3	Action Description	201
9.3.1	General Purpose Input/Output GPIO function.....	201
9.3.2	Peripheral Functions	202
9.3.3	Dual Peripheral Function.....	202
9.3.4	Event Port Input and Output Functions	202
9.3.5	External InterruptEIRQ Input Function	203
9.3.6	Analogfunction.....	203
9.3.7	General Control.....	204
9.4	Register Description	205
9.4.1	General Input Register (PIDRx)	206
9.4.2	General Output Data Register (PODRx)	206
9.4.3	General Output License Register (POERx)	206
9.4.4	General Output Set Register (POSRx)	207
9.4.5	General Output Reset Register (PORRx)	207
9.4.6	General Output Flip register (POTRx).....	207
9.4.7	Special Control Register (PSPCR)	208
9.4.8	Public Control R egister (PCCR).....	208

9.4.9	Input Control Register (PINAER)	209
9.4.10	Write Protection Register (PWPR).....	209
9.4.11	General Control Register (PCRxy)	210
9.4.12	Function Select Register (PFSRxy)	211
9.4.13	Event Port Direction Selection Register (PEVNTDIRM)	212
9.4.14	Event Port Input Data Register (PEVNTIDRM)	212
9.4.15	Event Port Output Data Register (PEVNTODRM).....	213
9.4.16	Event Port Output Data Reset Register (PEVNTORM)	213
9.4.17	Event Port Output Data Set Register (PEVNTOSRM)	214
9.4.18	Event Port Rising Edge Input Permission Register (PEVNTRISRM)	214
9.4.19	Event Port Falling Edge Input Enable Register (PEVNTFALRM)	215
9.4.20	Event Port Input Filter Control Register (PEVNTNFCR).....	216
9.4.21	32bit Access.....	217
9.5	Precautions.....	218
10	Interrupt Controller (INTC)	219
10.1	Introduction	219
10.2	INTC System Block Diagram	221
10.2.1	System Block Diagram.....	221
10.3	Directional Scale.....	222
10.3.1	Interrupt Digraph	222
10.3.2	Interrupt Event Request Sequence Number.....	227
10.4	Functional Description	245
10.4.1	Nomaskable Interrupt	245
10.4.2	External Pininterrupt Event Request	246
10.4.3	Interrupt Source Selection	246
10.4.4	Software Interrupt.....	246
10.4.5	Interrupt/Event Selection	247
10.4.6	WFE Wake-up Event Management	247
10.4.7	Digital Filter	249
10.4.8	Low Power Mode Return.....	250
10.4.9	Internal Trigger Event	250
10.5	Register Description	251
10.5.1	NMI Interrupt Control Register (INT_NMICR).....	256
10.5.2	NMI Interrupt Enable Register (INT_NMIENR)	257
10.5.3	NMI Flag Register (INT_NMIFR).....	258
10.5.4	NMI Clear Flag Register (INT_NMICFR)	259
10.5.5	EIRQ Control Register (INT_EIRQCRx) (x=0~15).....	260
10.5.6	EIRQ Flag Register (INT_EIFR)	261

10.5.7	EIRQ Flag Clear Register (INT_EICFR).....	261
10.5.8	Interrup Source Select Register (INT_SEL0~31).....	261
10.5.9	Interrup Source Select Register (INT_SEL32~127).....	262
10.5.10	Vector Sharing Interrup Source Select Register (INT_VSSEL128~143)	263
10.5.11	Soft-standby Wake Up Enable Register (INT_WUPEN)	264
10.5.12	Software Interrupt & Event Register (INT_SWIER).....	265
10.5.13	Event Enable Register (INT_EVTER)	265
10.5.14	Interrup Enable Register (INT_IER).....	265
10.6	Precautions for use.....	266
11	Automatic Operation System (AOS).....	267
11.1	Introduction.....	267
11.1.1	Function Overview	267
11.1.2	Module diagram.....	268
11.2	Functional Description	269
11.2.1	List of AOS Source Events.....	269
11.2.2	AOS Target List.....	269
11.3	Action Description	270
11.3.1	Dedicated Trigger Source.....	270
11.3.2	Public Trigger Source	270
11.4	Register Description	271
11.4.1	Peripheral Trigger Event Register (INTSFTTRG).....	272
11.4.2	DCU Trigger Source Selection Register (DCU_TRGSELx) (x=1~4).....	273
11.4.3	DMA1 Transfer Start Trigger Source Selection Register (DMA1_TRGSELx) (x=0~3)	274
11.4.4	DMA2 Transfer Start Trigger Source Selection Register (DMA2_TRGSELx) (x=0~3)	275
11.4.5	DMA Channel Reset Trigger Source Selection Register (DMA_TRGSELRC)	276
11.4.6	Timer6 Hardware Trigger Event Selection Register (TMR6_HTSSRx) (x=0~1)	277
11.4.7	Timer0 Hardware Trigger Event Selection Register (TMR0_HTSRR).....	278
11.4.8	Event Port Trigger Source Select Register (PEVNTRGSR12, PEVNTRGSR34)	279
11.4.9	TimerA Internal Trigger Event Selection Register 0 (TMRA_HTSR0)	280
11.4.10	TimerA Internal Trigger Event Selection Register 1 (TMRA_HTSR1)	281
11.4.11	OTS Trigger Source Selection Register (OTS_TRG).....	282
11.4.12	A/D1 Conversion Starts On-chip Trigger Source Selection Register ADC1_ITRGSELRx(x=0,1).....	283
11.4.13	A/D2 Conversion Starts On-chip Trigger Source Selection Register ADC2_ITRGSELRx(x=0,1).....	284
11.4.14	Common Trigger Source Selection Register 1 (AOS_COMTRG1).....	284

11.4.15 Common Trigger Source Selection Register 2 (AOS_COMTRG2).....	285
12 Keyboard Scan Control Module (KEYSCAN)	286
12.1 Introduction	286
12.2 KEYSAN System Block Diagram.....	286
12.3 Pin Description	287
12.4 Functional Description.....	287
12.4.1 Key Recognition Function.....	287
12.4.2 Keyboard Scan Function.....	287
12.4.3 Precautions for Use	288
12.5 Register Description	289
12.5.1 KEYSAN Scan Control Register (KEYSCAN_SCR).....	290
12.5.2 KEYSAN Scan Enable Register (KEYSCAN_SER)	291
12.5.3 KEYSAN Scan Status Register (KEYSCAN_SSR)	291
13 Memory Protection Unit (MPU)	292
13.1 Introduction	292
13.2 Functional Description.....	293
13.2.1 Locale Setting	293
13.2.2 Permission Settings.....	293
13.2.3 MPU Action Selection	293
13.2.4 Start MPU	293
13.3 Application Examples	294
13.3.1 Only Allow Partial Space Access.....	294
13.3.2 Only Some Space Access is Orohibited.....	294
13.4 Register Description	295
13.4.1 Area scope Description Register MPU_RGDn (n=0~15)	296
13.4.2 Regional Control Register MPU_RGCRn (n=0~15)	297
13.4.3 Control Register MPU_CR	298
13.4.4 Status Flag Register MPU_SR	299
13.4.5 Flag Clear Register MPU_ECLR	299
13.4.6 Write protection register MPU_WP	300
13.4.7 IP Access Protection Register MPU_IPPR.....	301
14 DMA Controller (DMA)	303
14.1 Introduction	303
14.2 Module Diagram	304
14.3 Functional Description	305
14.3.1 Enable DMA controller	305
14.3.2 Channel Selection and Channel Priority	305

14.3.3	Start DMA	305
14.3.4	Data Block	305
14.3.5	Transmission Address Control	305
14.3.6	Number of Transmissions.....	306
14.3.7	Interrupt and Event Signal Output.....	306
14.3.8	Chain Transmission	307
14.3.9	Discontinuous Address Transmission	309
14.3.10	Channel Reset.....	310
14.3.11	Premature Interrupt of Transmission.....	312
14.4	Application Examples	313
14.4.1	Memory-to-Memory Transmission	313
14.4.2	Memory-to-Peripheral Transmission	314
14.4.3	Memory-to-Memory Linkage	316
14.5	Register Description	318
14.5.1	DMA Enable Register (DMA _ EN).....	319
14.5.2	Interrupt State Register0 (DMA _ INTSTAT0)	319
14.5.3	Interrupt State Register0 (DMA _ INTSTAT1)	320
14.5.4	Interrupt Shield Register (DMA _ INTMASK0)	320
14.5.5	Interrupt Shield Register (DMA _ INTMASK1)	321
14.5.6	Interrupt Reset Register (DMA _ INTCLR0)	321
14.5.7	Interrupt Reset Register (DMA _ INTCLR1)	322
14.5.8	Channel Enable Register (DMA _ CHEN).....	322
14.5.9	Channel Reset Control Register (DMA_RCFGCTL).....	323
14.5.10	Transfer Request Status Register (DMA_REQSTAT).....	325
14.5.11	Channel State Observation Register (DMA _ CHSTAT).....	326
14.5.12	Transmission Source Address Register (DMA _ SARx) (x = 0 ~ 3).....	326
14.5.13	Transport Destination Address Register (DMA _ DARx) (x = 0 ~ 3).....	327
14.5.14	Data Control Register (DMA_DTCTLx) (x=0~3).....	327
14.5.15	Repeat Region Size Register (DMA_RPTx) (x=0~3)	328
14.5.16	Repeat area size register B (DMA_RPTBx) (x=0~3)	329
14.5.17	Source Device Discontinuous Address Transfer Control Register (DMA_SNSEQCTLx) (x=0~3) 330	
14.5.18	Source Device Discontinuous Address Transfer Control Register B (DMA_SNSEQCTLBx) (x=0~3)	331
14.5.19	Target Device Discontinuous Address Transfer Control Register (DMA_DNSEQCTLx) (x=0~3) 332	
14.5.20	Target Device Discontinuous Address Transfer Control Register B (DMA_DNSEQCTLBx) (x=0~3)	333

14.5.21	Chain Pointer Register (DMA_LL _P x) (x=0~3)	334
14.5.22	Channel Control Register (DMA_CH _x CTL) (x = 0 ~ 3)	335
14.5.23	Channel Monitor Registers (DMA_MONSAR _x , DMA_MONDAR _x , DMA_MONDTCTL _x , DMA_MONRPT _x , DMA_MONSEQCTL _x , DMA_MONDSEQCTL _x) (x=0~3)	336
14.6	Precautions for Use.....	337
15	Voltage Comparator (CMP)	338
15.1	Introduction.....	338
15.2	Functional Description.....	340
15.2.1	Voltage Comparison.....	340
15.2.2	Digital Filtering.....	341
15.2.3	Interrupt and Events.....	341
15.2.4	Scan Compare Mode.....	342
15.2.5	8bit-DAC Setting	343
15.3	Precautions.....	344
15.3.1	Module Stop Function	344
15.3.2	Act of Stopping a Module.....	344
15.3.3	Stop Action in Low Power Mode	344
15.3.4	Actions in Power-down Low-power Mode	344
15.4	Register Description	345
15.4.1	CMP Control Register (CMP_CTRL)	346
15.4.2	CMP Voltage Select Register (CMP_VLTSEL)	347
15.4.3	CMP Result Monitor Register (CMP_OUTMON).....	348
15.4.4	CMP Stabilization Time Register (CMP_CVSSTB)	348
15.4.5	CMP Compare Voltage Scan Period Register (CMP_CVSPRD)	349
15.4.6	8bit-DAC Control Register for CMP (CMP_DACR)	349
15.4.7	8bit-DAC Data Registers for CMP (CMP_DADR1, CMP_DADR2).....	349
15.4.8	CMP Internal Reference Voltage AD Conversion Register (CMP_RVADC)	350
16	Analog-to-Digital Conversion Module (ADC)	351
16.1	Introduction.....	351
16.2	ADC System Block Diagram	352
16.3	Functional Description.....	354
16.3.1	ADC Clock	354
16.3.2	Channel Selection	354
16.3.3	Trigger Source Selection	355
16.3.4	Sequence A Single Scan Mode.....	356
16.3.5	Sequence A Continuous Scanning Mode.....	357
16.3.6	Double Sequence Scanning Mode.....	358
16.3.7	Analogwatchdog Function.....	360

16.3.8	Sampling Time and Conversion Time of Analog Input.....	361
16.3.9	Automatic Clearing Function of A/D Data Register.....	362
16.3.10	Conversion Data Average Calculation Function	363
16.3.11	Programmable Gain Amplifier PGA	363
16.3.12	Multi-ADC Cooperative Working Mode	364
16.3.13	Interrupt and Event Signal Output.....	370
16.4	Register Description	372
16.4.1	A/D Launches Register (ADC _ STR).....	373
16.4.2	A/D Control Register 0 ADC_CR0.....	374
16.4.3	A/D Control Register1 (ADC _ CR1)	375
16.4.4	A/D Conversion Starts Triggering Register (ADC _ TRGSR).....	376
16.4.5	A/D Channel Select RegisterA (ADC _ CHSELRA0).....	377
16.4.6	A/D Channel Select Register A 1 ADC_CHSELRA1	377
16.4.7	A/D Channel Selection RegisterB (ADC _ CHSELRB0).....	378
16.4.8	A/D Channel Select Register B1 ADC_CHSELRB1.....	378
16.4.9	A/D Average Channel Selection Register ADC_AVCHSEL0.....	379
16.4.10	A/D Average Channel Selection Register 1 ADC_AVCHSEL1.....	379
16.4.11	A/D Sampling Status Register (ADC _ SSTR)	380
16.4.12	A/D Channel Mapping Control Register ADC_CHMUXR.....	381
16.4.13	A/D Interrupt State Register (ADC _ ISR).....	382
16.4.14	A/D Interrupt License Register (ADC _ ICR)	382
16.4.15	A/D Cooperative Mode Control Register ADC_SYNCCR	383
16.4.16	A/D Data Register (ADC _ DR).....	384
16.4.17	Analog Watchdog Control Register (ADC _ AWDCR).....	385
16.4.18	Analog Watchdog Threshold Register ADC_AWDDR0, ADC_AWDDR1	386
16.4.19	Analog Watchdog Compare Channel Select Register ADC_AWDCHSR0	387
16.4.20	Analog Watchdog Compare Channel Select Register 1 ADC_AWDCHSR1	387
16.4.21	Analog Watchdog Status Register ADC_AWDSR0.....	388
16.4.22	Analog Watchdog Status Register 1 ADC_AWDSR1.....	388
16.4.23	A/D Programmable Gain Amplifier Control Register ADC_PGACR.....	389
16.4.24	A/D Programmable Gain Multiplier Register ADC_PGAGSR	389
16.4.25	A/D Programmable Gain Amplifier Input Selection Register ADC_PGAINSRO	390
16.4.26	A/D Programmable Gain Amplifier Input Selection Register 1 ADC_PGAINSRI	390
16.5	Precautions for Use.....	391
16.5.1	Considerations when Reading Data Register	391
16.5.2	Scan Finish Interrupt Handling Considerations	391
16.5.3	Precautions for Module Stop and Low Power Consumption Setting	391
16.5.4	Pin Setting for A/D Converter Analog Channel Input.....	391

16.5.5	Noise Control	391
17	Temperature Sensor (OTS).....	392
17.1	Introduction	392
17.2	Usage Notes	393
17.3	Register Description	395
17.3.1	OTS Control Register (OTS_CTL)	395
17.3.2	OTS Data Register 1 (OTS_DR1)	396
17.3.3	OTS Data Register 2 (OTS_DR2)	396
17.3.4	OTS Error Compensation Register (OTS_ECR)	396
18	Advanced Control Timer (Timer6)	397
18.1	Introduction	397
18.2	Basic Block Diagram.....	397
18.3	Functional Description.....	399
18.3.1	Basic Action	399
18.3.2	Timer Selection.....	402
18.3.3	Counting Direction	402
18.3.4	Digital Filtering.....	403
18.3.5	Software Synchronization	404
18.3.6	Hardware Synchronization	405
18.3.7	Pulse Width Measurement	407
18.3.8	Periodic Measurement	407
18.3.9	Cache Function	407
18.3.10	General PWM Output	414
18.3.11	Orthogonal Coding Count	419
18.3.12	Periodic interval response.....	424
18.3.13	EMB Control	425
18.3.14	Typical Application	425
18.3.15	Function Summary Table.....	432
18.4	Interrupt and Event Description	434
18.4.1	Interrupt Output.....	434
18.4.2	Event Output	435
18.5	Register Description	436
18.5.1	General Counter Value Register (TMR6_CNTER)	438
18.5.2	Universal Period Reference Register (TMR6_PERA R-PERCR).....	438
18.5.3	General Compare Reference Registers (TMR6_GCMAR-GCMFR)	438
18.5.4	Dedicated Comparison Reference Registers (TMR6_SCMAR-SCMFR)	439
18.5.5	Dead Time Reference Register (TMR6_DTU<D>AR).....	439
18.5.6	General Control Register (TMR6_GCONR)	440

18.5.7	Interrupt Control Register (TMR6_ICONR)	441
18.5.8	Port Control Register (TMR6_PCONR).....	443
18.5.9	Cache Control Register (TMR6_BCONR)	445
18.5.10	Dead Band Control Register (TMR6_DCONR)	447
18.5.11	Filter Control Register (TMR6_FCONR)	448
18.5.12	Valid Period Register (TMR6_VPERR).....	449
18.5.13	Status Flag Register (TMR6_STFLR)	450
18.5.14	Hardware Start Event Select Register (TMR6_HSTAR).....	451
18.5.15	Hardware Stop Event Select Register (TMR6_HSTPR)	452
18.5.16	Hardware Clear Event Select Register (TMR6_HCLRR).....	453
18.5.17	Hardware Capture Event Select Register (TMR6_HCPAR).....	454
18.5.18	Hardware Capture Event Select Register (TMR6_HCPBR)	455
18.5.19	Hardware Increment Event Select Register (TMR6_HCUPR).....	456
18.5.20	Hardware Decrement Event Select Register (TMR6_HCDOR).....	457
18.5.21	Software Synchronous Start Control Register (TMR6_SSTAR)	459
18.5.22	Software Synchronous Stop Control Register (TMR6_SSPPR)	459
18.5.23	Software Synchronous Clear Control Register (TMR6_SCLRR)	460
19	General Control Timer (Timer4)	461
19.1	Introduction	461
19.2	Basic Block Diagram.....	461
19.3	Functional Description	463
19.3.1	Basic Action	463
19.3.2	Cache Function	466
19.3.3	General PWM output.....	472
19.3.4	Periodic Interval Response.....	478
19.3.5	EMB Control	480
19.4	Interrupt and Event Description	481
19.4.1	Count Comparison Match Interrupt.....	481
19.4.2	Count Cycle Match Interrupt	481
19.4.3	Heavy Load Count Matching Interrupt	481
19.4.4	Dedicated Comparison Matching Event.....	481
19.5	Register Description	483
19.5.1	Count Value Register (TMR4 _ CNTR)	485
19.5.2	Periodic Reference Register (TMR4 _ CPSR)	485
19.5.3	Control State Register (TMR4 _ CCSR)	486
19.5.4	Valid Period Register (TMR4 _ CVPR)	487
19.5.5	Generic Baseline Register (TMR4 _ OCCRm)	487
19.5.6	General Control State Register (TMR4 _ OCSRn).....	488

19.5.7	Generic Extended Control Register (TMR4 _ OCERN)	489
19.5.8	General Mode Control Register (TMR4 _ OCMRm).....	491
19.5.9	Special Reference Register (TMR4 _ SCCRm).....	496
19.5.10	Special Control State Register (TMR4 _ SCSRm)	497
19.5.11	Special Mode Control Register (TMR4 _ SCMRm)	498
19.5.12	PWM Basic Control Register (TMR4 _ POCRn)	499
19.5.13	PWM Filtering Control Register (TMR4 _ PFSRn)	500
19.5.14	PWM Dead Zone Control Register (TMR4 _ PDA < B > Rn)	500
19.5.15	Overload Control Status Register (TMR4 _ RCSR)	501
19.5.16	EMB Control State Register (TMR4 _ ECSR).....	502
19.5.17	EMB Extended Control Register (TMR4_ECR)	502
20	Emergency Brake Module (EMB)	503
20.1	Introduction.....	503
20.2	Functional Description	504
20.2.1	Overview.....	504
20.2.2	Stop PWM Signal Output When Input Level Of External Port Changes.....	504
20.2.3	Stop PWM Signal Output when the Level of PWM Output Port is in the Same Phase (Same High or Same Low).....	505
20.2.4	Stop PWM Signal Output According To Voltage Comparator Comparison Result....	506
20.2.5	Stop PWM Signal Output When External Oscillator Stops Oscillating.....	506
20.2.6	Write Register Software Control PWM Signal Output.....	506
20.3	Register Description	507
20.3.1	EMB Control Register 0 (EMB_CTL0)	508
20.3.2	EMB Control Register 1~3 (EMB_CTL1~3)	509
20.3.3	EMB Feedback Level Selection Register 0 (EMB_PWMLV0).....	510
20.3.4	EMB Feedback Level Selection Register 1~3 (EMB_PWMLV1~3)	511
20.3.5	EMB Software Output Enable Control Register (EMB_SOEx) (x=0~3).....	511
20.3.6	EMB Status Register (EMB_STATx) (x=0~3).....	512
20.3.7	EMB Status Reset Register (EMB_STATCLRx) (x=0~3)	513
20.3.8	EMB Interrupt Permission Register (EMB_INTENx) (x=0~3)	514
21	General Timer (TimerA)	515
21.1	Introduction.....	515
21.2	Basic Block Diagram.....	515
21.3	Functional Description	517
21.3.1	Basic Action	517
21.3.2	Timer Selection.....	520
21.3.3	Synchronous Startup	521
21.3.4	Digital Filtering.....	522

21.3.5	Cache Function	523
21.3.6	Cascade Count.....	524
21.3.7	PWM Output.....	525
21.3.8	Orthogonal Coding Count	526
21.4	Interrupt and Event Description	531
21.4.1	Compare Matched Interrupt and Events	531
21.4.2	Periodic Matching Interrupt and Events	531
21.5	Register description.....	532
21.5.1	Generic Count Value Register (TMRA _ CNTER).....	534
21.5.2	Periodic Reference Value Register (TMRA _ PERAR)	534
21.5.3	Comparison Base Value Register (TMRA_CMPAR1~8).....	534
21.5.4	Control State Register (TMRA _ BCSTR)	535
21.5.5	Interrupt Control Register (TMRA _ ICONR)	536
21.5.6	Event Control Register (TMRA _ ECONR)	537
21.5.7	Filter Control Register (TMRA _ FCONR)	538
21.5.8	Status Flag Register (TMRA _ STFLR)	539
21.5.9	Cache Control Register (TMRA_BCONR1~4)	540
21.5.10	Capture Control Register (TMRA_CCONR1~8).....	541
21.5.11	Port Control Register (TMRA_PCONR1~8)	542
21.5.12	Hardware Trigger Event Selection Register (TMRA _ HCONR)	543
21.5.13	Hardware Added Event Selection Register (TMRA _ HCUPR).....	544
21.5.14	Hardware Decrease Event Select Register (TMRA _ HCDOR)	545
22	General Timer (Timer0)	546
22.1	Introduction	546
22.2	Basic Block Diagram.....	546
22.3	Functional Description	547
22.3.1	Timer Selection.....	547
22.3.2	Basic Counting Action	548
22.3.3	Hardware Trigger Action.....	548
22.4	Interrupt and Event Description	549
22.4.1	Interrupt Output.....	549
22.4.2	Event Output	549
22.5	Register Description	550
22.5.1	Count Value Register (TMR0_CNTAR).....	550
22.5.2	Reference Value Register (TMR0_CMPAR).....	550
22.5.3	Basic Control Register (TMR0 _ BCONR).....	551
22.5.4	Status flag register (TMR0 _ STFLR).....	553
22.6	Precautions for Use.....	554

23 Real Time Clock (RTC)	555
23.1 Introduction	555
23.2 Basic block diagram	556
23.3 Functional Description	557
23.3.1 Power-on Setting	557
23.3.2 RTC Count Start Setting	557
23.3.3 System Low Power Mode Switching	557
23.3.4 Read Count Register	557
23.3.5 Write Count Register	558
23.3.6 Alarm Clock Setting	558
23.3.7 Clock Error Compensation	558
23.3.8 1Hz Output	558
23.4 Interrupt Description	560
23.4.1 Alarm Interrupt	560
23.4.2 Periodic Interrupt	560
23.5 Register Description	561
23.5.1 Control Register 0 (RTC_CR0)	562
23.5.2 Control Register 1 (RTC_CR1)	563
23.5.3 Control Register 2 (RTC_CR2)	564
23.5.4 Control Register 3 (RTC_CR3)	565
23.5.5 Second Count Register (RTC_SEC)	565
23.5.6 Minute Count Register (RTC_MIN)	566
23.5.7 Hour Count Register (RTC_HOUR)	567
23.5.8 Day Count Register (RTC_DAY)	569
23.5.9 Week Count Register (RTC_WEEK)	570
23.5.10 Month Count Register (RTC_MON)	571
23.5.11 Year Count Register (RTC_YEAR)	571
23.5.12 Minute Alarm Register (RTC_ALMMIN)	572
23.5.13 Hour Alarm register (RTC_ALMHOUR)	572
23.5.14 Week Alarm Register (RTC_ALM WEEK)	573
23.5.15 When Clock Error Compensation, Set The Counting Clock Error Compensation Registers RTC_ERRCRL, RTC_ERRCRH;	574
24 Watchdog Counter (WDT/ SWDT)	577
24.1 Introduction	577
24.2 Functional description	578
24.2.1 Start the watchdog	578
24.2.2 Hardware startup mode	578
24.2.3 Software Startup Mode	580

24.2.4 Refresh Action.....	581
24.2.5 Flag Bit.....	581
24.2.6 Interrupt Reduction.....	582
24.2.7 Count Underflow	582
24.2.8 Refresh Error.....	584
24.3 Register Description	585
24.3.1 Control Register (WDT_CR)	586
24.3.2 Status Register (SWDT_SR,WDT_SR).....	587
24.3.3 Refresh Register (SWDT_RR,WDT_RR)	587
24.4 Precautions for Use.....	588
25 General Synchronous Asynchronous Transceiver (USART).....	589
25.1 Introduction	589
25.2 USART System Block Diagram.....	590
25.3 Pin Description	590
25.4 Functional Description	591
25.4.1 UART	591
25.4.2 Multiprocessor Communication.....	600
25.4.3 Smart Card.....	605
25.4.4 Clock synchronization mode	609
25.4.5 Digital Filtering Function	615
25.5 Register Description	616
25.5.1 USART Status Register (USART_SR)	617
25.5.2 USART Data Register (USART_DR).....	620
25.5.3 USART Bit Rate Register (USART_BRR).....	621
25.5.4 USART Control Register 1 (USART_CR1).....	622
25.5.5 USART Control Register 2 (USART_CR2).....	625
25.5.6 USART Control Register 3 (USART_CR3).....	626
25.5.7 USART Prescaler Register (USART_PR)	627
25.6 Precautions for Use.....	628
25.6.1 UART Considerations.....	628
25.6.2 Clock Synchronization Mode Considerations.....	628
25.6.3 Other Considerations	628
26 Integrated Circuit Bus (I2C)	629
26.1 Introduction	629
26.2 I2C System Block Diagram	630
26.2.1 System Block Diagram.....	630
26.2.2 Structural Diagram	631
26.3 Action Description	632

26.3.1	I2C Protocol.....	632
26.3.2	Address Matching	641
26.3.3	SMBus Action	647
26.3.4	Reduction.....	649
26.3.5	Interrupt and Event Signal Output.....	649
26.3.6	Programmable Digital Filtering	650
26.4	Application Software Setting I2C Initialization Process	651
26.5	Register Description	652
26.5.1	I2C Control Register 1 (I2C_CR1)	653
26.5.2	I2C Control Register 1 (I2C_CR2)	654
26.5.3	I2C Controlled Register1(I2C_CR3).....	655
26.5.4	I2C Control Register 1 (I2C_CR4)	656
26.5.5	I2C Slave Address Register0 (I2C_SLR0)	657
26.5.6	I2C Slave Address Register 1 (I2C_SLR 1)	658
26.5.7	I2C SCL Level When Timeout Control Register (I2C_SLTR)	659
26.5.8	I2C State Register (I2C_SR).....	660
26.5.9	I2C State Reset Register (I2C_CLR)	663
26.5.10	I2C Data Sending Register (I2C_DTR)	664
26.5.11	I2C Data Receiving Register (I2C_DRR).....	664
26.5.12	I2C Data Shift Register (I2C_DSR).....	665
26.5.13	I2C Clock Control Register (I2C_CCR).....	666
26.5.14	I2C Filtering Control Register (I2C_FLTR)	668
27	Serial Peripheral Interface (SPI)	669
27.1	Introduction	669
27.2	SPI System Block Diagram	670
27.3	Pin Description	670
27.4	SPI Action System Description.....	671
27.4.1	Pin Status in Host Mode	671
27.4.2	Pin State in Slave Mode	671
27.4.3	SPI System Connection Examples.....	672
27.5	Data Communication Description.....	673
27.5.1	Baud Rate	673
27.5.2	Data Format.....	674
27.5.3	Transfer Format.....	675
27.5.4	Communication Mode	677
27.6	Operating Instructions.....	679
27.6.1	Operational Mode Summary	679
27.6.2	Host Action in SPI Operation Mode	680

27.6.3	The Slave Action in SPI Operation Mode	681
27.6.4	Host Action in Clock Synchronous Operation Mode	682
27.6.5	Slave Maneuver in Clock Synchronous Operation Mode	683
27.6.6	Processing Flow of Several SPI Actions	685
27.7	Self-diagnosis of Parity	686
27.8	Error Detection	687
27.8.1	Underload Error	687
27.8.2	Pattern Error	688
27.8.3	Overload Error	688
27.8.4	Parity Error	690
27.9	SPI Initialization	692
27.9.1	Clear SPE Bits for Initialization	692
27.9.2	System Reset Initialization	692
27.10	Interrupt Source	693
27.11	Available Event Trigger Sources	693
27.12	Register Description	694
27.12.1	SPI Data Register (SPI_DR)	694
27.12.2	SPI Control Register (SPI_CR1)	695
27.12.3	SPI Communication Configuration Register1 (SPI_CFG1)	696
27.12.4	SPI Status Register (SPI_SR)	697
27.12.5	SPI Communication Configuration Register2 (SPI_CFG2)	698
28	Quad-wire Serial Peripheral Interface (QSPI)	699
28.1	Introduction	699
28.2	Memory Map	701
28.2.1	Internal Bus Space	701
28.2.2	ROM Space and Bus Address Width	702
28.3	QSPI Bus	703
28.3.1	SPI Protocol	703
28.3.2	SPI mode	705
28.4	Timing Adjustment of QSPI Bus	706
28.4.1	QSPI Bus Reference Clock	706
28.4.2	SPI Bus Reference Clock	707
28.4.3	QSSN Signal The Minimum High Level Width	707
28.4.4	QSSN of Establishment Time	708
28.4.5	QSSN of Hold Time	708
28.4.6	Serial Data Receive Delay	709
28.5	Introduction to SPI Instructions for ROM Access	710
28.5.1	Existing QSPI-ROM Instruction Reference	710

28.5.2	Standard Read	711
28.5.3	Fast Read Command	711
28.5.4	Two-Wire Output Fast Read Instruction	713
28.5.5	Two-Wire Input Output Fast Read Instruction	714
28.5.6	Wire-Wire Output Fast Read Instruction	715
28.5.7	Wire-Wire Input Output Fast Read Instruction	716
28.5.8	Enter 4-Byte mode command	718
28.5.9	Exit 4-Byte Mode Command	718
28.5.10	Write Permission	719
28.6	Scheduling of QSPI Bus Cycle.....	719
28.6.1	Single Flash Read with Independent Conversion	719
28.6.2	Flash Read Using Read-Ahead	720
28.6.3	Termination of Prefetching	721
28.6.4	Prefetch Status Monitoring.....	721
28.6.5	Flash Read using QSPI Bus Cycle Stretching.....	721
28.7	XIP Control.....	722
28.7.1	Setting of XIP Mode.....	723
28.7.2	Exit from XIP Mode.....	723
28.8	Pin Status of QSIO2 and QSIO3.....	724
28.9	Direct Communication Mode	725
28.9.1	About Communication Mode.....	725
28.9.2	Setting of Direct Communication Mode	725
28.9.3	QSPI Bus Cycle Generation In Direct Communication Mode	725
28.10	Interrupt	726
28.11	Precautions for Use.....	726
28.11.1	Setting Sequence of QSPI Registers.....	726
28.11.2	Module Stop Signal Setting	726
28.12	Register Description	727
28.12.1	QSPI Control Register (QSCR)	728
28.12.2	QSPI Chip selection Control Register (QSCSCR)	730
28.12.3	QSPI Format Control Register (QSFCR).....	731
28.12.4	QSPI Status Register (QSSR)	732
28.12.5	QSPI Command Code Register (QSCCMD)	733
28.12.6	QSPI Direct Communication Command Register (QSDCOM)	733
28.12.7	QSPI XIP Mode Code Register (QSXCMD)	733
28.12.8	QSPI System Configuration Register (QSSR2)	734
28.12.9	QSPI External Extended Address Register (QSEXAR)	734
29	Integrated Circuit Built-in Audio Bus Module (I2S)	735

29.1	Introduction	735
29.2	I2S System Block Diagram	736
29.3	Pin Description	736
29.4	Functional Description	737
29.4.1	I2S General Description	737
29.4.2	Communication Mode	737
29.4.3	Supported Audio Protocols	737
29.4.4	Clock Generator	743
29.4.5	I2S Host Mode	745
29.4.6	I2S Slave Mode	746
29.4.7	I2S Interrupt	747
29.4.8	Precautions for Use	748
29.5	Register Description	751
29.5.1	I2S Control Register (I2S_CTRL)	752
29.5.2	I2S Status Register (I2S_SR)	753
29.5.3	I2S Error Status Register (I2S_ER)	753
29.5.4	I2S Configuration Register (I2S_CFGR)	754
29.5.5	I2S Transmit Buffer FIFO Data Register (I2S_TXBUF)	754
29.5.6	I2S Receive Buffer FIFO Data Register (I2S_RXBUF)	755
29.5.7	I2S Prescaler Register (I2S_PR)	755
30	Controller Area Network (CAN)	756
30.1	Introduction	756
30.2	CAN System Block Diagram	757
30.3	Pin Description	757
30.4	Functional Description	758
30.4.1	Baud Rate Setting	758
30.4.2	Send Buffer	759
30.4.3	receive buffer	760
30.4.4	Receive Filter Register Set	760
30.4.5	Data Transmission	761
30.4.6	Single Data Transmission	761
30.4.7	Cancel Data Sending	762
30.4.8	Data Reception	762
30.4.9	Error Handling	762
30.4.10	Node Down	763
30.4.11	Lost Arbitration Position Capture	763
30.4.12	Loop Mode	764
30.4.13	Silent Mode	765

30.4.14 Software Reset Function	766
30.4.15 Upward Compatible with CAN-FD Function	768
30.4.16 Time-Triggered TTCAN.....	768
30.4.17 Interrupt.....	771
30.5 Register Description	772
30.5.1 CAN Receive Buffer Registers (CAN_RBUF).....	774
30.5.2 CAN Transmit Buffer Registers (CAN_TBUF)	776
30.5.3 CAN Configuration and Status Register (CAN_CFG_STAT)	778
30.5.4 CAN Command Register (CAN_TCMD).....	779
30.5.5 CAN Transmit Control Register (CAN_TCTRL)	781
30.5.6 CAN Receive Control Register (CAN_RCTRL)	783
30.5.7 CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE)	784
30.5.8 CAN Receive and Transmit Interrupt Status Register (CAN_RTIF).....	785
30.5.9 CAN ERRor INTerrupt Enable and Flag Register (CAN_ERRINT)	787
30.5.10 CAN Bit Timing Register (CAN_BT)	788
30.5.11 CAN Error and Arbitration Lost Capture Register (CAN_EALCAP)	789
30.5.12 CAN Warning Limits Register (CAN_LIMIT)	789
30.5.13 CAN Receive Error CouNT Register (CAN_RECNT).....	790
30.5.14 CAN Transmit Error CouNT Register (CAN_TECNT)	790
30.5.15 CAN Acceptance Filter Control Register (CAN_ACFCTRL)	791
30.5.16 CAN Acceptance Filter Enable Register (CAN_ACFEN).....	792
30.5.17 CAN Acceptance Filter Code And Mask Register (CAN_ACF)	793
30.5.18 TTCAN TB Slot Pointer Register (CAN_TBSLOT)	794
30.5.19 TTCAN TB Slot Pointer Register (CAN_TTCFG)	795
30.5.20 TTCAN Reference Message Register (CAN_REF_MSG)	796
30.5.21 TTCAN Reference Message Register (CAN_TRG_CFG)	797
30.5.22 TTCAN Reference Message Register (CAN_TT_TRIG)	798
30.5.23 TTCAN Watch Trigger Time Register (CAN_TT_WTRIG)	798
30.6 Precautions for Use.....	799
30.6.1 CAN Bus Anti-Jamming Measures	799
30.6.2 CAN Controller Noise Constraints.....	799
31 USB2.0 Full Speed Module (USBFS)	800
31.1 Introduction to USBFS.....	800
31.2 Main Features of USBFS.....	800
31.2.1 General Features	800
31.2.2 Host Mode Features	801
31.2.3 Device Mode Properties	801
31.3 USBFS System Block Diagram	802

31.4	USBFS Pin Description	802
31.5	USBFS Function Description.....	803
31.5.1	USBFS Clock And Working Mode	803
31.5.2	USBFS Mode Decision	803
31.5.3	USBFS Host Function.....	804
31.5.4	USBFS Device Capabilities	809
31.5.5	USBFS SOF Pulse Pin Output Function.....	813
31.5.6	USBFS Power Control	813
31.5.7	USBFS Dynamically Updates the USBFS_HFIR Register	814
31.5.8	USBFS Data FIFO.....	814
31.5.9	USBFS Host FIFO Architecture.....	815
31.5.10	USBFS Device FIFO Architecture	817
31.5.11	USBFS FIFO RAM Allocation.....	818
31.5.12	USBFS System Performance	819
31.5.13	USBFS Interrupt and Events.....	820
31.6	USBFS Programming Model	821
31.6.1	USBFS Module Initialization.....	821
31.6.2	USBFS Host Initialization.....	821
31.6.3	USBFS Device Initialization	822
31.6.4	USBFS DMA Mode	822
31.6.5	USBFS Host Programming Model	823
31.6.6	USBFS Device Programming Model.....	824
31.6.7	USBFS Operational Model	826
31.7	Register Description	845
31.7.1	USBFS System Control Register	848
31.7.2	USBFS Global Registers.....	849
31.7.3	USBFS Host Mode Register	868
31.7.4	USBFS Device Mode Register.....	881
31.7.5	USBFS Clock Gating Control Register.....	905
32	Cryptographic Coprocessing Module (CPM)	906
32.1	Introduction	906
32.2	Encryption and Decryption Algorithm Processor (AES).....	906
32.2.1	Algorithm Introduction	906
32.2.2	Encryption Operation Process.....	907
32.2.3	Decryption Operation Process.....	907
32.2.4	Encryption and Decryption Time.....	908
32.2.5	Operation Precautions	908
32.2.6	Register Description.....	909

32.3	Secure Hash Algorithm (HASH).....	912
32.3.1	Algorithm Introduction.....	912
32.3.2	Operating Procedures	913
32.3.3	Message Padding	913
32.3.4	Register Description.....	915
32.4	True Random Number Generator (TRNG)	919
32.4.1	Block Diagram	919
32.4.2	Operating Procedures	919
32.4.3	Interrupt and Event Output.....	919
32.4.4	Operation Precautions	919
32.4.5	Register Description.....	920
33	Data Computing Unit (CRC)	922
33.1	Introduction.....	922
33.2	Functional Description	923
33.2.1	Additive Mode.....	923
33.2.2	Subtraction Mode.....	923
33.2.3	Hardware Triggered Boot Mode.....	923
33.2.4	Compare Mode.....	923
33.2.5	Interrupt and Event Signal Output.....	924
33.3	Application Examples	925
33.3.1	Additive Mode.....	925
33.3.2	Trigger Plus Mode.....	925
33.3.3	Compare Mode.....	925
33.4	Register Description	926
33.4.1	DCU Control Register (DCU_CTL)	928
33.4.2	DCU Flag Register (DCU_FLAG).....	929
33.4.3	DCU Data Register (DCU_DATAx) (x=0,1,2)	930
33.4.4	DCU Flag Reset Register (DCU_FLAGCLR).....	931
33.4.5	DCU Interrupt Condition Selection Register (DCU_INTEVTSEL)	932
34	CRC Operation (CRC)	933
34.1	Introduction	933
34.2	Functional Block Diagram.....	933
34.3	Functional Description	934
34.3.1	CRC Code Generation	934
34.3.2	CRC Check	934
34.3.3	CRC check for XOROUT, REFOUT, REFIN not all 1	935
34.4	Register Description	936
34.4.1	Control Register (CRC _ CR)	936

34.4.2	Results Register (CRC _ RESLT)	937
34.4.3	Flag Register (CRC_FLG)	938
34.4.4	Data Register (CRC _ DAT)	938
35	SDIO Controller (SDIOC)	939
35.1	Introduction	939
35.2	Functional Description	939
35.2.1	Port Assignment.....	939
35.2.2	Basic Access Method	940
35.2.3	Data Transmission.....	940
35.2.4	SD Clock	940
35.2.5	Interrupts and DMA Start Requests.....	941
35.2.6	Host and Device Initialization	942
35.2.7	SD/MMC Single Block (Single Block) Read and Write	943
35.2.8	SD/MMC Multi-block (Multi-block) Read and Write	945
35.2.9	Transfer Abort, Suspend and Resume	946
35.2.10	Read Wait.....	947
35.2.11	Wake Up.....	948
35.3	Register Description	950
35.3.1	Data Block Size Register (BLKSIZE)	951
35.3.2	Data Block Count Register (BLKCNT)	951
35.3.3	Parameter Register 0 (ARG0).....	952
35.3.4	Parameter Register 1 (ARG1).....	952
35.3.5	Transfer Mode Register (TRANSMODE).....	952
35.3.6	Command Register (CMD)	953
35.3.7	Response Register 0 (RESP0).....	953
35.3.8	Response Register 1 (RESP1).....	954
35.3.9	Response Register 2 (RESP2).....	954
35.3.10	Response Register 3 (RESP3).....	954
35.3.11	Response Register 4 (RESP4).....	955
35.3.12	Response Register 5 (RESP5).....	955
35.3.13	Response Register 6 (RESP6).....	955
35.3.14	Response Register 7 (RESP7).....	956
35.3.15	Data Buffer Register 0 (BUF0).....	956
35.3.16	Data Buffer Register 1 (BUF1).....	956
35.3.17	Host Status Register (PSTAT)	957
35.3.18	Host Control Register (HOSTCON)	958
35.3.19	Power Control Register (PWRCON)	958
35.3.20	Data Block Gap Control Register (BLKGPCON)	959

35.3.21	Clock Control Register (CLKCON)	959
35.3.22	Timeout Control Register (TOUTCON)	960
35.3.23	Software Reset Register (SFTRST).....	960
35.3.24	Normal Interrupt Status Register (NORINTST).....	961
35.3.25	Error Interrupt Status Register (ERRINTST).....	961
35.3.26	Normal Interrupt Status Enable Register (NORINTSEN)	962
35.3.27	Error Interrupt Status Enable Register (ERRINTSEN)	963
35.3.28	Normal Interrupt Signal Enable Register (NORINTSGEN)	964
35.3.29	Error Interrupt Signal Enable Register (ERRINTSGEN).....	965
35.3.30	Auto Command Error Status Register (ATCERRST).....	966
35.3.31	Force Autocommand Error Status Control Register (FEA)	966
35.3.32	Force Error Status Control Register (FEE).....	967
35.3.33	MMC Mode Enable Register (MMCER)	967
36	Debug Controller (DBG)	968
36.1	Introduction.....	968
36.2	DBG System Block Diagram	969
36.3	SWJ-DP Debug Port (SWD and JTAG)	970
36.3.1	Switching Mechanism of JTAG-DP or SW-DP	971
36.4	Pinout and Debug Port Pinout.....	972
36.4.1	SWJ Debug Port Pin	972
36.4.2	Flexible SWJ-DP pin assignment.....	972
36.4.3	Internal Pull-ups on JTAG Pins.....	973
36.4.4	Use the Serial Interface and Release Unused Debug Pins for GPIO	973
36.5	Register	974
36.5.1	DBG State Register (MCUDBGSTAT)	974
36.5.2	Peripheral Debugging Pause Register (MCUSTPCTL).....	975
36.5.3	Debug Component Configuration Register (MCUTRACECTL).....	977
36.6	SW Debug Port	978
36.6.1	Introduction to SW Agreement.....	978
36.7	TPIU (Trace Port Interface Unit)	978
36.7.1	Introduction	978
36.7.2	TRACE Pin Allocation.....	979
36.7.3	MCU Internal TRACECLKIN Connection.....	980
36.7.4	TPIU Register	980
36.7.5	TPIU Configuration Example	980
Version Revision History	981

Table List

Table 1-1	Memory-Map	42
Table 1-2	QSPI Address Space Allocation.....	46
Table 1-3	Sample Destination Address Configuration	47
Table 1-4	Register List	48
Table 3-1	Reset Methods and Generation Conditions	54
Table 3-2	Reset Methods and Reset Flags.....	55
Table 3-3	List of RMU Registers	65
Table 4-1	Specifications of Each Internal Clock.....	71
Table 5-1	BOR configuration	114
Table 5-2	PVD1/PVD2 Characteristics	115
Table 5-3	Operation Mode.....	120
Table 5-4	Low Power Modes	120
Table 5-5	Operating conditions of low power consumption mode and status of each module in low power consumption mode.....	121
Table 5-6	Operating Mode Description.....	123
Table 5-7	Power-down Mode Submodes.....	128
Table 5-8	RAM Module and RAM Power-down Control Bits	132
Table 5-9	Register Protection List	133
Table 5-10	Register Summary.....	134
Table 6-1	Register Summary.....	162
Table 7-1	Comparison Table of CPU Clock Frequency and FLASH Read Wait Period	172
Table 7-2	FLASH Actual Read Cycles.....	173
Table 7-3	OTP address composition	179
Table 7-4	Register Summary.....	181
Table 8-1	The relationship between the wait cycle setting of SRAM read and write access and the CPU clock frequency.....	191
Table 8-2	SRAM Space Allocation.....	192
Table 9-1	PORT Register Overview 1.....	205
Table 9-2	PORT Register Overview 2.....	205
Table 9-3	List of PORT registers for 32bit access	217
Table 10-1	Interrupt Vector Table	222
Table 10-2	Interrupt Event Request Sequence Number and Selection.....	227
Table 12-1	KEYSCAN pin description.....	287
Table 12-2	KEYSCAN register list	289
Table 14-1	Channel Reset Description	310
Table 15-1	CMP Detailed Specifications	338

Table 15-2	List of CMP Registers	345
Table 16-1	Specifications of each ADC unit	353
Table 16-2	Register Setting Method when Switching Internal Channels	355
Table 16-3	Sequences A and B of Various Competitions	358
Table 16-4	AD Conversion Time	362
Table 16-5	List of ADC Registers 1/2	372
Table 17-1	List of OTS Registers	395
Table 18-1	Basic functions and features of Timer6	397
Table 18-2	Timer6 port list.....	398
Table 18-3	Comparison of functions in different modes.....	432
Table 18-4	Register List	436
Table 19-1	Basic Functions And Features Of Timer4	461
Table 19-2	Timer4 Port List	462
Table 19-3	Register List	483
Table 21-1	Basic Functions and Features of TimerA.....	515
Table 21-2	TimerA port list.....	516
Table 21-3	Register List	532
Table 22-1	Register List	550
Table 23-1	Basic Specifications of RTC.....	555
Table 23-2	Register List	561
Table 24-1	Basic characteristics of watchdog counter.....	577
Table 24-2	Register List	585
Table 25-1	USART Pin Descriptions	590
Table 25-2	Tolerance of UART receiver when DIV_Fraction is 0	596
Table 25-3	Tolerance of UART receiver when DIV_Fraction is not 0	596
Table 25-4	UART interrupt/Event Table.....	600
Table 25-5	Multi-Processor Mode Interrupt/ Event	604
Table 25-6	Smart Card Mode Interrupt/Event	609
Table 25-7	Clock Synchronization Mode Interrupt/Event Table	615
Table 25-8	List of USART Registers	616
Table 25-9	Baud rate calculation formula (invalid fractional baud rate FBME=0).....	621
Table 25-10	Baud rate calculation formula valid fractional baud rate FBME=1).....	621
Table 26-1	Input/output Pin.....	631
Table 26-2	Register Summary.....	652
Table 27-1	Characteristics of SPI	669
Table 27-2	Pin Description	670
Table 27-3	SPI Pin Status Description in Host Mode.....	671
Table 27-4	SPI Pin Status Description in Slave machine Mode.....	671

Table 27-5 Lower speed of partial setting	673
Table 27-6 SPI Pattern and Register Setting Relationship	679
Table 27-7 Correspondence Table for Error Detection.....	687
Table 27-8 SPI interrupt Source Description	693
Table 28-1 QSPI main Specifications.....	699
Table 28-2 QSPI Pin.....	700
Table 28-3 QSPI Bus Reference Clock Selection List	706
Table 28-4 List of Reference Commands.....	710
Table 28-5 Pin Status of QIO2 and QIO3	724
Table 28-6 List of QSPI Registers	727
Table 29-1 I2S Main Features.....	735
Table 29-2 I2S Pin Description	736
Table 29-3 Audio Frequency Accuracy (for VCO input frequency=1MHz)	744
Table 29-4 I2S interrupt request	747
Table 29-5 List of I2S Registers.....	751
Table 30-1 CAN Pin Descriptions	757
Table 30-2 CAN bit time setting rules	759
Table 30-3 Software Reset Range Table	766
Table 30-4 CAN Interrupt Table	771
Table 30-5 List of CAN registers.....	772
Table 30-6 CAN Register BYTE/HALFWORD/WORD Access Arrangement Table	772
Table 30-7 Standard Format CAN Receiving Mailbox Format	774
Table 30-8 Extended Format CAN Receive Mailbox Format.....	775
Table 30-9 Standard Format CAN Sending Mailbox Format.....	776
Table 30-10 Extended Format CAN Send Mailbox Format.....	777
Table 31-1 USBFS Pin Descriptions	802
Table 31-2 USBFS interrupt/Event Table.....	820
Table 31-3 USBFS System Control Register List	845
Table 31-4 USBFS System Control Register List	846
Table 34-1 CRC register list.....	936
Table 36-1 SWJ Debug Port Pins	972
Table 36-2 Flexible SWJ-DP pin assignment	972

Figure List

Figure 2-1	Bus architecture	51
Figure 3-1	Power-on Reset	56
Figure 3-2	NRST Reset Timing	57
Figure 3-3	Brownout Reset.....	57
Figure 3-4	Programmable Voltage Detection 1 Reset	58
Figure 3-5	Programmable Voltage Detection 2 Reset	59
Figure 3-6	Watchdog and Dedicated Watchdog Reset	60
Figure 3-7	Power-down Wake-up Reset.....	61
Figure 3-8	Software Reset.....	61
Figure 3-9	MPU Error Reset.....	61
Figure 3-10	RAM Parity Reset.....	62
Figure 3-11	RAMECC Reset	62
Figure 3-12	Clock Frequency Abnormal Reset.....	63
Figure 3-13	External High-Speed Oscillation Abnormal Reset.....	63
Figure 4-1	Clock System Block Diagram	68
Figure 4-2	Clock Frequency Measurement Block Diagram.....	69
Figure 4-3	External High-Speed Oscillator Connection Example.....	72
Figure 4-4	Example Connection Diagram Of External Clock Input.....	73
Figure 4-5	Example of External High-Speed Oscillator Fault Detection.....	74
Figure 4-6	XTAL Is Selected As The System Clock, And XTAL Oscillation Fault Is Detected.....	75
Figure 4-7	External Low-speed Oscillator Connection Example	76
Figure 4-8	Clock Frequency Measurement Timing Diagram.....	83
Figure 5-1	Power supply diagram	112
Figure 5-2	Power-on Rset, power-off Reset Waveforms.....	113
Figure 5-3	Brownout Reset Waveform.....	114
Figure 5-4	PVD1 Interrupt/Reset Block Diagram	116
Figure 5-5	Interrupt/Reset Block Diagram.....	116
Figure 5-6	Power Monitor 1 Interrupt Timing Diagram.....	117
Figure 5-7	Power Monitor 1 Reset Timing Diagram	117
Figure 5-8	Power Supply Monitoring 2 Interrupt Operation Timing Diagram.....	118
Figure 5-9	Power supply monitoring 2 reset operation sequence diagram.....	118
Figure 5-10	Internal Voltage Sampling Diagram	119
Figure 5-11	PTWK _n structure block diagram.....	130
Figure 7-1	FLASH Address Structure (512KB product)	169
Figure 7-2	FLASH Address Structure (256KB product)	170
Figure 7-3	Start sector swap function 1.....	180

Figure 7-4 Start swap function 2.....	180
Figure 9-1 Port Basic Structure Diagram.....	200
Figure 10-1 Interrupt System Block Diagram.....	221
Figure 10-2 Interrupt event selection	247
Figure 10-3 Schematic diagram of digital filter operation	249
Figure 12-1 KEYS CAN System Block Diagram.....	286
Figure 12-2 Schematic Diagram of Keyboard Scanning Function.....	288
Figure 14-1 DMA Structure Diagram.....	304
Figure 14-2 Chain transmission diagram	308
Figure 14-3 Schematic Diagram of Discontinuous Address Transmission	309
Figure 14-4 Schematic Diagram of Discontinuous Reset	311
Figure 14-5 Application Example 1: Memory-to-Memory Transfer	314
Figure 14-6 Application Example 2: Transfer from Memory to Peripheral Circuits	315
Figure 15-1 CMP, 8bitDAC function connection diagram	339
Figure 15-2 Schematic diagram of CMP work	340
Figure 15-3 Schematic Diagram of Scanning Mode Action	342
Figure 16-1 ADC Block Diagram.....	352
Figure 16-2 Internal Analog Channel Selection.....	354
Figure 16-3 Sequence A Single Sweep Mode	356
Figure 16-4 Continuous Scanning	357
Figure 16-5 Double sequence scan mode (sequence A restarts from interrupted channel)	359
Figure 16-6 Double sequence scan mode (sequence A restarts from the first channel).....	359
Figure 16-7 Analog Watchdog Protection Area (comparison conditions).....	360
Figure 16-8 A/D conversion time	361
Figure 16-9 Conversion action when averaging function is valid.....	363
Figure 16-10 Schematic diagram of PGA work.....	364
Figure 16-11 Single Parallel Trigger Mode (Three ADCs)	365
Figure 16-12 Single delay trigger mode (three ADCs)	366
Figure 16-13 Cyclic Parallel Trigger Mode (Three ADCs)	368
Figure 16-14 Cyclic Delay Trigger Mode (Two ADCs)	369
Figure 16-15 Cyclic Delay Trigger Mode (Three ADCs).....	369
Figure 16-16 ADC Interrupt and Event Output Timing	371
Figure 17-1 OTS functional block diagram.....	392
Figure 18-1 Timer6 basic block diagram.....	398
Figure 18-2 Sawtooth Waveform (Counting Up).....	399
Figure 18-3 Triangle Waveform	399
Figure 18-4 Compare Output Action	400
Figure 18-5 Capturing Input Actions	401

Figure 18-6 Filtering Function of Capture Input Port	403
Figure 18-7 Software Synchronization Action	404
Figure 18-8 Hardware Synchronous Action	406
Figure 18-9 Single-buffer mode comparison output timing	408
Figure 18-10 Double buffer capture input timing.....	409
Figure 18-11 Counting Buffer Action in Sawtooth Wave Mode.....	410
Figure 18-12 Counting Buffer Action in Triangular Wave A Mode.....	411
Figure 18-13 Counting buffer action in triangular wave B mode	412
Figure 18-14 TIM6_<t>_PWMA output PWM wave.....	414
Figure 18-15 Triangle wave A mode software setting GCMBR complementary PWM wave output ...	415
Figure 18-16 Triangle wave B mode hardware setting GCMBR complementary PWM wave output (symmetrical dead zone)	416
Figure 18-17 6-phase PWM wave.....	417
Figure 18-18 Three-phase complementary PWM wave output with dead time in triangular wave A mode.....	418
Figure 18-19 Position Mode - Basic Counting.....	419
Figure 18-20 Position Counting Mode - Phase Difference Counting (1 times counting)	420
Figure 18-21 Position Counting Mode - Phase Difference Counting (2 times counting)	420
Figure 18-22 Position Counting Mode - Phase Difference Counting (4 times counting)	420
Figure 18-23 Position Counting Mode - Direction Counting.....	421
Figure 18-24 Revolution Counting Mode – Z-Phase Counting	421
Figure 18-25 Revolution Counting Mode - Position Overflow Counting	422
Figure 18-26 Revolution Counting Mode - Mixed Counting	422
Figure 18-27 Revolution Counting Mode - Mixed Counting Z-Phase Shielding Action Example 1.....	423
Figure 18-28 Revolution Counting Mode - Mixed Counting Z-Phase Shielding Action Example 2.....	424
Figure 18-29 Cycle Interval Valid Request Signal Action.....	425
Figure 18-30 Example Of Interrupt & Event Output In Sawtooth Wave Mode.....	435
Figure 19-1 Timer4 Basic Block Diagram	462
Figure 19-2 Timer4 Sawtooth Waveform.....	463
Figure 19-3 Timer4 Triangle Wave Waveform	463
Figure 19-4 Timer4 sawtooth wave mode counting action	463
Figure 19-5 Timer4 Triangular Wave Mode Counting Action	464
Figure 19-6 Waveform Output Example in Sawtooth Mode.....	465
Figure 19-7 Triangle Wave Mode Waveform Output Example	465
Figure 19-8 Modify the sawtooth count period when the cache is invalid	466
Figure 19-9 Modify the sawtooth count period when the cache is enabled	467
Figure 19-10 Modify The Triangular Wave Count Period When The Cache Is Enabled.....	467

Figure 19-11 OCCR Buffer Data Transmission (When The Cycle Interval Response Link Is Disabled)	468
Figure 19-12 OCCR Buffer Data Transmission (Period Interval Response Link Enable)	469
Figure 19-13 Output Compare Buffer Data Transfer (OCMR Buffer Enabled)	470
Figure 19-14 SCCR Buffer Transmission Operation (Period Interval Response Link Transmission Disabled).....	471
Figure 19-15 SCCR Buffer Transmission Operation (When Periodic Interval Response Link Transmission Is Enabled).....	472
Figure 19-16 Sawtooth Wave Independent PWM Output Example.....	473
Figure 19-17 Triangle Wave Independent PWM Output Example	473
Figure 19-18 Sawtooth Wave Extended PWM Output	474
Figure 19-19 Software Complementary PWM Output	475
Figure 19-20 Complementary PWM output in dead-band timer mode.....	476
Figure 19-21 Waveform output in dead-band timer mode when the pulse width is abnormal.....	476
Figure 19-22 Complementary PWM output in dead-band timer Filtering mode	477
Figure 19-23 Cycle Interval Response Timing Diagram	478
Figure 19-24 Dedicated Event Output Signal Periodic Interval Response Output	479
Figure 19-25 Output Timing of Dedicated Event Output Signal in Delayed Start Mode.....	482
Figure 21-1 TimerA Basic Block Diagram.....	516
Figure 21-2 Sawtooth Waveform (Counting Up).....	517
Figure 21-3 Triangle Waveform	517
Figure 21-4 Compare Output Action	518
Figure 21-5 Capturing Input Actions	519
Figure 21-6 Software Synchronization Action	521
Figure 21-7 Filtering function of the clock input port	522
Figure 21-8 Cache action in sawtooth wave mode	523
Figure 21-9 32-bit Cascade Counting Action.....	524
Figure 21-10 General PWM Output Example.....	525
Figure 21-11 Position Mode - Basic Counting	527
Figure 21-12 Position Counting Mode - Phase Difference Counting (1 Times Counting)	527
Figure 21-13 Position Counting Mode - Phase Difference Counting (2 Times Counting)	528
Figure 21-14 Position Counting Mode - Phase Difference Counting (4 Times Counting)	528
Figure 21-15 Position Counting Mode - Direction Counting	528
Figure 21-16 Revolution Counting Mode - Z Phase Counting	529
Figure 21-17 Revolution Counting Mode - Position Overflow Counting	530
Figure 21-18 Revolution Counting Mode - Mixed Counting	530
Figure 22-1 Timer0 Basic Block Diagram.....	546
Figure 22-2 Timer0 Counting Timing Diagram	548

Figure 23-1 Basic block diagram of RTC.....	556
Figure 24-1 Hardware Startup Example.....	579
Figure 24-2 Software Startup Example	580
Figure 24-3 Examples of refresh action timing (action confirmation, refresh request signal falling edge, etc.).....	581
Figure 24-4 Examples Of Counter Underflow.....	583
Figure 24-5 Examples of Counter Refresh.....	584
Figure 25-1 USART System Block Diagram	590
Figure 25-2 UART Data Format	592
Figure 25-3 UART send data legend 1.....	594
Figure 25-4 UART send data legend 2.....	594
Figure 25-5 USART internal synchronization and sampling timing.....	595
Figure 25-6 UART Received Data Legend 1.....	597
Figure 25-7 UART Received Data Legend 2.....	597
Figure 25-8 Multiprocessor Communication Diagrams	601
Figure 25-9 Multiprocessor Mode Data Format	601
Figure 25-10 Diagram of Multi-Processor Mode Send Data.....	602
Figure 25-11 Diagram of Multiprocessor Mode Receive Data 1	603
Figure 25-12 Diagram of Multiprocessor Mode Receive Data 2	604
Figure 25-13 Smart Card Pattern Connected Legend.....	605
Figure 25-14 Smart Card Pattern synchronization and sampling timing Figure	606
Figure 25-15 Smart Card Pattern synchronization and sampling timing Figure	606
Figure 25-16 Smart Card Pattern Sending Data Legend	608
Figure 25-17 Smart card Pattern Receive data legend	608
Figure 25-18 Clock Synchronization Mode Data Format	610
Figure 25-19 Diagram of Clock Synchronization Mode Sending Data 1	612
Figure 25-20 Diagram of Clock Synchronization Mode Sending Data 2	612
Figure 25-21 Diagram of Clock Synchronization Mode Receive Data 1.....	613
Figure 25-22 Diagram of Clock Synchronization Mode Receive Data 2.....	614
Figure 26-1 I2C System Block Diagram	630
Figure 26-2 Example of I2C Bus Structure	631
Figure 26-3 Sequence Diagram of I ² C bus Timing Diagram.....	632
Figure 26-4 I2C Data format of I ² C bus	633
Figure 26-5 Sequence Diagram of Host Sending Data in 7-bit Address Format (Example).....	634
Figure 26-6 Sequence Diagram of Received Data for Host in 7-bit Address Format (Example)	635
Figure 26-7 Sequence Diagram of Slave Send Mode in 7-bit Address Format (Example)	636
Figure 26-8 Sequence Diagram of 7-bit Address Format Slave Receive Mode (Example)	637
Figure 26-9 SSCL Synchronization Time Series.....	638

Figure 26-10	Sending Sequence Diagram of Slave (1)	639
Figure 26-11	Sending Sequence Diagram of Slave (2)	640
Figure 26-12	Timing When Selecting 7-Bit Address Format	642
Figure 26-13	Timing When Selecting 10-Bit Address Format	643
Figure 26-14	Digital Filtering Circuit Block Diagram	650
Figure 27-1	System Block Diagram.....	670
Figure 27-2	Host mode structure	672
Figure 27-3	Three-Wire Clock Synchronization Operation.....	672
Figure 27-4	Data Format.....	674
Figure 27-5	Data Transfer Format Diagram (CPHA = 0)	675
Figure 27-6	Data Transfer Format (CPHA = 1)	676
Figure 27-7	Full Duplex Synchronous Serial Communication.....	677
Figure 27-8	Sending Only	678
Figure 27-9	Parity Checking Process.....	686
Figure 27-10	Overload Error Handling	688
Figure 27-11	Schematic Diagram Of The Action When The Clock Automatic Stop Function Is Enabled (CPHA=1).....	689
Figure 27-12	Schematic Diagram Of The Action When The Clock Automatic Stop Function Is Enabled (CPHA=0).....	690
Figure 27-13	Parity Error.....	691
Figure 28-1	Module composition diagram of QSPI	700
Figure 28-2	Default Area Setting and AHB Bus Space Memory Mapping Relationship Diagram	701
Figure 28-3	QSPI-ROM Space Memory Map	702
Figure 28-4	Schematic Diagram of Extended SPI Protocol Action 1 (Fast Read Mode).....	703
Figure 28-5	Schematic Diagram Of Extended SPI Protocol Action 2 (Wire Input Fast Read Mode)	703
Figure 28-6	Schematic diagram of two-wire SPI protocol action (fast read mode)	704
Figure 28-7	Schematic Diagram Wire SPI Protocol Action (Fast Read Mode).....	704
Figure 28-8	Basic Timing Diagram of the Serial Interface.....	705
Figure 28-9	Schematic Diagram Of Output Clock Duty Ratio Correction When HCLK Is Divided By Three As The Reference Clock.....	707
Figure 28-10	Schematic Diagram Of QSSL Establishment Time Configuration	708
Figure 28-11	Schematic Diagram Of QSSN Hold-on Time Configuration.....	708
Figure 28-12	Schematic Diagram Of Data Receiving Delay.....	709
Figure 28-13	Schematic Diagram Of Standard Read Bus Cycle	711
Figure 28-14	Schematic Diagram Of Fast Read Bus Cycle	712
Figure 28-15	Schematic diagram of aSelect XIP Mode Four Fast Read Bus Cycle	712
Figure 28-16	Schematic Diagram Of Fast Read Bus Cycle	713
Figure 28-17	Schematic Diagram of Select XIP Mode Four Fast Read Bus Cycle.....	713

Figure 28-18 Schematic Diagram Of Fast Read Bus Cycle	714
Figure 28-19 Schematic Diagram of Select XIP Mode Four Fast Read Bus Cycle.....	715
Figure 28-20 Schematic Diagram of Fast Read Bus Cycle.....	716
Figure 28-21 Schematic Diagram Of Select XIP Mode Four Fast Read Bus Cycle	716
Figure 28-22 Schematic Diagram Of Fast Read Bus Cycle	717
Figure 28-23 Schematic Diagram of Select XIP Mode Four Fast Read Bus Cycle.....	717
Figure 28-24 Enter 4-Byte Mode Command Bus Cycle Diagram	718
Figure 28-25 Exit 4-Byte Mode Command Bus Cycle Diagram	718
Figure 28-26 Schematic Diagram of Write Permission Command Bus Cycle	719
Figure 28-27 Schematic Diagram Of A Single Flash Memory Data Read Operation With Independent Conversion.....	720
Figure 28-28 Schematic Diagram Of Data Reading Operation When The Pre-Reading Function is Valid	720
Figure 28-29 Schematic Diagram Of Data Read Operation Using QSPI Bus Cycle Extension Function	722
Figure 28-30 Schematic Diagram Of XIP Mode Control.....	722
Figure 29-1 I2S System Block Diagram.....	736
Figure 29-2 I2S Philips protocol waveform (16/32 bit full precision)	738
Figure 29-3 I2S Philips Protocol Waveform (16-Bit Data Encapsulated In 32-Bit Frame)	738
Figure 29-4 I2S Philips Protocol Waveform (24-Bit Data Encapsulated in 32-Bit Frame)	739
Figure 29-5 I2S MSB Protocol Waveform (16/32 Bit Full Precision).....	739
Figure 29-6 I2S MSB Protocol Waveform (16-Bit Data Encapsulated In 32-Bit Frame)	739
Figure 29-7 I2S MSB Protocol Waveform (24-Bit Data Encapsulated In 32-Bit Frame)	740
Figure 29-8 I2S LSB Protocol Waveform (16/32 Bit Full Precision).....	740
Figure 29-9 I2S LSB Protocol Waveform (16-Bit Data Encapsulated In 32-Bit Frame)	741
Figure 29-10 I2S LSB Protocol Waveform (24-Bit Data Encapsulated In 32-Bit Frame)	741
Figure 29-11 I2S PCM Protocol Waveform (16/32 Bit Full Precision).....	741
Figure 29-12 I2S PCM Protocol Waveform (16 Bit Data Encapsulated in 32-Bit Frames).....	742
Figure 29-13 I2S PCM Protocol Waveform (24-Bit Data Encapsulated in 32-Bit Frame)	742
Figure 29-14 Audio Sampling Frequency Definition	743
Figure 29-15 Clock Generator Architecture.....	743
Figure 29-16 The Host Only Receives And Temporarily Stops Receiving.....	748
Figure 29-17 PCM Short Frame Host Resends After The Suspension Of Transmission Method 1.....	749
Figure 29-18 PCM Short Frame Host Resends After The Suspension Of Transmission Method 2.....	750
Figure 30-1 CAN System Block Diagram.....	757
Figure 30-2 CAN bit time definition diagram	758
Figure 30-3 CAN TBUF Register Write Transmit Buffer And Schematic Diagram	759
Figure 30-4 Schematic Diagram Of CAN RBUF Register Read Receive Buffer.....	760

Figure 30-5 Schematic Diagram Of CAN ACF Register Access Filter Group.....	761
Figure 30-6 Schematic Diagram of CAN LBMI and LBME	765
Figure 31-1 USBFS System Block Diagram	802
Figure 31-2 USBFS Host Mode System Construction Diagram	804
Figure 31-3 USBFS Device Mode System Construction Diagram	809
Figure 31-4 Schematic Diagram Of USBFS Dynamically Updating The USBFS_HFIR Register.....	814
Figure 31-5 Schematic Diagram Of FIFO Architecture In USBFS Host Mode.....	815
Figure 31-6 Schematic Diagram of FIFO Architecture in USBFS Device Mode.....	817
Figure 32-1 AES Encryption Flow Diagram.....	906
Figure 32-2 HASH Algorithm Flowchart.....	912
Figure 32-3 TRNG System Block Diagram.....	919
Figure 34-1 CRC Module Block Diagram	933
Figure 36-1 Commissioning Control System	969
Figure 36-2 Commissioning control system.....	970
Figure 36-3 JTAG-DP to SW-DP Switching Timing	971
Figure 36-4 TPIU Block Diagram	978

Overview

This series is a high-performance MCU based on ARM® Cortex®-M4 32-bit RISC CPU with a maximum operating frequency of 200MHz. 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 HC32F460_F45x_A460 series integrates high-speed on-chip memory, including a maximum of 512KB of Flash and a maximum of 192KB 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 master includes CPU, DMA, USB dedicated DMA, etc. In addition to the bus matrix, it supports data transfer between peripherals, basic arithmetic operations and mutual triggering of events, which can significantly reduce the transaction processing load of the CPU.

This series integrates rich peripheral functions. Including 2 independent 12bit 2.5MSPS ADCs, 1 adjustable gain PGA, 3 voltage comparators (CMP), 3 multifunctional 16bit PWM Timers (Timer6) supporting 6 complementary PWM outputs, 3 motor PWM Timers (Timer4) Support 18 complementary PWM outputs, 6 16bit general-purpose Timers (TimerA) support 3 3-phase quadrature encoding inputs and 48 Duty independent PWM outputs, 11 serial communication interfaces (I2C/UART/SPI), 1 QSPI interface, 1 channel CAN, 4 I2S supports audio PLLs, 2 SDIOs, 1 USB FS Controller with on-chip FS PHY supporting Device/Host.

This series supports wide voltage range (1.8-3.6V), wide temperature range (-40-105 °C) and low power mode. In Run mode and Sleep mode, you can switch between ultra-high speed mode ($\leq 200\text{MHz}$), high-speed mode ($\leq 168\text{MHz}$) and ultra-low speed mode ($\leq 8\text{MHz}$). Supports fast wake-up in low power consumption mode, the fastest wake-up to STOP mode is 2us, and the fastest wake-up to Power-down mode is 20us.

Typical Application

This series provides 48pin, 64pin, 100pin LQFP package, 32pin, 48pin, 60pin QFN package, 100pin VFBGA package, suitable for automotive electronics, high-performance motor frequency control, intelligent hardware, IoT connection modules and other fields.

About this manual

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

1 Memory Mapping

1.1 Memory Mapping

This MCU supports a 4GB linear address space, with addresses from 0x0000_0000 to 0xFFFF_FFFF.
For memory map details, see Table 1-1.

Table 1-1 Memory-Map

Memory classification		Start address	End address	Space size	Module^{*3}	Protection^{*4}	Note
System	Private Peripheral External Bus	0xE010_0000	0xFFFF_FFFF	511MB	Reserved		Custom space
		0xE00F_F000	0xE00F_FFFF	4KB	ROMTABLE		Debug Control Register Area The MCU has no ETM
		0xE004_2400	0xE00F_EFFF	755KB			
		0xE004_2000	0xE004_23FF	1KB	DBG		
		0xE004_1000	0xE004_1FFF	4KB	ETM		
		0xE004_0000	0xE004_0FFF	4KB	TPIU		
	Private Peripheral Internal Bus	0xE000_F000	0xE003_FFFF	196KB			System control space NVIC MPU, etc.
		0xE000_E000	0xE000_EFFF	4KB	SCS		
		0xE000_3000	0xE000_DFFF	44KB			
		0xE000_2000	0xE000_2FFF	4KB	FPB		
		0xE000_1000	0xE000_1FFF	4KB	DWT		
		0xE000_0000	0xE000_0FFF	4KB	ITM		
External equipment	-	0xA000_0000	0xDFFF_FFFF	1024MB	Reserved		
External RAM	AHB5 Clock: HCLK	0x9800_0000	0x9FFF_FFFF	128MB	QSPI		
		0x6000_0000	0x97FF_FFFF	896MB	Reserved		
Peripheral equipment	-	0x4400_0000	0x5FFF_FFFF	448MB	Reserved		Reserved outside of CPU
		0x4200_0000	0x43FF_FFFF	32MB	PeriBitBand		
		0x4010_0000	0x41FF_FFFF	31MB	Reserved		
	AHB3 Clock: PCLK1	0x400C_0000	0x400F_FFFF	256KB	USBFS		
	AHB2 Clock: PCLK1	0x4007_0800	0x400B_FFFF	318KB	BLANK		
		0x4007_0400	0x4007_07FF	1KB	CAN		
		0x4007_0000	0x4007_03FF	1KB	SDIOC_2		

Memory classification		Start address	End address	Space size	Module ^{*3}	Protection ^{*4}	Note
Peripheral equipment	AHB4 Clock: PCLK1	0x4006_FC00	0x4006_FFFF	1KB	SDIOC_1		
		0x4006_0000	0x4006_FBFF	63KB	BLANK		
	AHB1 Clock: HCLK	0x4005_5800	0x4005_FFFF	42KB	BLANK		
		0x4005_5400	0x4005_57FF	1KB	PERIC		
		0x4005_4800	0x4005_53FF	3KB	BLANK		
		0x4005_4000	0x4005_47FF	2KB	SYSC	With protection	
		0x4005_3800	0x4005_3FFF	2KB	GPIO		
		0x4005_3400	0x4005_37FF	1KB	DMA_2		
		0x4005_3000	0x4005_33FF	1KB	DMA_1		
		0x4005_2C00	0x4005_2FFF	1KB	DCU_4		
		0x4005_2800	0x4005_2BFF	1KB	DCU_3		
		0x4005_2400	0x4005_27FF	1KB	DCU_2		
		0x4005_2000	0x4005_23FF	1KB	DCU_1		
		0x4005_1000	0x4005_1FFF	4KB	INTC	with protection	
		0x4005_0C00	0x4005_0FFF	1KB	KEYSCAN		
	APB4 Clock: PCLK3	0x4005_0800	0x4005_0BFF	1KB	RAMIF	with protection	
		0x4005_0400	0x4005_07FF	1KB	BLANK		
		0x4005_0000	0x4005_03FF	1KB	DMPU	with protection	
APB3	APB4 Clock: PCLK3	0x4004_EC00	0x4004_FFFF	5KB	BLANK		
		0x4004_E800	0x4004_EBFF	1KB	I2C_3		
		0x4004_E400	0x4004_E7FF	1KB	I2C_2		
		0x4004_E000	0x4004_E3FF	1KB	I2C_1		
		0x4004_C400	0x4004_DFFF	7KB	BLANK		
		0x4004_C000	0x4004_C3FF	1KB	RTC	with protection	
		0x4004_A800	0x4004_BFFF	6KB	BLANK		
		0x4004_A400	0x4004_A7FF	1KB	OTS		
		0x4004_A000	0x4004_A3FF	1KB	CMP		
		0x4004_9800	0x4004_9FFF	2KB	BLANK		
	APB3	0x4004_9400	0x4004_97FF	1KB	SWDT	with protection	
		0x4004_9000	0x4004_93FF	1KB	WDT	with protection	
		0x4004_8800	0x4004_8FFF	2KB	BLANK		
		0x4004_8400	0x4004_87FF	1KB	FCM		
	APB3	0x4004_8000	0x4004_83FF	1KB	MSTP	with protection	
		0x4004_1400	0x4004_7FFF	27KB	BLANK		

Memory classification		Start address	End address	Space size	Module ^{*3}	Protection ^{*4}	Note
Peripheral equipment	Clock: PCLK4	0x4004_1000	0x4004_13FF	1KB	TRNG	with protection	
		0x4004_0800	0x4004_0FFF	2KB	BLANK		
		0x4004_0400	0x4004_07FF	1KB	ADC_2		
		0x4004_0000	0x4004_03FF	1KB	ADC_1		
	APB2 Clock: PCLK1	0x4002_5000	0x4003_FFFF	108KB	BLANK		
		0x4002_4C00	0x4002_4FFF	1KB	Timer4_3		
		0x4002_4800	0x4002_4BFF	1KB	Timer4_2		
		0x4002_4400	0x4002_47FF	1KB	Timer0_2		
		0x4002_4000	0x4002_43FF	1KB	Timer0_1		
		0x4002_2800	0x4002_3FFF	6KB	BLANK		
		0x4002_2400	0x4002_27FF	1KB	I2S_4		
		0x4002_2000	0x4002_23FF	1KB	I2S_3		
		0x4002_1800	0x4002_1FFF	2KB	BLANK		
		0x4002_1400	0x4002_17FF	1KB	USART_4		
		0x4002_1000	0x4002_13FF	1KB	USART_3		
		0x4002_0800	0x4002_0FFF	2KB	BLANK		
	APB1 Clock: PCLK1	0x4002_0400	0x4002_07FF	1KB	SPI_4		
		0x4002_0000	0x4002_03FF	1KB	SPI_3		
		0x4001_E800	0x4001_FFFF	6KB	BLANK		
		0x4001_E400	0x4001_E7FF	1KB	I2S_2		
		0x4001_E000	0x4001_E3FF	1KB	I2S_1		
		0x4001_D800	0x4001_DFFF	2KB	BLANK		
		0x4001_D400	0x4001_D7FF	1KB	USART_2		
		0x4001_D000	0x4001_D3FF	1KB	USART_1		
		0x4001_C800	0x4001_CFFF	2KB	BLANK		
		0x4001_C400	0x4001_C43F	1KB	SPI_2		
		0x4001_C000	0x4001_C3FF	1KB	SPI_1		
		0x4001_8000	0x4001_BFFF	16KB	Timer6	Counting clock: PCLK0	
		0x4001_7C00	0x4001_7FFF	1KB	EMB		
		0x4001_7400	0x4001_7BFF	2KB	BLANK		
		0x4001_7000	0x4001_73FF	1KB	Timer4_1		
		0x4001_6800	0x4001_6FFF	2KB	BLANK		
		0x4001_6400	0x4001_67FF	1KB	TimerA_6		
		0x4001_6000	0x4001_63FF	1KB	TimerA_5		
		0x4001_5C00	0x4001_5FFF	1KB	TimerA_4		

Memory classification		Start address	End address	Space size	Module ^{*3}	Protection ^{*4}	Note
SRAM	AHB3 Clock: PCLK1	0x4001_5800	0x4001_5BFF	1KB	TimerA_3		
		0x4001_5400	0x4001_57FF	1KB	TimerA_2		
		0x4001_5000	0x4001_53FF	1KB	TimerA_1		
		0x4001_0C00	0x4001_4FFF	17KB	BLANK		
		0x4001_0800	0x4001_0BFF	1KB	AOS		Internal trigger event register area
		0x4001_0400	0x4001_07FF	1KB	EFM	with protection	
		0x4001_0000	0x4001_03FF	1KB	BLANK		
		0x4000_9000	0x4000_FFFF	28KB	BLANK		
		0x4000_8C00	0x4000_8FFF	1KB	CRC	with protection	
		0x4000_8800	0x4000_8BFF	1KB	BLANK		
	SRAM Clock: HCLK	0x4000_8400	0x4000_87FF	1KB	HASH256	with protection	
		0x4000_8000	0x4000_83FF	1KB	AES128	with protection	
		-	0x4000_0000	0x4000_7FFF	32KB	Reserved	
		-	0x2400_0000	0x3FFF_FFFF	448MB	Reserved	
		-	0x2200_0000	0x23FF_FFFF	32MB	SRAMBitBand	Reserved outside of CPU
	CODE	-	0x2010_0000	0x21FF_FFFF	31MB	Reserved	
		-	0x200F_1000	0x200F_FFFF	60KB	BLANK	
		-	0x200F_0000	0x200F_0FFF	4KB	Ret_SRAM	
		-	0x2002_7000	0x200E_FFFF	804KB	BLANK	
		-	0x2002_0000	0x2002_6FFF	28KB	SRAM3	ECCRAM
		-	0x2001_0000	0x2001_FFFF	64KB	SRAM2	
		-	0x2000_0000	0x2000_FFFF	64KB	SRAM1	
	CODE	SRAM Clock: HCLK	0x1FFF_8000	0x1FFF_FFFF	32KB	SRAMH	
		-	0x0318_0000	0x1FFF_7FFF	462.4MB	BLANK	
		Flash Clock: HCLK	0x0317_FFE0	0x0317_FFFF	32B	Data Security Protection	Used to configure data security protection
		-	0x0210_0000	0x0317_FFDF	16.5MB	BLANK	
		REMAP Clock: HCLK	0x0208_0000	0x020F_FFFF	512KB	REMAP1	Address Remapping Area 1
		-	0x0200_0000	0x0207_FFFF	512KB	REMAP0	Address Remapping Area 0
		-	0x0008_0000	0x01FF_FFFF	31.5MB	BLANK	

Memory classification		Start address	End address	Space size	Module ^{*3}	Protection ^{*4}	Note
	Flash Clock: HCLK	0x0000_0000	0x0007_FFFF	512KB	Embedded Flash ^{*5}		

*1 Please refer to ARM Cortex-M4 + Description Manual Memory System.

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

*3 Reserved: Access to the bus will cause a bus error; BLANK: Write access is invalid, and read 0 during read access.

*4 For modules with protection function, only CPU privileged mode access is supported when the protection function is valid. For specific registers and descriptions, refer to the DMPU chapter.

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

1.2 External Space Mapping

The QSPI space is divided into two segments, the QSPI I/O register space is 64MB and the external QSPI device space is 64MB. Please refer to Table 1-2 for distribution relationship

Table 1-2 QSPI Address Space Allocation

QSPI	0x9800_0000	0x9FFF_FFFF	128MB	QSPI I/O registers	0x9C00_0000	0x9FFF_FFFF	64MB
				External QSPI device	0x9800_0000	0x9BFF_FFFF	64MB

1.3 Bit Band Region

The Cortex™-M4F memory map consists of two bit band regions. These regions map each word in the memory alias region to the corresponding bits in the memory bit band region. When you write to an alias region, it 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 are only available for Cortex™ -M4F + access and are not valid for other bus master interfaces such as DMA.

1.4 Address Remapping

This MCU provides 2 remapping addresses, and the memory address remapping function can be configured. The source address can be set as the main flash memory FLASH address and the high-speed SRAM address.

Remap address 0:

0x0200_0000~0x0208_0000 (depending on the remapping space MMF_REMCR0/1.RMSIZE[4:0])

Remap address 1:

0x0208_0000~0x0210_0000 (depending on the remapping space MMF_REMCR0/1.RMSIZE[4:0])

The address correspondence is shown in Table1-3.

Table 1-3 Sample Destination Address Configuration

Register setting	Remap address (CPU address - CPUADDR[31:0])	Source address		
		High 3-bit address	Middle address	Low address
RMSIZE[4:0]=01110 case (remap space: 16K)	0x0200_0000~0x0200_3FFF	all 0	RMTADDR[16:2]	CPUADDR[13:0]
RMSIZE[4:0]=01111 case (remap space: 32K)	0x0200_0000~0x0200_7FFF	all 0	RMTADDR[16:3]	CPUADDR[14:0]
RMSIZE[4:0]=10000 case (remap space: 64K)	0x0208_0000~0x0208_FFFF	all 0	RMTADDR[16:4]	CPUADDR[15:0]
RMSIZE[4:0]=10001 case (remap space: 128K)	0x0208_0000~0x0209_FFFF	all 0	RMTADDR[16:5]	CPUADDR[16:0]

For example, use the function of remapping address 0, set the source address as the main flash memory FLASH address 0x0000_8000, and the remapping space is 32K, and the register MMF_REMCR0 needs to be set to 0x8000_800F.

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

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

1.5 Remap Register

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

Register base address: 0x4001_0500

Table 1-4 Register List

Register name	Symbol	Offset address	Bit width	Reset value
access protection register	MMF_REMPRT	0x0000	32	0x0000_0000
Remap Control Register 0	MMF_REMCR0	0x0004	32	0x0000_0000
Remap Control Register 1	MMF_REMCR1	0x0008	32	0x0000_0000

1.5.1 Access Protection Register (MMF_REMPRT)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MMF_REMPRT[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~16	Reserved	-	Read as "0", write as "0"										R		
b15~0	MMF_REMPRT[15:0]	protection register	Registers MMF_REMCR0 and MMF_REMCR1 are write-protected: First write 0x0123 to MMF_REMPRT[15:0] and then write 0x3210 to release the protection; When the registers MMF_REMCR0 and MMF_REMCR1 are in the write protection state, the read register is 0 When the registers MMF_REMCR0 and MMF_REMCR1 are released from the write protection state, the read register is 1										R/W		

1.5.2 Remap Control Register (MMF_REMCRx) (x=0, 1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
EN	-	RMTADDR[16:4]														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
RMTADDR[3:0]				-						RMSIZE[4:0]						
<hr/>																
Bit	Marking		Place name		Function								Read and write			
b31	EN		remap valid bit		0: no remapping 1: remapping is valid								R/W			
b30~29		Reserved		-		Read as "0", write as "0"								R		
b28~12		RMTADDR[16:0]		source address		The effective number of digits is related to the setting of RMSIZE[4:0]. Settings can be referred to Table1-3.								R/W		
b11~b5		Reserved		-		Read as "0", write as "0"								R		
b4~b0		RMSIZE[4:0]		remap space		Set remapping space 00000~01011: Reserved, setting prohibited 01100: 4KB 01101: 8KB 01110: 16KB 01111: 32KB 10000: 64KB 10001: 128KB 10010: 256KB 10011: 512KB (256KB product settings prohibited) 10100~11111: Reserved, setting prohibited								R/W		

2 Bus Architecture (BUS)

2.1 Overview

The main 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
- 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 32KB) bus
 - System SRAMA (SRAM1 64KB) bus
 - System SRAMB (SRAM2 64KB, SRAM3 28KB, Ret_SRAM 4KB) bus
 - APB1 peripheral bus (EMB/Timers/SPI/USART/I2S)
 - APB2 Peripheral Bus (Timers/SPI/USART/I2S)
 - APB3 peripheral bus (ADC/PGA/TRNG)
 - APB4 peripheral bus (FCM/WDT/CMP/OTS/RTC/WKTM/I2C)
 - AHB1 peripheral bus (KEYSCAN/INTC/DCU/GPIO/SYSC)
 - AHB2 peripheral bus (CAN/SDIOC)
 - AHB3 peripheral bus (AES/HASH/CRC/USBFS)
 - AHB4 peripheral bus (SDIOC)
 - AHB5 peripheral bus (QSPI)

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

2.2 Bus Architecture

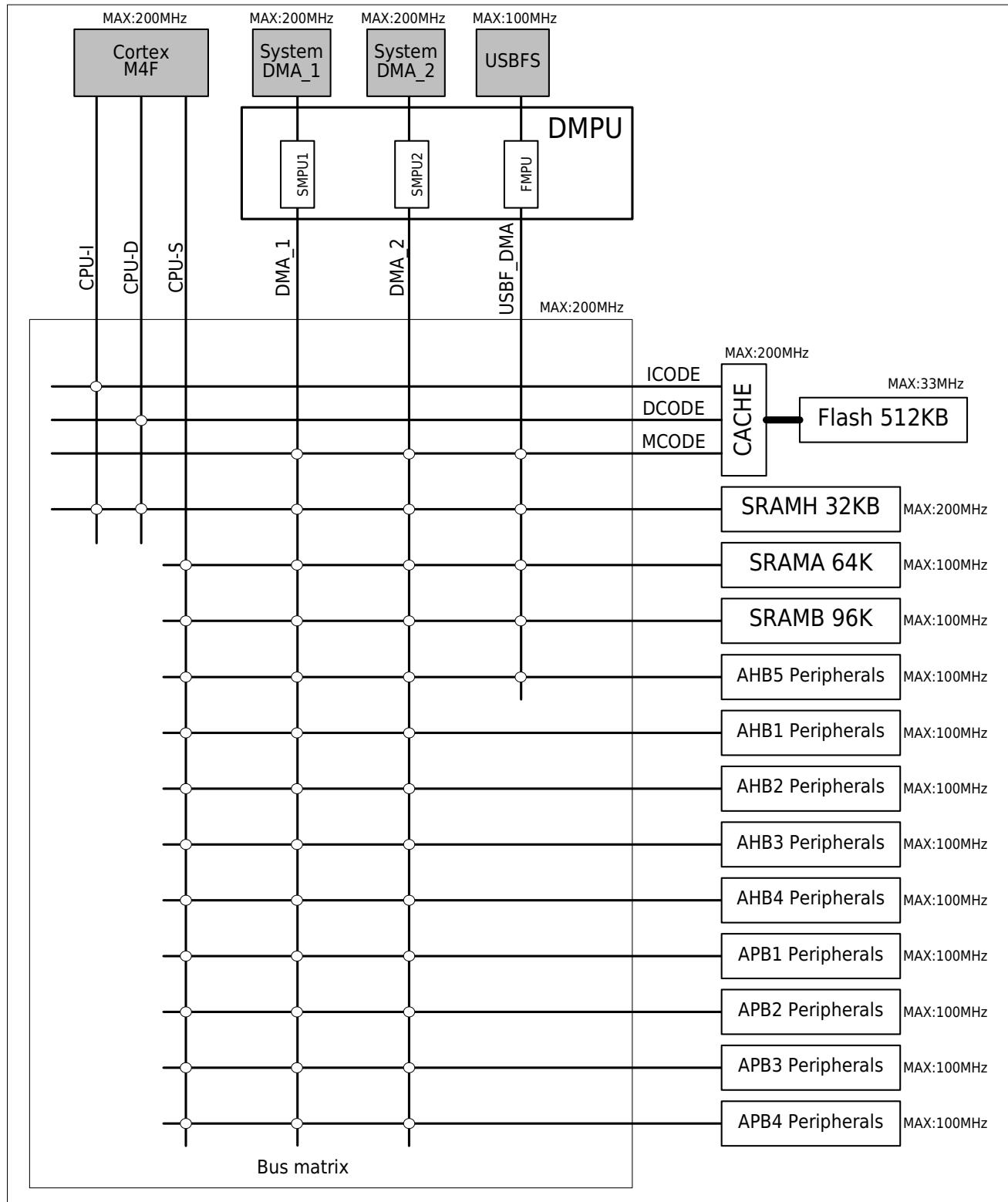


Figure 2-1 Bus architecture

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

- CPU-I bus

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

- CPU-D bus

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

- CPU-S bus

The system bus of the M4F core, the CPU accesses peripherals or system SRAM through this bus, and can also obtain instructions and immediate data through this bus (the efficiency is lower than that through the CPU-I bus and CPU-D bus). The access objects are SRAMA/SRAMB/all peripherals and AHB5 external expansion space.

- 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/SRAMA/SRAMB/all peripherals and AHB5 external expansion space.

- USBFS_DMA bus

DMA dedicated bus for USBFS, through which USBFS accesses all memory spaces. The access object is the external expansion space of Flash/SRAMH/SRAMA/SRAMB/AHB5.

2.3 Bus Function

The bus is responsible for the host's read and writes access to the slave computer. When the operating frequency of the host module is higher than that of the slave module (such as CPU-S accessing RTC), the bus will automatically perform frequency reduction synchronization processing. When the operating frequency of the master module is lower than that of the slave module (such as USBFS_DMA accessing SRAMH), the bus automatically performs up-frequency synchronization processing.

Through the bus matrix, when the access targets of different host buses 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 SRAMA, DMA_2 accesses SRAMB, and USBFS_DMA accesses the external expansion space of AHB5. These accesses can be performed simultaneously.

3 Reset Control (RMU)

3.1 Introduction

The chip is configured with 14 reset modes.

- Power-on Reset (POR)
- NRST pin reset (NRST)
- Brown-out Reset (BOR)
- Programmable Voltage Detect 1 Reset (PWD1R)
- Programmable Voltage Detect 2 Reset (PWD2R)
- Watchdog Reset (WDTR)
- Special watchdog reset (SWDTR)
- Power-down wake-up reset (PDRST)
- Software Reset (SRST)
- MPU Error Reset (MPUR)
- RAM Parity Reset (RAMPR)
- RAMECC reset (RAMECCR)
- Clock exception reset (CKFER)
- External High Speed Oscillator Abnormal Shock Reset (XTALER)

3.2 Reset Mode and Reset Flag Bit

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

Table 3-1 Reset Methods and Generation Conditions

Reset mode	Generating condition
Power-on reset	VCC powered on
Reset pin reset Pin reset is a reset caused by the drive of the NRST pin to a low level.	NRST pin input low level
Brown-out reset	VCC voltage drops below VBOR voltage
Programmable voltage detection 1 reset	VCC voltage drops below PVD1 voltage
Programmable voltage detection 2 reset	VCC voltage drops below PVD2 voltage
watchdog reset	Watchdog timer generates a refresh error or an overflow error
Special watchdog resetting	Refresh error or overflow error occurred in dedicated watchdog
Power-down wake-up reset	By setting the reset generated by the power-down mode, the core wakes up from the reset state after the power-down wake-up event occurs
Software reset	Set the reset register bit (ARM register AIRCR.SYSRESETREQ bit)
MPU error reset	Reset generated by MPU access error
RAM parity reset	Reset generated when a parity error occurs in RAM
RAM ECC error reset	Reset generated when ECC error occurs in RAM
Clock frequency exception reset	When the clock frequency monitoring function (FCM) detects a clock cycle error
External High Speed Oscillator Abnormal Shock Reset	Resetting of External High-speed Oscillator Abnormal Shock

When a reset occurs, the chip will set the corresponding reset flag according to the reset method, and the reset flag is shown in Table 3-2. For example, pin reset occurs, pin reset flag bit PINRF is set to 1, PINRF can be reset by writing CLRF after PINRF is set.

Table 3-2 Reset Methods and Reset Flags

Reset flag	Reset mode									
	External High Speed Oscillator Abnormal Shock Reset	Clock frequency exception reset	RAM ECC error reset	RAM parity error reset	MPU error reset	Software reset	Power-down wake-up reset	Special watchdog resetting	watchdog reset	Voltage detection 2 reset
Power-on reset flag (RMU_RSTF0.PORF)	✓	—	—	—	—	—	—	—	—	—
Pin Reset Flag (RMU_RSTF0.PINRF)	—	✓	—	—	—	—	—	—	—	—
Brown-out reset flag (RMU_RSTF0.BORF)	—	—	✓	—	—	—	—	—	—	—
Programmable voltage sense 1 reset flag (RMU_RSTF0.PVD1RF)	—	—	—	✓	—	—	—	—	—	—
Programmable voltage sense 2 reset flag (RMU_RSTF0.PVD2RF)	—	—	—	—	✓	—	—	—	—	—
Watchdog Reset Flag (RMU_RSTF0.WDRF)	—	—	—	—	—	—	✓	—	—	—
Dedicated watchdog reset flag (RMU_RSTF0.SWDRF)	—	—	—	—	—	—	—	✓	—	—
Power-down wake-up reset flag (RMU_RSTF0.PDRF)	X	—	—	—	—	—	—	—	✓	—
Software reset flag (RMU_RSTF0.SWRF)	X	X	X	—	—	—	—	X	✓	—
MPU error reset (RMU_RSTF0.MPUERF)	X	X	X	—	—	—	—	X	—	✓
RAM parity error reset (RMU_RSTF0.RAPERF)	X	X	X	—	—	—	—	X	—	✓
RAM ECC reset (RMU_RSTF0.RAECRF)	X	X	X	—	—	—	—	X	—	✓
Clock Frequency Abnormal Reset (RMU_RSTF0.CKFERF)	X	X	X	—	—	—	—	X	—	✓
External high-speed oscillator abnormal stop reset(RMU_RSTF0.XTALERF)	X	X	X	—	—	—	—	X	—	✓

✓: set X: clear -: unchanged

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, and the timing is shown in Figure 3-1. Power-on reset occurs when the NRST pin is connected to the power supply at a high level. After a certain period of time (T_{RSTPOR}) after the VCC voltage is higher than the power-on reset voltage V_{POR} , the internal reset of the chip is released, and the CPU starts to execute the code. When generating power - on reset, power - on reset flag RMU_RSTF0.PORF is set. Please refer to [5.3.1 Description of Action of Power-on/Power-off Reset] for details on power-on reset.

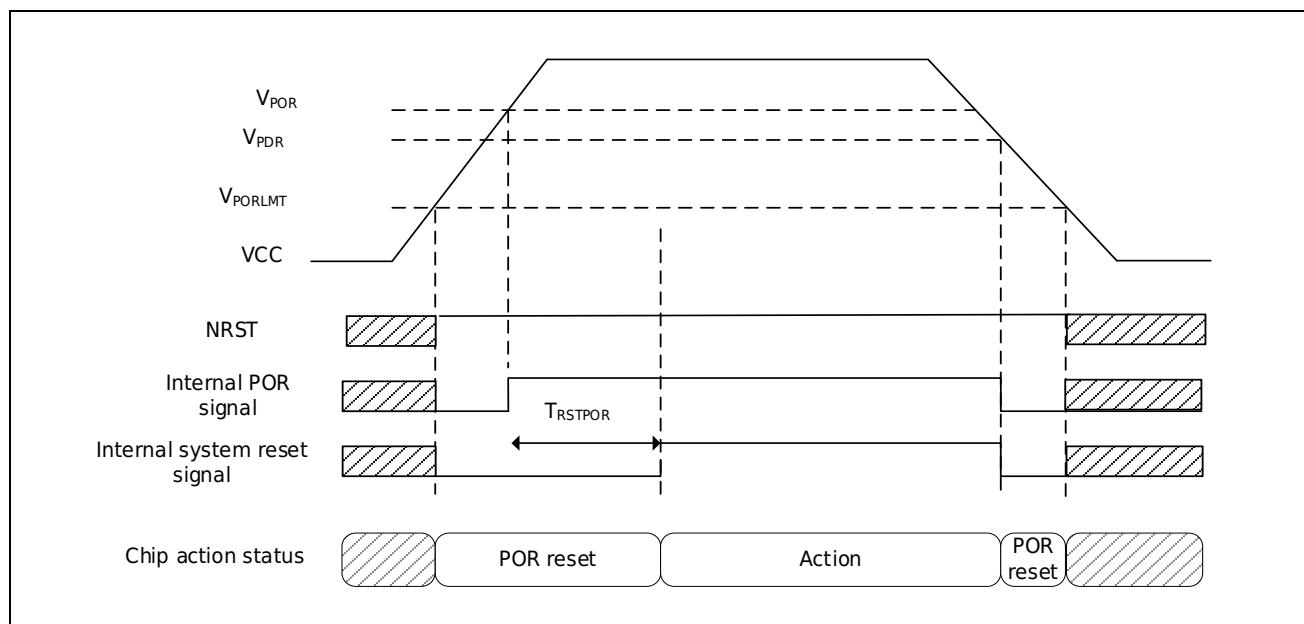


Figure 3-1 Power-on Reset

3.3.2 Reset Pin Reset Pin Reset is a Reset caused by the Drive of the NRST Pin to a Low Level.

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

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

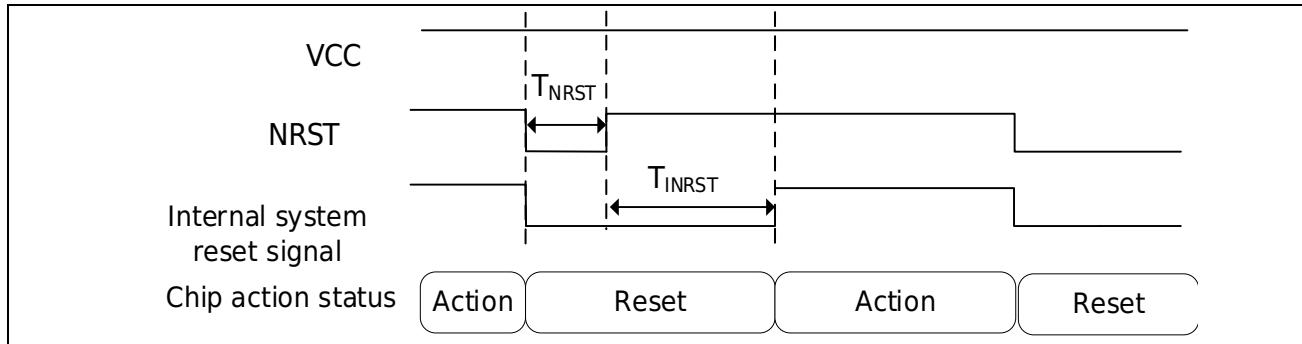


Figure 3-2 NRST Reset Timing

3.3.3 Brown-out Reset

Brown-out reset is an internal reset caused by the voltage monitoring circuit, the timing is shown in Figure 3-3. After the undervoltage is set as reset enable through the ICG register, if the VCC voltage is lower than the monitoring voltage V_{BOR} , RMU_RSTF0.VBORF will be set. When the VCC voltage is higher than the monitoring voltage V_{BOR} , the reset will be released after the V_{BOR} reset time (T_{RSTB0R}).

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

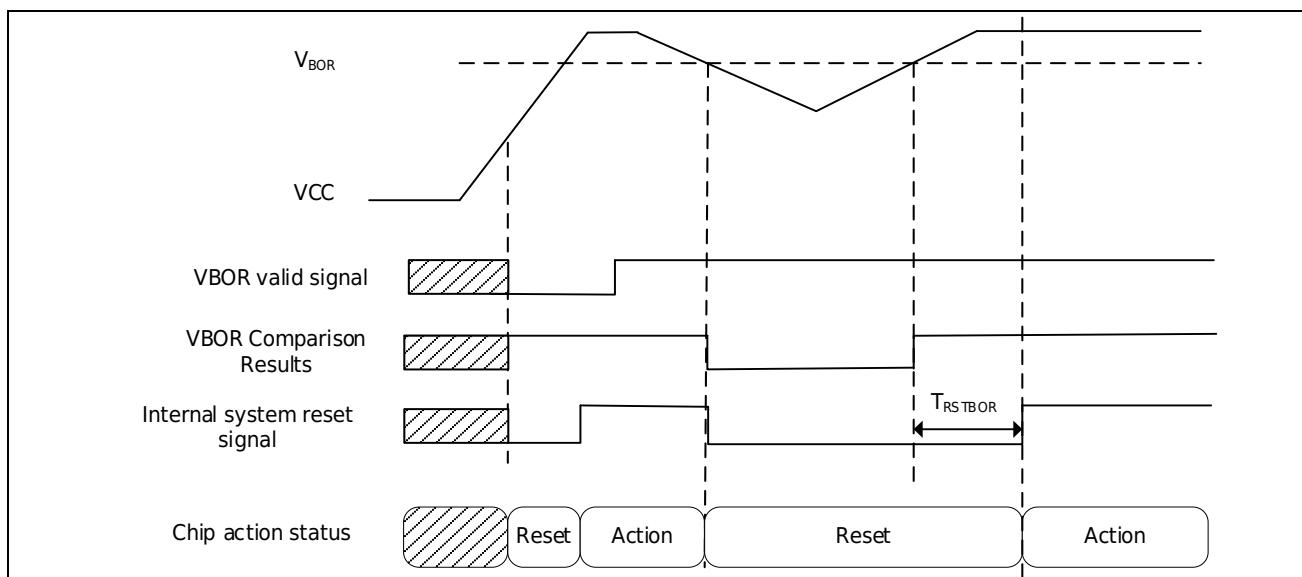


Figure 3-3 Brownout Reset

3.3.4 Programmable Voltage Detection 1 Reset, Programmable Voltage Detection 2 Reset

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

After the programmable voltage detection 1 is enabled and reset is enabled, if VCC is lower than the monitoring voltage of the programmable voltage detection 1, a reset of the programmable voltage detection 1 is generated, and RMU_RSTF0.PVD1F is set. When the VCC voltage is higher than the monitoring voltage of programmable voltage detection 1, the reset will be released after the reset time of PVD1 (T_{IPVD1}).

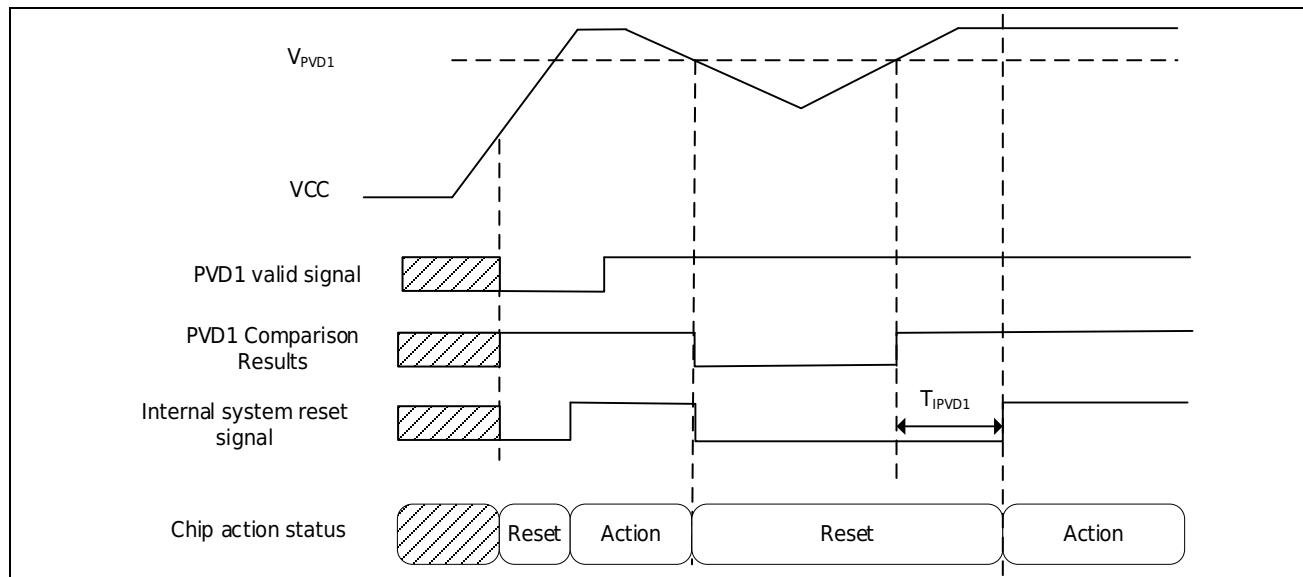


Figure 3-4 Programmable Voltage Detection 1 Reset

After the programmable voltage detection 2 is enabled and reset is enabled, if VCC is lower than the monitoring voltage of the programmable voltage detection 2, a reset of the programmable voltage detection 2 is generated, and RMU_RSTF0.PVD2F is set. When the VCC voltage is higher than the monitoring voltage of programmable voltage detection 2, the reset will be released after the reset time of PVD2 (T_{IPVD2}).

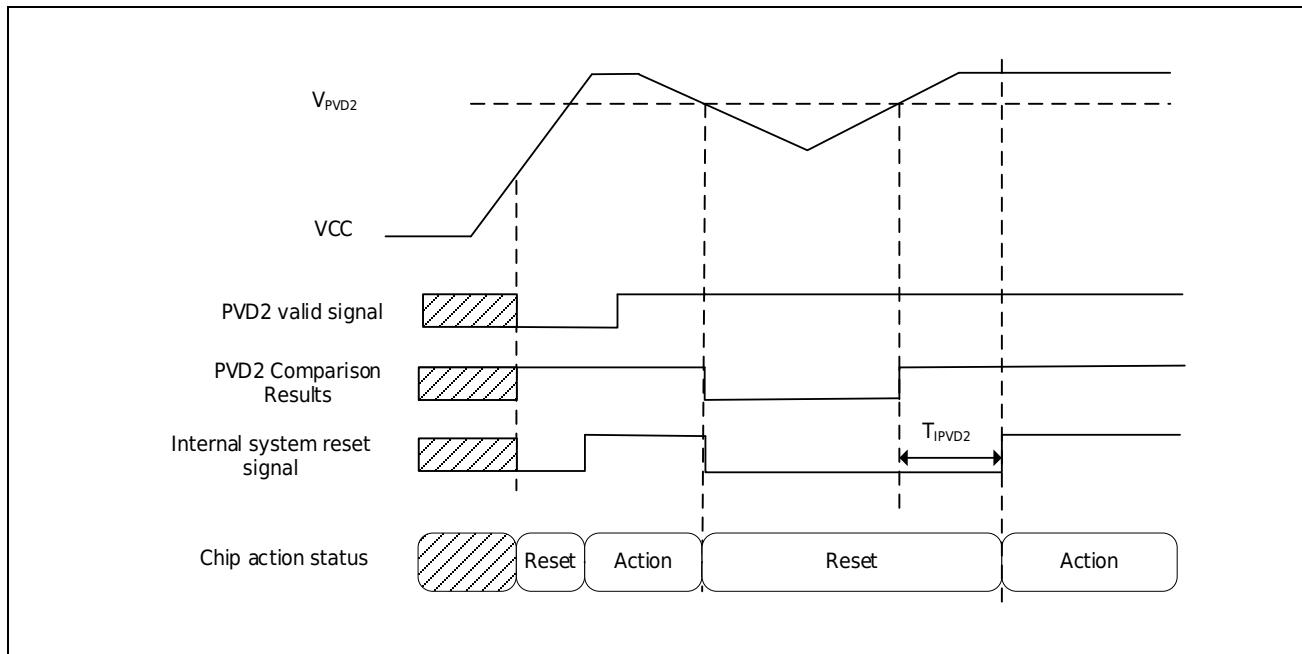


Figure 3-5 Programmable Voltage Detection 2 Reset

For reset settings of programmable voltage detection 1 and programmable voltage detection 2, please refer to [5.3 Power Voltage Detection Unit (PVD) Description].

3.3.5 Watchdog Reset, Dedicated Watchdog Reset

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

After the watchdog reset is set to be valid, the watchdog reset will be generated when the watchdog timer underflows or the write operation is not performed during the refresh period. A watchdog reset sets RMU_RSTF0.WDRF. After the watchdog reset is generated, the chip is released from reset after the internal reset time T_{RIPT} .

After the dedicated watchdog reset is set to be valid, the watchdog reset will be generated when the dedicated watchdog timer underflows or the write operation is not performed during the refresh period. Special watchdog reset RMU_RSTF0.SWDRF setup. After generating special watchdog reset, after internal reset time T_{RIPT} , the chip is released.

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

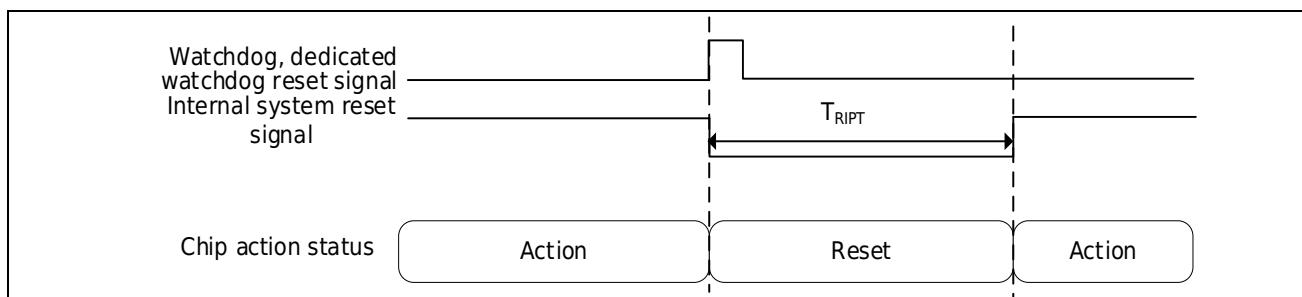


Figure 3-6 Watchdog and Dedicated Watchdog Reset

3.3.6 Power-down Wake-up Reset

Power-down wake-up reset is an internal reset generated when the chip executes the WFI command when PWC_PWRC0.PWDN is set to 1, enters power-down mode, and releases power-down mode through a power-down mode wake-up event. The timing is shown in Figure 3-7. After the power-down mode is released 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, the minimum is in power-down mode 1, and the maximum is in power-down mode 3.

For details about power-down wake-up reset, please refer to [5.4.4 Power Down Mode].

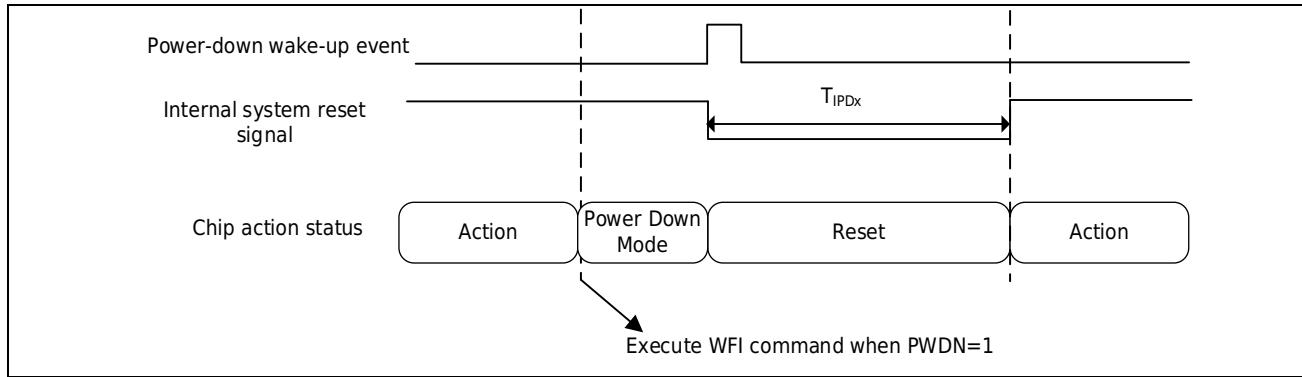


Figure 3-7 Power-down Wake-up Reset

3.3.7 Software Reset

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

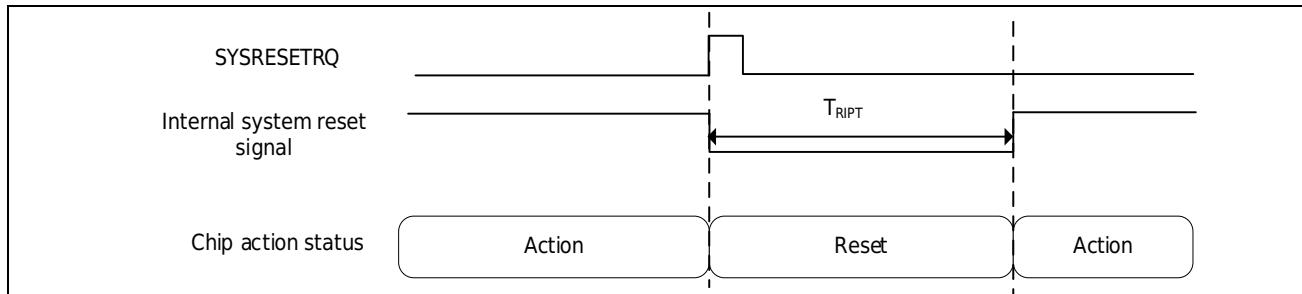


Figure 3-8 Software Reset

3.3.8 MPU Error Reset

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

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

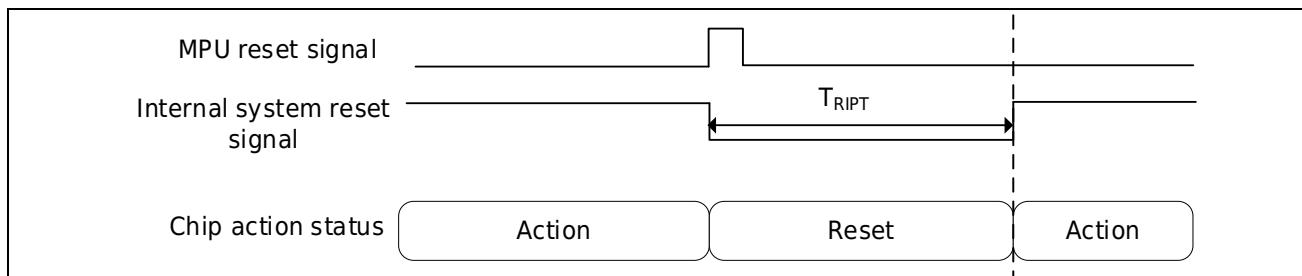


Figure 3-9 MPU Error Reset

3.3.9 RAM Parity Reset

When an error occurs in the RAM parity check, a reset of the RAM parity check is generated, and the timing is shown in Figure 3-10. A RAM parity error sets RMU_RSTF0.RAPERF. After the internal reset time T_{RIPT} , the chip is de-reset.

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

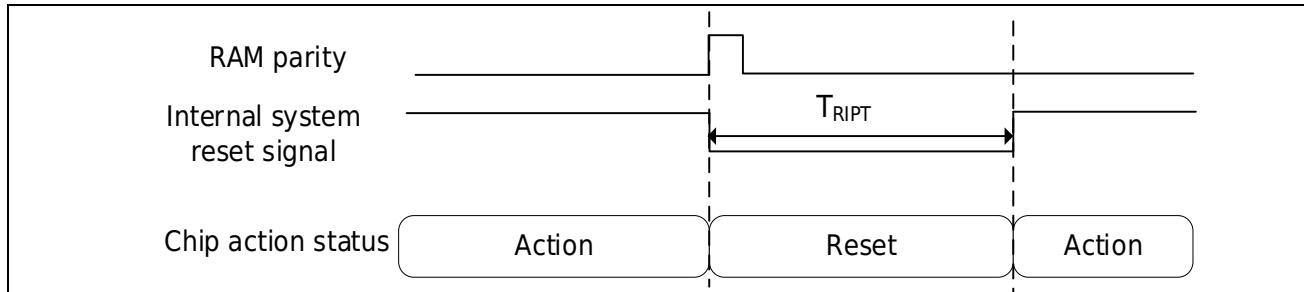


Figure 3-10 RAM Parity Reset

3.3.10 RAMECC Reset

When an error occurs in the RAMECC verification, a RAMECC reset is generated, and the timing is shown in Figure 3-11. A RAMECC reset sets RMU_RSTF0.RAECRF. After the internal reset time T_{RIPT} , the chip is de-reset.

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

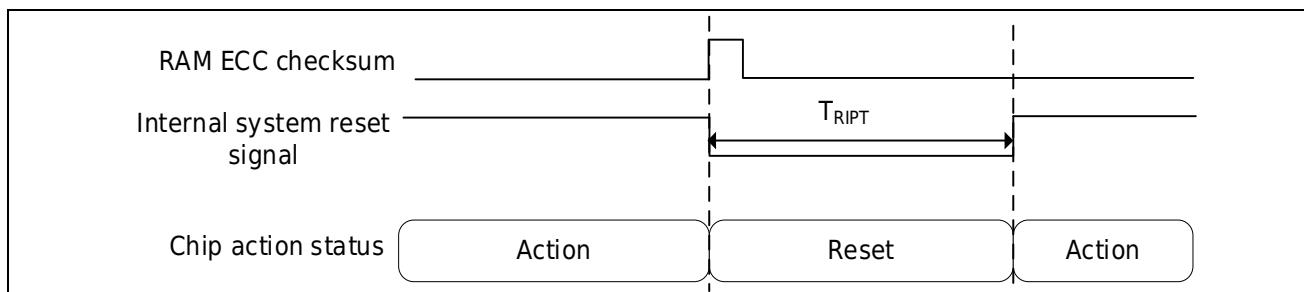


Figure 3-11 RAMECC Reset

3.3.11 Clock Frequency Exception Reset

When the built-in FCM module of the chip detects that the clock frequency is abnormal, if it is set to reset, it will generate a clock frequency abnormal reset. The timing is shown in Figure 3-12. Clock frequency exception reset sets RMU_RSTF0.CKFERF. After the internal reset time T_{RIPT} , the chip is de-reset.

Please refer to [4.10 Clock Frequency Measurement] for setting of clock frequency abnormal reset.

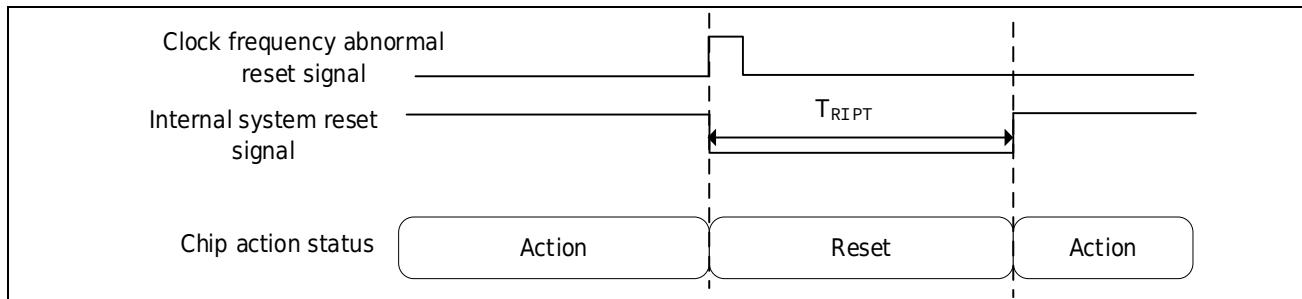


Figure 3-12 Clock Frequency Abnormal Reset

3.3.12 External High Speed Oscillator Abnormal Shock Reset

RMU_RSTF0 is generated when the external high-speed oscillator abnormally stops vibration when the shock stop detection module is valid and the reset is enabled. XTALERF is set. After the internal reset time T_{RIPT} , the chip is de-reset.

Please refer to [4.5.2 External High-Speed Oscillator Fault Detection] for the setting of external high-speed oscillator abnormal stop vibration reset.

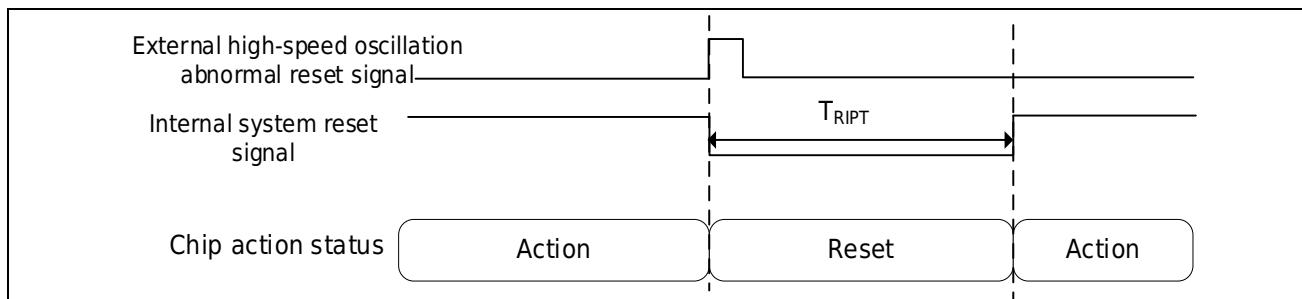


Figure 3-13 External High-Speed Oscillation Abnormal Reset

3.3.13 Judgment of Reset Mode

The reset mode can be judged according to the reset flag of RMU_RSTF0. Multiple reset flags may occur when two or more resets occur simultaneously. Multiple resets occur when the MULTIRF bit in RMU_RSTF0 is 1. After reading RMU_RSTF0, all reset flags can be cleared to 0 by setting the CLRF bit. After setting and clearing RMU_RSTF0, wait at least 6 CPU clock cycles before reading the RMU_RSTF0 register again.

3.3.14 Reset Conditions for Each Module

Modules	Register	reset source
Debug controller(DBGC)	MCUSTPCTL MCUTRACECTL MCUDBGSTAT	1. Power-on reset 2. Power-down wake-up reset
Real Time Clock (RTC)	RTC internal registers	Module software reset control bit: RTC CR0.RESET
Power control (PWC) Clock control (CMU)	PWC_PWRC0 PWC_PWRC1 PWC_PDWKE0 PWC_PDWKE1 PWC_PDWKE2 PWC_PDWKES CMU_XTALCFGR	All reset sources other than the following: 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_PVDCR1 PWC_PVDFCR PWC_PVDCR0 PWC_PWRC2 PWC_PWCMR	1. Power-on reset 2. Pin reduction 3. Brown-out reset 4. watchdog reset 5. Special watchdog resetting 6. Power-down mode 3 wake-up reset
	PWC_PVDICR[0] PWC_PVDDSR.PVD1DETFLG PWC_PVDICR[4] PWC_PVDDSR.PVD2DETFLG	1. Power-on reset 2. Pin reduction 3. Brown-out reset 4. watchdog reset 5. Special watchdog resetting 6. Power-down wake-up reset
	Other than the above	All reset sources
Registers other than the above modules		All reset sources

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

3.4 Register Description

The list of registers is shown Table 3-3 below.

BASE ADDR: 0x400540C0

Table 3-3 List of RMU Registers

Register name	Symbol	Offset address	Bit width	Reset value
Reset status register	RMU_RSTF0	0x00	16	Different Reset Values According to Different Reset Methods

3.4.1 Reset Flag Register0 (RMU _ RSTF0)

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

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CLRF	MULTIRF	XTALERF	CKFERF	RAECRF	RAPERF	MPUERF	SWRF	PDRF	SWDRF	WDRF	PVD2RF	PVD1RF	BORF	PINRF	PORF
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b15	CLRF	Clear reset Mark	The software is set to 1 to clear the reset flag bit. Read at 0. The set action must be performed after RMU _ RSTF0 is read. 0: No operation 1: Reset mark	R/W											
b14	MULTIRF	More than 2 reduction Occurrence flag bit	When two or more resets occur, the hardware is set. Reset by setting CLRF. 0: No reduction of two or more 1: When two or more resets occur	R/W											
b13	XTALERF	External high-speed oscillator exception Stop-and-reset flag	When an external high-speed oscillator abnormally stops and resets, the hardware is set. Reset by setting CLRF. 0: No External High-Speed Oscillator Abnormal Shock Reset 1: External high-speed oscillator abnormal shuffling reset	R/W											
b12	CKFERF	abnormal frequency Reset flag	clock Set by hardware when a clock frequency exception reset occurs. Reset by setting CLRF. 0: No clock frequency exception reset occurs 1: A clock frequency abnormal reset occurs	R/W											
b11	RAECRF	RAMECC reset Mark	Set by hardware when a RAMECC reset occurs. Reset by setting CLRF. 0: No RAMECC reset occurred 1: RAMECC reset occurs	R/W											
b10	RAPERF	RAM parity error Reset flag	When a RAM parity error is reset, the hardware is set. Reset by setting CLRF. 0: No RAM parity error reset 1: RAM parity error reset	R/W											
b9	MPUERF	MPU error reset flag	Set by hardware when an MPU error reset occurs. Reset by setting CLRF. 0: No MPU error reset occurred 1: MPU error reset occurred	R/W											
b8	SWRF	Software reset flag	Set by hardware when a software reset occurs. Reset by setting CLRF. 0: No software reset has occurred 1: A software reset occurred	R/W											
b7	PDRF	Power-down mode reset	Set by hardware when a power-down reset occurs. Cleared by writing to CLRF. 0: Power-down mode reset has not occurred 1: Power-down mode reset occurred	R/W											
b6	SWDRF	dedicated watchdog Reset flag	When a special watchdog is reset, the hardware is set. Cleared by writing to CLRF. 0: No special watchdog reset 1: Special watchdog resetting occurs	R/W											
b5	WDRF	watchdog reset Mark	Set by hardware when a watchdog reset occurs. Cleared by writing to CLRF. 0: A watchdog reset has not occurred 1: Watchdog reset occurred	R/W											

b4	PVD2RF	Programmable voltage detection 2 Reset flag	Set by hardware when a programmable voltage sense 2 reset occurs. Cleared by writing to CLRF. 0: Programmable voltage detection 2 reset does not occur 1: Programmable voltage detection 2 reset occurs	R/W
b3	PVD1RF	Programmable voltage detection 1 Reset flag	Set by hardware when a Programmable Voltage Detect 1 reset occurs. Cleared by writing to CLRF. 0: Programmable voltage detection 1 reset does not occur 1: Programmable voltage detection 1 reset occurs	R/W
b2	BORF	Undervoltage Reset flag	Set by hardware when a brown-out reset occurs. Cleared by writing to CLRF. 0: Brown-out reset has not occurred 1: Brown-out reset occurred	R/W
b1	PINRF	NRST pin Reset flag	When pin reset occurs, it is set by hardware. Cleared by writing to CLRF. 0: No NRST reset occurred 1: NRST reset occurs	R/W
b0	PORF	Power-on reset Mark	When a power-on reset occurs, the hardware is set. Cleared by writing to CLRF. 0: No power-on reset 1: Power-on reset	R/W

4 Clock Controller (CMU)

4.1 Introduction

The clock control unit provides a series of frequency clock functions, including: an external high-speed oscillator, an external low-speed oscillator, two PLL clocks, an internal high-speed oscillator, an internal medium-speed oscillator, an internal low-speed oscillator, A SWDT dedicated internal low-speed oscillator, clock prescaler, clock multiplexing and clock gating circuits.

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 all originate from the system clock. The system clock source has three timers:

- 1) External high-speed oscillator (XTAL)
- 2) External low-speed oscillator (XTAL32)
- 3) MPLL clock (MPLL)
- 4) Internal high-speed oscillator (HRC)
- 5) Internal Medium Speed Oscillator (MRC)
- 6) Internal low-speed oscillator (LRC)

The maximum operating clock frequency of the system clock can reach 200 MHz. SWDT has an independent clock source: SWDT dedicated internal low-speed oscillator (SWDTLRC). The real-time clock (RTC) uses an external low-speed oscillator or an internal low-speed oscillator as a clock source. The 48MHz clock of USB-FS can choose system clock, MPLL, UPLL as the clock source.

For each timer, you can turn it on and off separately when not in use to reduce power consumption.

4.2 System Block Diagram

4.2.1 System Block Diagram

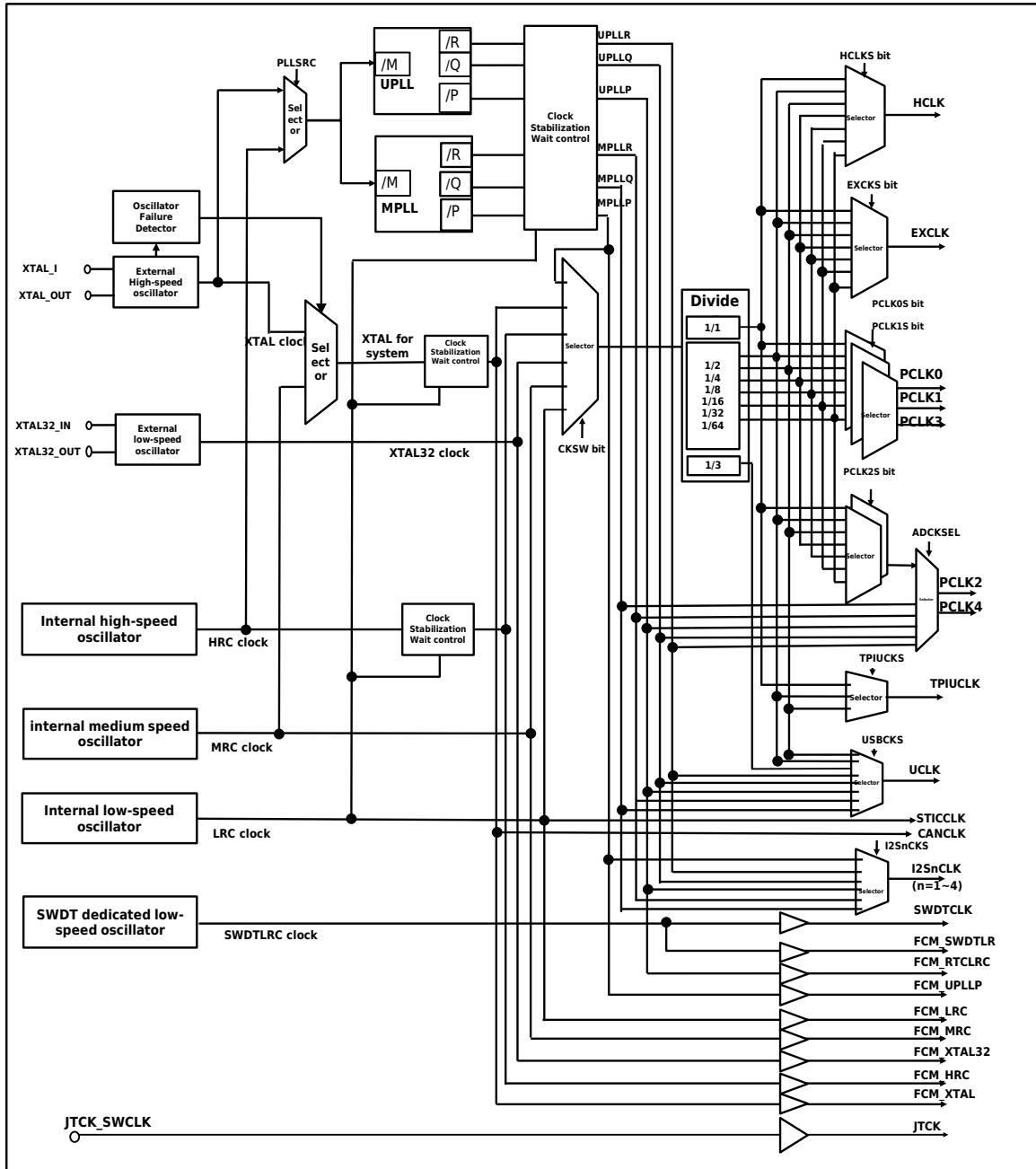


Figure 4-1 Clock System Block Diagram

4.2.2 Clock Frequency Measurement Block Diagram

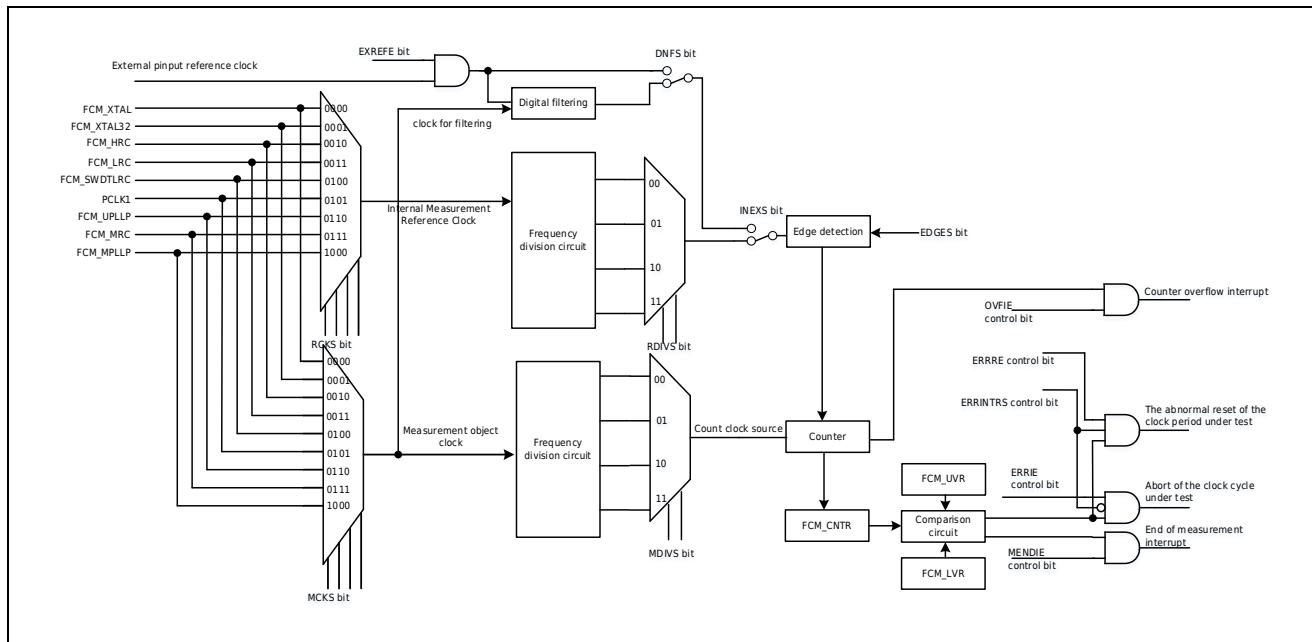


Figure 4-2 Clock Frequency Measurement Block Diagram

4.3 Timer Specification

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

Timer	Specifications
External high-speed oscillator (XTAL)	Frequency range of crystal oscillator: 4~25MHz External clock input: Up to 25 MHz Oscillator Fault Detection Function
External low-speed oscillator (XTAL32)	Crystal oscillator frequency: 32.768KHz
MPLL clock (MPLL)	Input clock: external high-speed oscillator or internal high-speed oscillator MPLL input clock frequency division: 1~24 arbitrary frequency division optional PFD input frequency = input clock/MPLL input clock frequency division, frequency range 1MHz~25MHz MPLL multiplication factor: 20~480 times VCO oscillation frequency: 240MHz~480MHz MPLLQ output divider ratio: 2~16 arbitrary frequency division MPLLP output divider ratio: 2~16 arbitrary frequency division MPLL R output divider ratio: 2~16 arbitrary frequency division MPLLP output frequency = (input clock/MPLL input clock frequency division)*MPLL multiplication factor/MPLLP output frequency division ratio MPLLQ output frequency = (input clock / MPLL input clock frequency division) * MPLL multiplication factor / MPLLQ output frequency division ratio MPLL R output frequency = (input clock/MPLL input clock frequency division)*MPLL frequency multiplication factor/MPLL R output frequency division ratio
UPLL Clock (UPLL)	Input clock: external high-speed oscillator or internal high-speed oscillator UPLL input clock frequency division: 1~24 arbitrary frequency division optional PFD input frequency = input clock/UPLL input clock frequency division, frequency range 1MHz~25MHz UPLL multiplication factor: 20~480 times VCO oscillation frequency: 240MHz~480MHz UPLLP output frequency division ratio: 2~16 arbitrary frequency division UPLLQ output frequency division ratio: 2~16 arbitrary frequency division UPLL R output frequency division ratio: 2~16 arbitrary frequency division UPLLP output frequency = (input clock / UPLL input clock frequency division) * UPLL frequency multiplication factor / UPLLP output frequency division ratio UPLLQ output frequency = (input clock / UPLL input clock frequency division) * UPLL frequency multiplication factor / UPLLQ output frequency division ratio UPLL R output frequency = (input clock / UPLL input clock frequency division) * UPLL frequency multiplication factor / UPLL R 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 Can be used as count clock of RTC, count clock of wake-up timer WKT, backup clock of XTAL32
SWDT dedicated internal low-speed oscillator (SWDTRC)	Frequency: 10 KHz

4.4 Working Clock Specification

Table 4-1 Specifications of Each Internal Clock

Clock	Scope of action	Specifications
HCLK	CPU, DMAn (n=1, 2), EFM (main flash memory), SRAM1, SRAM2, SRAM3, SRAMH, Ret-SRAM, MPU, GPIO, DCU, INTC, QSPI	Maximum frequency 200 MHz Frequency division of optional clock source: 1, 2, 4, 8, 16, 32, 64
PCLK0	Timer6 counter clock	Maximum frequency 200 MHz Frequency division of optional clock source: 1, 2, 4, 8, 16, 32, 64
PCLK1	USARTn (n=1~4), SPI(n=1~4), USBFS (control logic), TimerOn(n=1, 2), TimerAn(n=1~6), Timer4n(n=1~3) , Timer6 (control logic), EMB, CRC, HASH, AES, I2Sn (n=1~4) control logic	Maximum frequency 100 MHz Frequency division of optional clock source: 1, 2, 4, 8, 16, 32, 64
PCLK2	ADC conversion clock	Maximum frequency 60 MHz Frequency division of optional clock source: 1, 2, 4, 8, 16, 32, 64 Independently selectable clock sources: UPLLP, UPLLQ, UPLL, MPLLP, MPLLQ, MPLLR
PCLK3	RTC (control logic), I2C n(n=1, 2, 3), CMP, WDT, SWDT (control logic)	Maximum frequency 50 MHz Frequency division of optional clock source: 1, 2, 4, 8, 16, 32, 64
PCLK4	ADC (control logic), TRNG	Maximum frequency 100 MHz Frequency division of optional clock source: 1, 2, 4, 8, 16, 32, 64 Independently selectable clock sources: UPLLP, UPLLQ, UPLL, MPLLP, MPLLQ, MPLLR
EXCLK	SDION(n=1,2) , CAN	Maximum frequency 100 MHz Frequency division of optional clock source: 1, 2, 4, 8, 16, 32, 64
UCLK	Clock for USBFS communication	Frequency 48 MHz The clock source can be divided into 2, 3, and 4 of the system clock. Independently selectable clock sources: UPLLP, UPLLQ, UPLL, MPLLP, MPLLQ, MPLLR
CANCLK	CAN communication clock	The frequency range is 4-25 MHz
STICCLK	CPU's SysTick Time rcounter clock, timer is LRC	Configurable as clock source LRC or system clock
SWDCLK	SWDT Mcounter clock	Frequency 10 KHz
JTCK	JTAG clock	Maximum frequency 25 MHz
TPIUCLK	Clock for Cortex-M4 debug tracer	Maximum frequency 50 MHz Frequency division of optional clock source: 1, 2, 4
I2SnCLK (n=1~4)	I2Sn(n=1~4)	Maximum frequency 200 MHz Independently selectable clock sources: UPLLP, UPLLQ, UPLL, MPLLP, MPLLQ, MPLLR

Note:

The following rules must be observed between the clocks:

- HCLK frequency>=PCLK1 frequency, HCLK frequency>=PCLK3 frequency, HCLK frequency>=PCLK4 frequency
- HCLK frequency: EXCLK frequency = 2:1, 4:1, 8:1, 16:1, 32:1
- PCLK0 frequency>=PCLK1 frequency, PCLK0 frequency>=PCLK3 frequency
- HCLK frequency: PCLK0 frequency = N:1,1:N

- PCLK2 frequency: PCK4 frequency = 1:4, 1:2, 1:1, 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 more accurate timer for the system clock. The frequency range is 4-25 MHz.

XTAL opens and closes through 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 must be set at a stable time greater than or equal to that required by the crystal oscillator manufacturer.

The circuit constant of the crystal oscillator varies with the parasitic capacitance of the crystal oscillator and the installed circuit, so it must be discussed 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 adjusted appropriately depending on the oscillator selected. It is not possible to pass the signal line near the oscillating circuit, otherwise it may not be able to oscillate normally due to inductance.

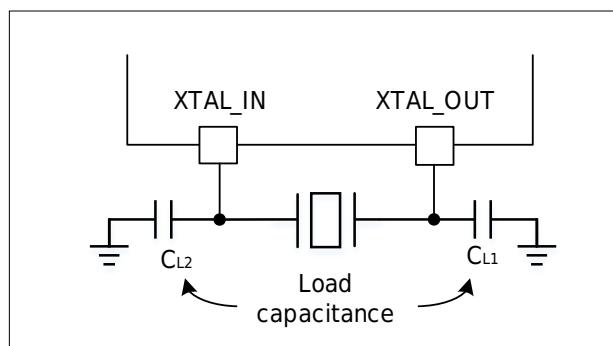


Figure 4-3 External High-Speed Oscillator Connection Example

4.5.1.2 Clock Input Mode

External timer must be provided in clock input mode. This mode is selected by XTALMS bit "1" of CMU_XTALCFGR and XTALSTP bit "0" of 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 settings.

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

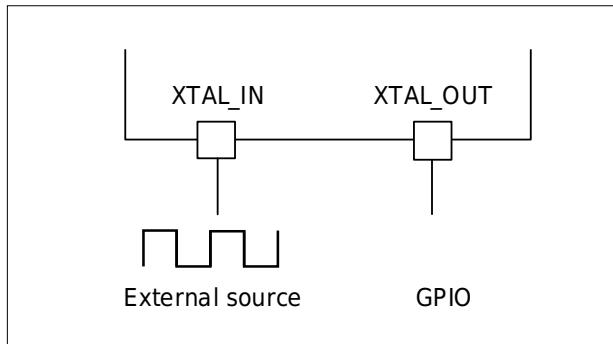


Figure 4-4 Example Connection Diagram Of External Clock Input

4.5.2 External High-Speed Oscillator Fault Detection

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

Open or close by register CMU_XTALSTDCR XTALSTDE bit.

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, you must enable the external high-speed oscillator to oscillate, and wait until the external high-speed oscillator is stable, that is, CMU_OSCSTBSR.XTALSTBF is 1, and turn it on through the XTALSTDE bit of the register CMU_XTALSTDCR.

When MPLL and UPLL select XTAL clock as the input source, they can only select the XTAL oscillation fault to generate reset function.

Since the oscillator fault detection is to detect the oscillator abnormal oscillation caused by external factors, the oscillator fault detection function is disabled when the oscillator external high-speed oscillator is to be stopped or transferred to the stop mode and power-off mode by software.

If the external high-speed oscillator fails, the action waveform is shown in the following figure. Operation process reference [XTAL fault detection action detected].

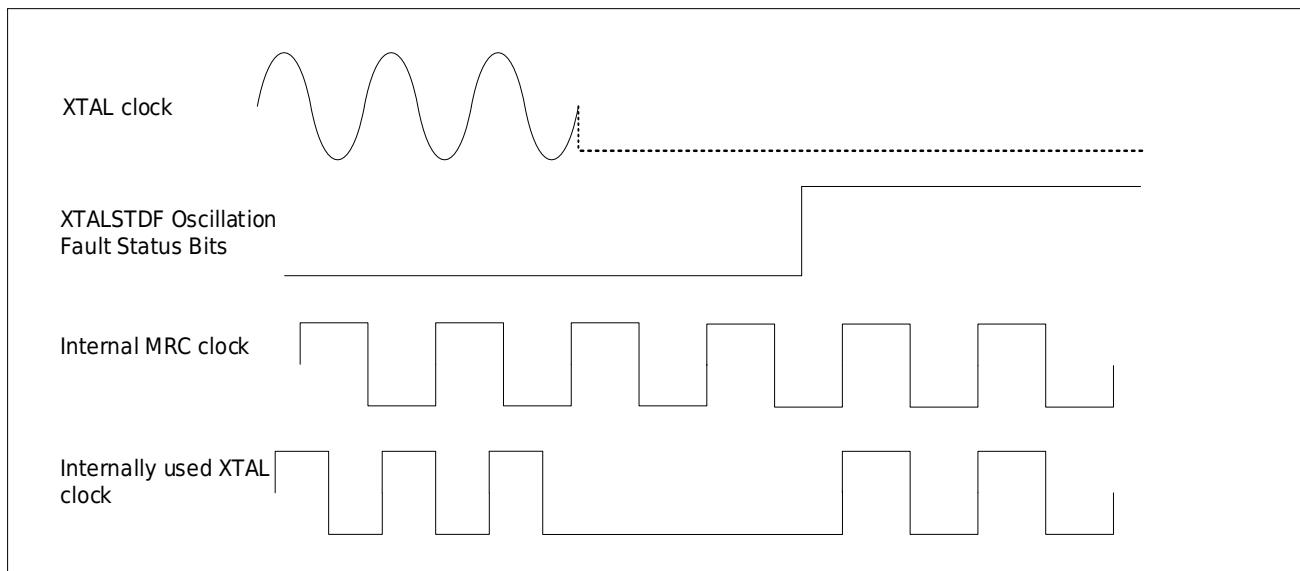


Figure 4-5 Example of External High-Speed Oscillator Fault Detection

4.5.2.1 XTAL Fault Detection Action Detected

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

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

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

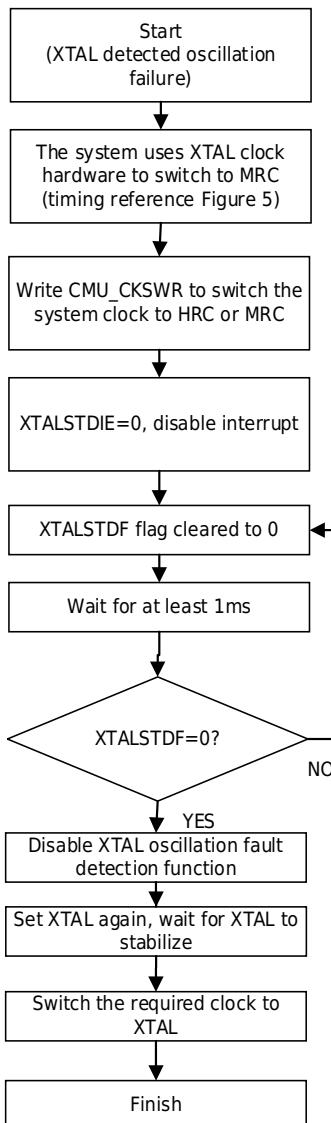


Figure 4-6 XTAL Is Selected As The System Clock, And XTAL Oscillation Fault Is Detected

4.5.2.2 Xtal Oscillation Fault Generated Termination Restoration Detected

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

When the XTAL oscillation fault is configured as reset, the XTAL oscillation fault is detected, and the chip resets. For the reset action, refer to Chapter [Reset Control (RMU)].

4.5.3 External Low-speed Oscillator

The 32.768KHz external low-speed oscillator can provide a more accurate 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 the XTAL32STP bit of CMU_XTAL32CR.

The circuit constant of the crystal oscillator varies with the parasitic capacitance of the crystal oscillator and the installed circuit, so it must be discussed 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 possible to pass the signal line near the oscillating circuit, otherwise it may not be able to oscillate normally due to inductance.

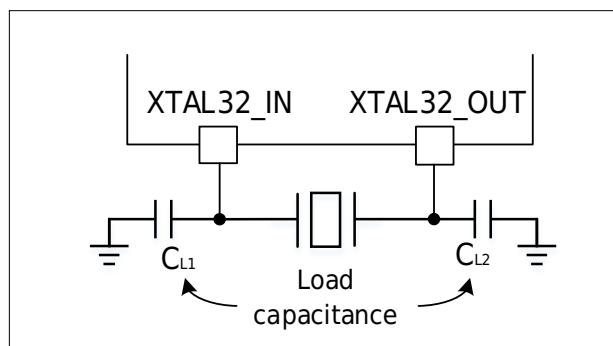


Figure 4-7 External Low-speed Oscillator Connection Example

The initialization process of XTAL32 power-on for the first time is as follows:

1. CMU_XTAL32CR.XTAL32STP bit write 1, stop XTAL32
2. Set the matching XTAL32 drive capability through CMU_XTAL32FGR
3. Set filter function by CMU_XTAL32FGR
4. CMU_XTAL32CR.XTAL32STP bit write 0, XTAL32 oscillation
5. The software waits for the XTAL32 to stabilize, and the stabilization time refers to the electrical characteristics chapter.

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

4.6 Internal RC clock

4.6.1 HRC Clock

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

HRC oscillator has the advantage of being less expensive (no external components needed). In addition, the start-up speed is higher than that of the XTAL crystal vibrator, but the accuracy is lower than that of the external crystal vibrator 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 accuracy. Refer to the **internal high-speed (HRC) oscillator chapter in the electrical characteristics of the Data Manual**.

If the application is subject to temperature variations, this may also affect the speed of the RC oscillator. Users can fine-tune the HRC frequency through the CMU_HRCTR register.

The HRCSTBF flag in CMU_OSCSTBSR indicates the stability of HRC. The HRC is not available until the hardware places this position at startup.

HRC can control HRCSTP bits in register to open or close through CMU_HRCCR.

4.6.2 MRC Clock

The MRC clock signal is generated by an internal 8MHz medium-speed oscillator and can be directly used as the system clock.

The advantage of the MRC oscillator is that it starts up quickly, and it can be used without waiting for stabilization after startup.

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 accuracy. Refer to the **internal medium-speed (MRC) oscillator chapter in the electrical characteristics of the Data Manual**.

If the application is subject to temperature variations, this may also affect the speed of the RC oscillator. Users can fine-tune the MRC frequency through the CMU_MRCTR register.

MRC can control MRCSTP bits in register to open or close through CMU_MRCCR.

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

4.6.3 LRC Clock

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

LRC oscillator can be started quickly and can be used without any stability.

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 accuracy. Refer to the **internal low-speed (LRC) oscillator chapter in the electrical characteristics of the Data Manual**.

If the application is affected by a voltage or temperature change, this may also affect the speed of RC oscillator. Users can fine-tune the LRC frequency through the CMU_LRCTRM register.

The LRC can control the LRCSTP bits in register to open or close through CMU_LRCCR.

4.6.4 SWDTRC Clock

The SWDTRC clock signal is generated by the internal 10KHz low-speed oscillator, and the SWDT exclusive clock. If SWDT has been started by ICG setting, the internal low-speed oscillator dedicated to 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 accuracy. Refer to the **SWDT dedicated internal low-speed (SWDTLRC) oscillator chapter in the electrical characteristics of the Data Manual**.

4.7 PLL Clock

The HC32F46xx devices have two PLLs:

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

UPLL uses the same input clock source as MPLL, and can choose HRC or XTAL oscillator as the clock source, which is configured by the CMU_PLLCFGR.PLLSRC bit. After the HRC or XTAL oscillator is stable, configure the PLL.

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

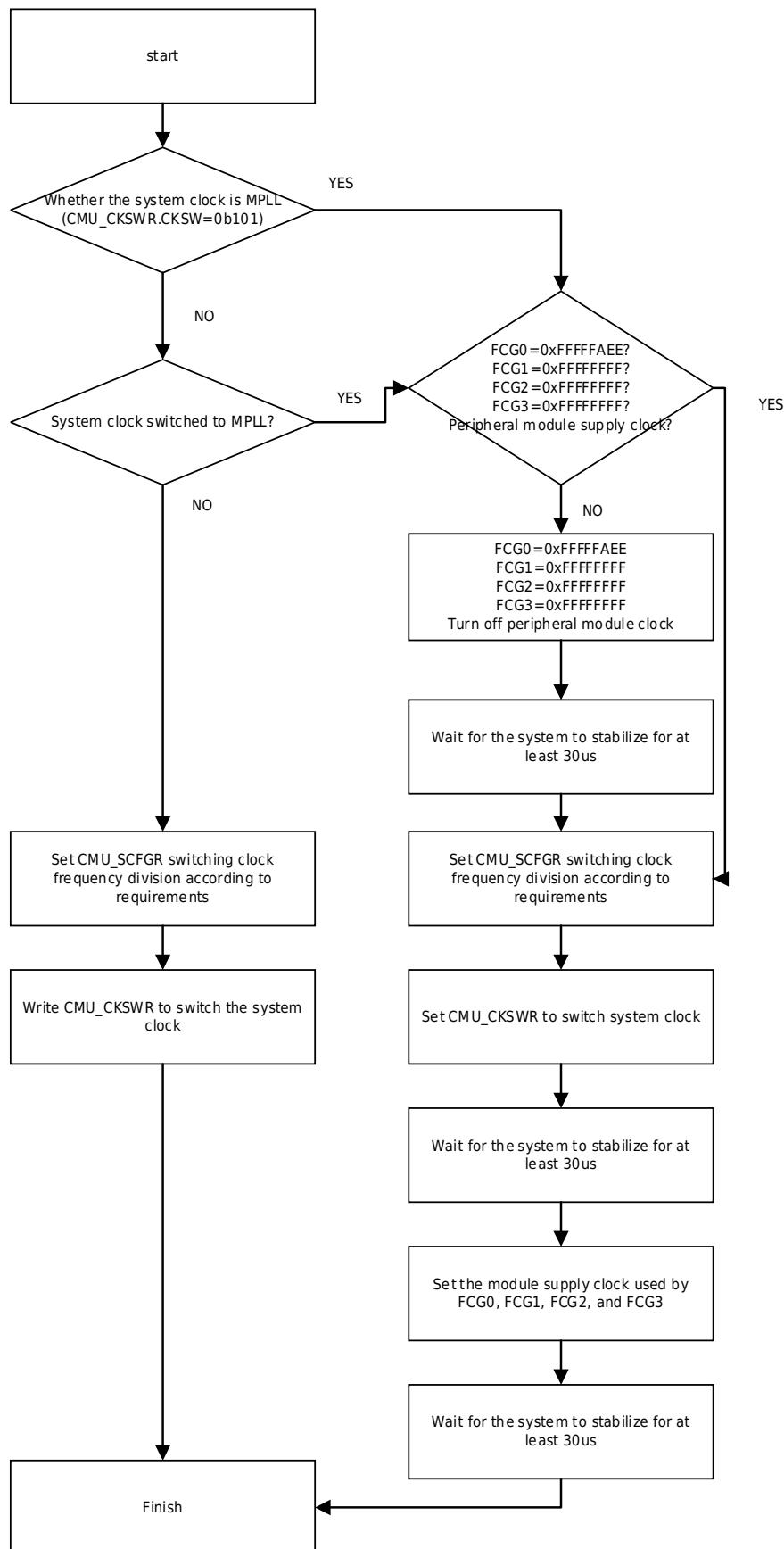
Both PLLs are disabled by hardware when entering power-down and stop modes.

4.8 Clock Switching Step

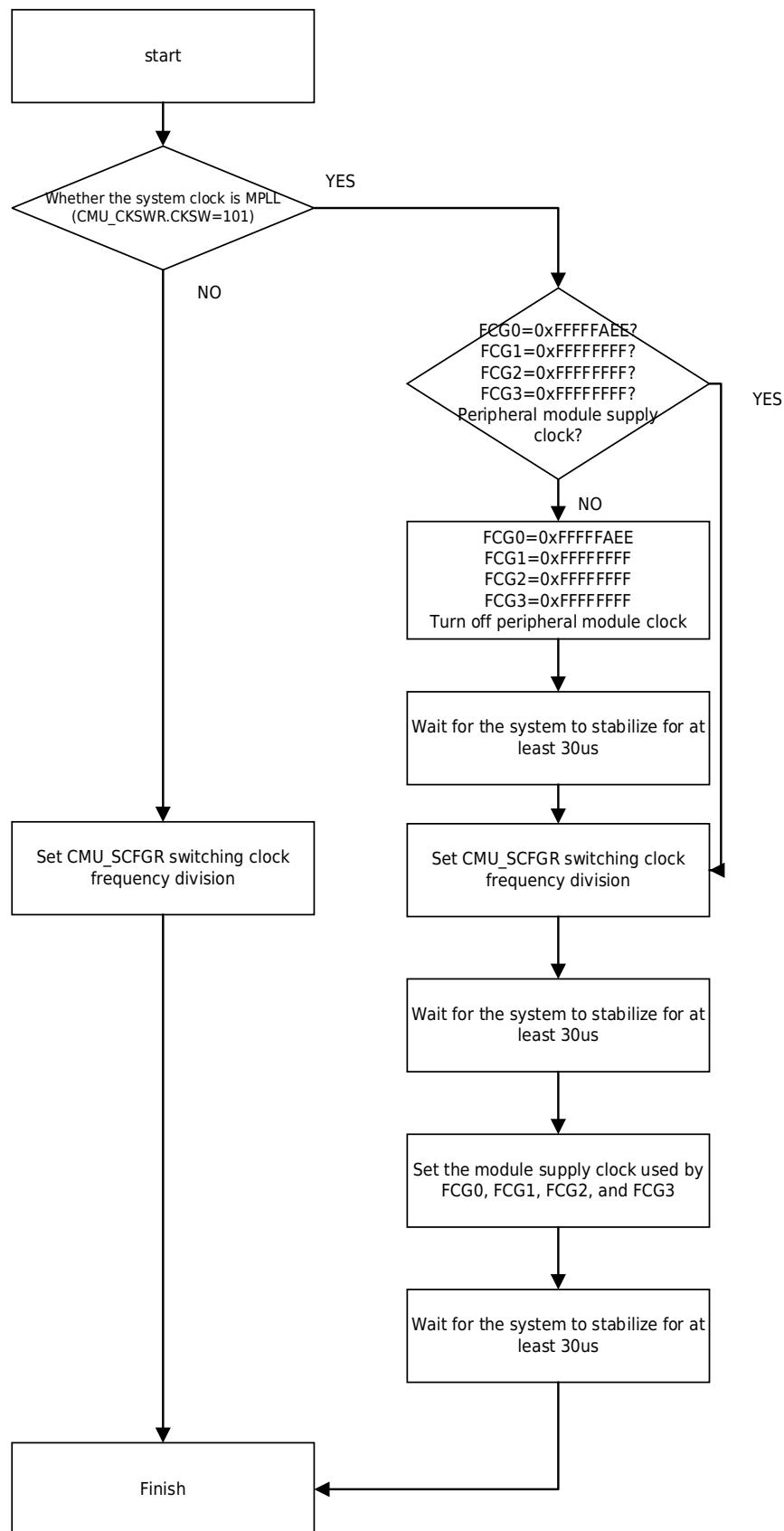
After the system is reset, the default system clock is MRC. Switch the clock source by setting the register CMU_CKSW, refer to the switching stepsTimer Switchover. You can switch from one timer to another only if the target timer is stable.

When switching the clock, it is necessary to correctly configure the waiting period of FLASH/SRAM to prevent the system clock frequency from being higher than the maximum operating frequency of FLASH/SRAM. Refer to [Relationship between CPU Clock and FLASH Read Time], [Built-in SRAM (SRAM)]configuration.

4.8.1 Timer Switchover



4.8.2 Clock Divider Switching



4.9 Clock Output Function

There are two clock outputs:

- MCO_1

Users can output different timers to the MCO_1 pin through a configurable pre-distributor (from 1 to 128):

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

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

- MCO_2

Users can output different timers to the MCO_2 pin through a configurable pre-distributor (from 1 to 128):

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

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

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

4.10 Clock Frequency Measurement

4.10.1 Clock Frequency Measurement

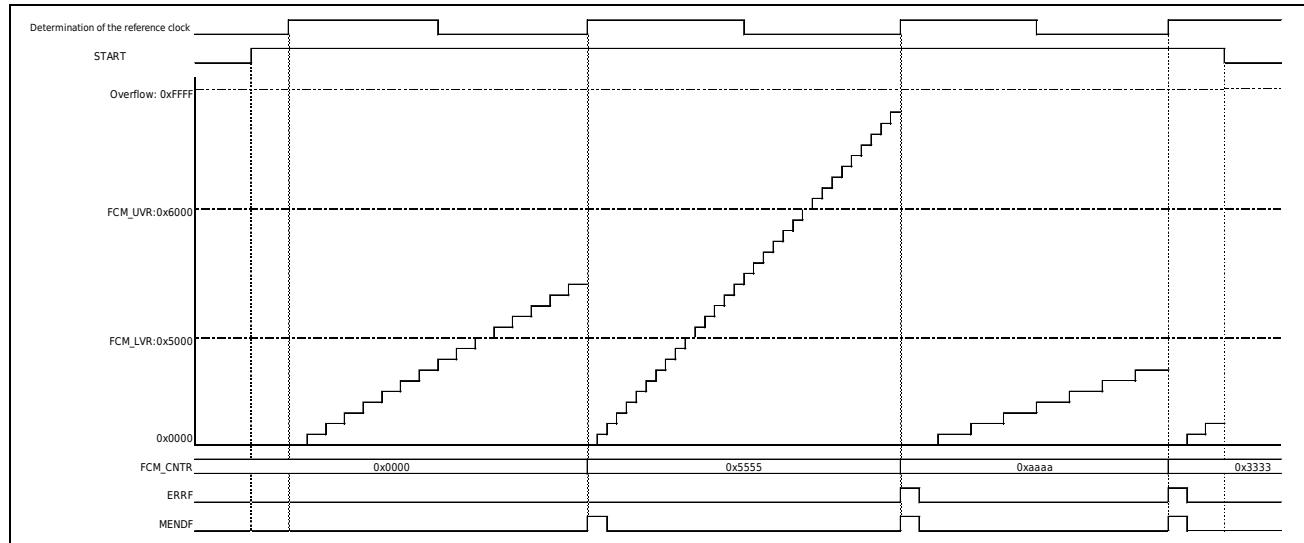


Figure 4-8 Clock Frequency Measurement Timing Diagram

1. Use FCM_MCCR/FCM_RCCR to select the reference clock to be measured, clock frequency division, and select the effective edge of the reference clock.
2. After writing 1 to the START bit of FCM_STR, the valid edge selected by the EDGES bit is detected, and the counter starts counting up.
3. When the valid edge selected by the next EDGES bit of the reference clock is detected, the value of the counter is saved to the FCM_CNTR register and compared with the set value of FCM_LVR/ FCM_UVR. When $FCM_LVR \leq FCM_CNTR \leq FCM_UVR$, the measured clock frequency measurement is normal. When $FCM_LVR > FCM_CNTR$ or $FCM_CNTR > FCM_UVR$, the measured clock frequency is abnormal, and an interrupt or reset can occur according to the ERRINTRS/ ERRRE/ ERRIE settings.
4. After the START bit of FCM_STR is written to 0, the counter counting is stopped and cleared.

4.10.2 Digital Filtering Function

External pin input reference clock FCMREF has digital filtering function. The digital filter function performs three samplings according to the sampling clock selected by the DNFS bit, and when the level of the three samplings is the same, the level is sent to the inside.

The digital filtering function can set the digital filtering function to be valid or invalid and the sampling clock.

4.10.3 Interrupt/Reset Function

There are three types of interrupt requests for the clock frequency measurement circuit. They are:

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

The clock frequency measurement circuit has a reset request:

- 1) Frequency abnormal reset

4.11 Register Description

Base address 1: 0x4004_8400

Register name	Symbol	Offset address	Bit width	Reset value
FCM Lower Limit Compare Value Register	FCM_LVR	0x00	32	0x0000_0000
FCM Upper Limit Compare Value Register	FCM_UVR	0x04	32	0x0000_0000
FCM Counter Value Register	FCM_CNTR	0x08	32	0x0000_0000
FCM start stop register	FCM_STR	0x0C	32	0x0000_0000
FCM Measurement Object Control Register	FCM_MCCR	0x10	32	0x0000_0000
FCM Measurement Reference Control Register	FCM_RCCR	0x14	32	0x0000_0000
FCM interrupt reset control register	FCM_RIER	0x18	32	0x0000_0000
FCM flags register	FCM_SR	0x1C	32	0x0000_0000
FCM flag bit clear register	FCM_CLR	0x20	32	0x0000_0000

Base address 2: 0x40054000

Register name	Symbol	Offset address	Bit width	Reset value
CMU_XTAL configuration register	CMU_XTALCFG	0x410	8	0x80
CMU_XTAL Stability 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_XTALSTDSCR	0x041	8	0x00
CMU_HRC Calibration Register	CMU_HRCTRM	0x062	8	0x00
CMU_HRC control register	CMU_HRCCR	0x036	8	由ICG1.HRCSTP值决定
CMU_MRC Calibration Register	CMU_MRCTR	0x061	8	0x00
CMU_MRC Control Register	CMU_MRCCR	0x038	8	0x80
CMU_MPLL configuration register	CMU_PLLCFG	0x100	32	0x1110_1300
CMU_MPLL Control Register	CMU_PLLCR	0x02A	8	0x01
CMU_UPLL configuration register	CMU_UPLLCFG	0x104	32	0x1110_1300
CMU_UPLL Control Register	CMU_UPLLCR	0x02E	8	0x01
CMU_timer stability register	CMU_OSCSTBSR	0x03C	8	0x00
CMU_System timer switchover register	CMU_CKSWR	0x026	8	0x01
CMU_Clock Divider Configuration Register	CMU_SCFGR	0x020	32	0x0000_0000
CMU_USBFS Clock Configuration Register	CMU_UFSCKCFG	0x024	8	0x40
CMU_AD/TRNG clock configuration register	CMU_PERICKSEL	0x010	16	0x0000
CMU_I2S clock configuration register	CMU_I2SCKSEL	0x012	16	0xBBB
CMU_DEBUG CLOCK CONFIG REGISTER	CMU_TPIUCKCFG	0x03F	8	0x00
CMU_MCO1 clock output configuration register	CMU_MCO1CFG	0x03D	8	0x00
CMU_MCO2 Clock Output Configuration Register	CMU_MCO2CFG	0x03E	8	0x00
CMU_XTAL32 Control Register	CMU_XTAL32CR	0x420	8	0x00
CMU_XTAL32 configuration register	CMU_XTAL32CFG	0x421	8	0x00
CMU_XTAL32 filter register	CMU_XTAL32NFR	0x425	8	0x00
CMU_LRC control register	CMU_LRCCR	0x427	8	0x00
CMU_LRC Calibration Register	CMU_LRCTR	0x429	8	0x00

4.11.1 CMU XTAL Configuration Register (CMU_XTALCFGR)

Reset value: 0x80

b7	b6	b5	b4	b3	b2	b1	b0
SUPDRV	XTALMS	XTALDRV[1:0]		-	-	-	-
<hr/>							
Bit	Marking	Place name			Function		Read and write
b7	SUPDRV	XTAL Hyper-Speed Drive Allowance			0: Prohibiting hyperspeed drive 1: Allow super-high-speed drive When ultra-high-speed drive is allowed, XTAL stabilizes, disregards this bit setting, prohibits ultra-high-speed drive, reduces power consumption.		R/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 (recommended $20 < f_{XTAL_IN} \leq 25MHz$ crystal oscillator) 01: Medium drive capability (recommended $16 < f_{XTAL_IN} \leq 20MHz$ crystal oscillator) 10: Small driving capability (recommended $8f_{XTAL_IN} \leq 16MHz$ crystal oscillator) 11: Ultra-small driving capability (recommended $4 \leq f_{XTAL_IN} \leq 8MHz$ crystal oscillator)		R/W
b3~b0	Reserved	-			Read at "0", write at "0"		R/W

4.11.2 CMU XTAL Stability 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 at "0", write at "0"		R/W
b3~b0	XTALSTB[3:0]	XTAL Stability Time Selection			0001: Stable counter 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 A counting period of a stable counter = LRC period / 8 This register is configured with CMU_XTALCR.XTALSTP bit 1 and CMU_OSCSTBSR.XTALSTBF bit 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 at "0", write at "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 system clock or MPLL/UPLL clock source, XTALSTP is prohibited to write "1" to stop XTAL oscillator.
- The XTAL oscillator is set by software to oscillate. After confirming that the XTAL oscillator is stable through the XTALSTBF bit, it can enter stop mode, power-down mode or stop the XTAL oscillator by software setting.
- The XTAL oscillator is set to stop by software, and the XTAL oscillator can be entered into stop mode, power-down mode or restarted after confirming that the XTAL oscillator is stopped through the XTALSTBF bit.

4.11.4 CMU XTAL Oscillatory Fault Control Register (CMU _ XTALSTDRCR)

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 Allowed	0: Disable XTAL Oscillation Fault Detection 1: Allow XTAL Oscillation Fault Detection Note: Oscillator fault detection is to detect the oscillator abnormal oscillation caused by external factors. Please disable the oscillator oscillation fault detection function before entering stop mode or power-off mode.	R/W
b6~b3	Reserved	-	Read at "0", write at "0"	R/W
b2	XTALSTDRI	XTAL Oscillatory Fault Reset Interrupt Selection	0: XTAL OSCILLATION FAULT GENERATES TERRATE 1: XTAL Oscillation Fault Reset Note: When MPLL and UPLL select XTAL clock as the input source, they can only select the XTAL oscillation fault to generate reset function.	R/W
b1	XTALSTDRE	XTAL Oscillatory Fault Reset Allowance	0: Disable XTAL Oscillatory Fault Reset 1: Allow XTAL Oscillation Fault Reset	R/W
b0	XTALSTDIE	XTAL Oscillation Fault Interrupt Allowance	0: Disable XTAL OSCILLATION FAULT TERMINATION 1: Allow XTAL OSCILLATION FAULT TERMINATION Set the PWM output of Timer6/Timer4 to Hiz output through EMB, and the XTALSTDIE bit needs to be set to 1.	R/W

Note:

- When XTAL is selected as system clock or MPPLL/UPLL clock source, XTALSTP is prohibited to write "1" to stop XTAL oscillator.

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 at "0", write at "0"	R/W
b0	XTALSTDF	XTAL Oscillatory Fault State Bit	0: No XTAL oscillation fault detected 1: XTAL Oscillation Fault Detected Setting conditions: XTAL OSCILLATION FAULT UNDER XTALSTDE = 1 Reset 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_XTAL32CFGR)

Reset value: 0x00

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

Bit	Marking	Place name	Function	Read and write
b7~b3	Reserved	-	Read at "0", write at "0"	R/W
b2~b0	XTAL32DRV[2:0]	XTAL32 drive capability selection	000: Medium drive capability 001: Large drive capacity Other: prohibitions Note: For the method of use, refer to the chapter of 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: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	XTAL32NF[1:0]
Bit	Marking	Place name	Function				Read and write
b7~b2	Reserved	-	Read at "0", write at "0"				R/W
b1~b0	XTAL32NF[1:0]	XTAL32 Oscillator Filter Selection	00: RUN mode/stop mode/power down mode, 3us filter of XTAL32 is valid 01: The 3us filter of XTAL32 is valid in RUN mode, and the 3us filter of XTAL32 in stop mode or power-down mode is invalid 10: Prohibiting 11: RUN mode/stop mode/power down mode, the 3us filter 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 at "0", write at "0"				R/W
b0	XTAL32STP	XTAL32 oscillator start stop bit	0: XTAL32 oscillator oscillates 1: XTAL32 oscillator stopped				R/W

Note:

- When XTAL32 is selected as the system clock source, XTAL32STP is prohibited from writing "1" to stop the XTAL32 oscillator.
- The software sets the XTAL32 action to start, and waits for 5 XTAL32 cycles before stopping the XTAL32 again.
- The software sets the XTAL32 to stop, and waits for 5 XTAL32 cycles before starting the 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 bit	Frequency calibration needs to be within the guaranteed range of HRC frequency. 10000000: -128 10000001: -127 11111111: -1 00000000: Center Code 00000001: +1 01111110: +126 01111111: +127				R/W

4.11.10 CMU HRC Control Register (CMU_HRCCR)

Reset value: determined by ICG1.HRCSTP value

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	HRCSTP
Bit	Marking	Place name	Function				Read and write
b31~b1	Reserved	-	Read at "0", write at "0"				R/W
b0	HRCSTP	HRCoscillator open stop bit	0: HRCoscillator Oscillation 1: HRCoscillator stops According to ICG1.HRCSTOP configuration, HRC starts to stop after reset.				R/W

Note:

- When HRC is selected as the system clock source or MPLL/UPLL clock source, it is forbidden to write "1" in CMU_HRCCR.HRCSTP to stop the HRC clock.
- The HRC oscillator is set by software, and the HRC can be entered into stop mode, power-down mode or stop HRC only after confirming that the HRC is stable through the HRCSTBF bit.
- The software sets the HRC to stop, and the MPLL can be stopped after confirming the MPLL stop through the HRCSTBF bit before entering the stop mode, power-down mode or restarting the HRC.

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
			10000000: -128				
			10000001: -127				
						
b7~b0	MRCTRIM[7:0]	MRC Frequency Calibration Bits	11111111: -1				R/W
			00000000: Center Code				
			00000001: +1				
						
			01111110: +126				
			01111111: +127				

Note:

- The frequency calibration needs to be within the MRC frequency guaranteed 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	-	-	Read at "1", write at "1"				R/W
b6~b1	Reserved	-	Read at "0", write at "0"				R/W
b0	MRCSTP	MRC oscillator open stop bit	0: MRC oscillator starts 1: MRC oscillator stops Note: 1) When the XTAL oscillation fault function is valid, this bit is cleared to 0 at the same time, and MRC oscillates. 2) The stop mode wake-up action when the PWC_STPMCR.CKSMRC bit is 1 is set when the MRC oscillator is in the oscillating state.				R/W

Note:

- When MRC is selected as system timer, MRCSTP writes "1" to stop MRC clock.
- The software sets the MRC oscillation and waits for 5 MRC cycles before entering stop mode, power-down mode or stopping MRC.
- The software sets the MRC to stop, and waits for 5 MRC cycles before entering the stop mode, power-down mode or restarting the MRC.
- MRC is used as RTC calibration clock. Possibility of MRC oscillation when RTC is not initialized. When the RTC calibration function is enabled, the MRC oscillates regardless of the MRCSTP bit setting.

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
			10000000: -128				
			10000001: -127				
						
b7~b0	LRCTRIM[7:0]	LRC Frequency Calibration Bits	11111111: -1				R/W
			00000000: Center Code				
			00000001: +1				
						
			01111110: +126				
			01111111: +127				

Note:

- The frequency calibration needs to be within the LRC frequency guaranteed 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 at "0", write at "0"				R/W
b0	LCRSTP	LCRoscillator open stop bit	0: LCRoscillator Oscillation 1: LCRoscillator stops				R/W

Note:

- When LRC is selected as system timer, LCRSTP writes "1" to stop LRC clock.
- The software sets the start of the LRC action, and waits for 5 LRC cycles before entering the stop mode, power-down mode or stopping the LRC.
- The software sets the LRC to stop, and 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, MPLL, and UPLL clocks to be stable, the LCRSTP bit setting is ignored, and the LRC is forced to oscillate.
- When RTC selects LRC as clock source, LRC ignores this register bit, and LRC oscillates. When the RTC is not initialized, the LRC may oscillate.

4.11.15 CMU MPLL Configuration Register (CMU_PLLCFGR)

Reset value: 0x1110_1300

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16										
MPLL P[3:0]		MPLL Q[3:0]		MPLL R[3:0]		-		-		-		MPLLN[8]													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0										
MPLL N[7:0]		PLL SRC		-		-		MPLL M[4:0]																	
<hr/>																									
Bit	Marking	Place name	Function										Read and write												
b31-b28	MPLL P[3:0]	MPLL frequency division factor for system clock	Frequency used for MPLLP clock, write MPLLP in MPLL stop condition. MPLL output clock frequency = VCO frequency of MPLL/MPLL 0000: Setting prohibited 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	MPLL Q[3:0]	MPLL frequency division factor for system clock	Frequency used for MPLLQ clock, write to MPLLQ during MPLL stop condition. MPLL output clock frequency = MPLL VCO frequency / MPLLQ 0000: Setting prohibited 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	MPLL R[3:0]	MPLL frequency division factor for system clock	Frequency used for MPLLR clock, write to MPLLR during MPLLR stop condition. MPLLR output clock frequency = MPLLR VCO frequency / MPLLR 0000: Setting prohibited 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 at "0", write at "0"										R/W												
b16-b8	MPLL N[8:0]	MPLL multiplication factor	It is used to control the multiplier coefficient of VCO of MPLL, write MPLLN under MPLL stop condition. Make sure the VCO frequency of the MPLL is between 240MHz and 480MHz. VCO frequency of MPLL = PFD input frequency of MPLL * MPLLN 000010011: 20 000010100: 21 000010101: 22 000010110: 23 111011101: 478 111011110: 479 111011111: 480										R/W												
b7	PLLSRC	MPLL/UPLL input clock source selection	0: Select the external high-speed oscillator as the input clock of MPLL/UPLL 1: Select the internal high-speed oscillator as the input clock of MPLL/UPLL										R/W												
b6-b5	-	-	Read at "0", write at "0"										R/W												
b4-b0	MPLL M[4:0]	MPLL input clock frequency division factor	Used to divide the MPLL input clock before the MPLL's VCO. Write to MPLLM during MPLL STOP condition. Make sure the PFD input frequency to the MPLL is between 1MHz and 25MHz. 00000: 1 frequency division 00001: 2 frequency division 00010: 3 frequency division 10111: 24 frequency division										R/W												

Other prohibited

4.11.16 CMU MPLL Control Register (CMU_PLLCR)

Reset value: 0x01

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

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

Note:

- When MPLL is selected as the system clock source, it is forbidden to write "1" to MPLLOFF to stop the MPLL clock.
- The software sets the MPLL action to start, and after the MPLLSTBF bit confirms that the MPLL is stable, it can enter the stop mode, power-down mode or software setting to stop the MPLL.
- The software sets the MPLL to stop, and the MPLL can be entered into the stop mode, power-down mode or start the MPLL again after confirming that the MPLL is stopped through the MPLLSTBF bit.
- When the MPLL selects the XTAL oscillator as the clock source, the MPLL operation can be set to start after confirming that the XTAL oscillator is stable through the XTALSTBF bit. When MPLL selects HRC as the clock source, the MPLL operation can be set to start after confirming that HRC is stable through the HRCSTBF bit.

4.11.17 CMU UPLL Configuration Register (CMU_UPLLCFGR)

Reset value: 0x1110_1300

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16										
UPLL[3:0]		UPLLQ[3:0]		UPLL[3:0]		-		-		-		UPLLN[8]													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0										
UPLLN[7:0]		-		-		-		UPLLM[4:0]																	
Bit	Marking	Place name	Function										Read and write												
b31-b28	UPLL[3:0]	UPLL frequency division factor for system clock	Used to control the frequency of UPLL clock, write UPLL in UPLL stop condition. UPLL output clock frequency = VCO frequency of UPLL / UPLL 0000: Setting prohibited 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	UPLLQ[3:0]	UPLL frequency division factor for system clock	Used to control the frequency of UPLLQ clock, write UPLLQ in UPLL stop condition. UPLL output clock frequency = VCO frequency of UPLL / UPLLQ 0000: Setting prohibited 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	UPLL[3:0]	UPLL frequency division factor for system clock	Used to control the frequency of UPLL[3:0] clock, write UPLL[3:0] in UPLL stop condition. UPLL output clock frequency = VCO frequency of UPLL / UPLL[3:0] 0000: Setting prohibited 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 at "0", write at "0"										R/W												
b16-b8	UPLLN[8:0]	UPLL multiplication factor	It is used to control the VCO multiplier coefficient of UPLL, write UPLLN in UPLL stop condition. Make sure the VCO frequency of the UPLL is between 240MHz and 480MHz. VCO frequency of UPLL = PFD input frequency of UPLL * UPLLN 000010011: 20 000010100: 21 000010101: 22 000010110: 23 111011101: 478 111011110: 479 111011111: 480 Other prohibitions										R/W												
b7-b5	-	-	Read at "0", write at "0"										R/W												
b4-b0	UPLLM[4:0]	UPLL input clock frequency division factor	Used to divide the UPLL input clock before the UPLL's VCO. Write UPLLM in UPLL STOP condition. Make sure the PFD input frequency to the UPLL is between 1MHz and 25MHz. 00000: Setting prohibited 00001: 2 frequency division 00010: 3 frequency division 10111: 24 frequency division										R/W												

4.11.18 CMU UPLL Control Register (CMU_UPLLCSR)

Reset value: 0x01

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	UPLLOFF
Bit	Marking	Place name	Function				Read and write
b7~b1	Reserved	-	Read at "0", write at "0"				R/W
b0	UPLLOFF	UPLL enable	Used to start and stop UPLL. 0: UPLL action starts 1: UPLL stop				R/W

Note:

- When UPLL is selected as the clock source of I2S/TRNG/ADC/USBFS, it is forbidden to write "1" to stop UPLL clock from UPLLOFF.
- The software sets the UPLL action to start, and the UPLL can only enter the stop mode, power-down mode or software setting to stop the UPLL after confirming that the UPLL is stable through the UPLLSTBF bit.
- The software sets the UPLL to stop, and the UPLL can only enter the stop mode, power-down mode or start the UPLL again after confirming that the UPLL is stopped through the UPLLSTBF bit.
- When UPLL selects the XTAL oscillator as the clock source, the UPLL operation can be set to start after confirming that the XTAL oscillator is stable through the XTALSTBF bit. When UPLL selects HRC as the clock source, the UPLL action can be set to start after confirming that HRC is stable through the HRCSTBF bit.

4.11.19 CMU Timer Stabilizer (CMU _ OSCSTBSR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	UPLLSTBF	MPLLSTBF	-	XTALSTBF	-	-	HRCSTBF
Bit	Marking	Place name	Function				Read and write
b7	Reserved	-	Read at "0", write at "0"				R
b6	UPLLSTBF	UPLL stable sign bit	0: UPLL stopped or unstabilized 1: UPLL stability				R
b5	MPLLSTBF	MPLL stable sign bit	0: MPLL stopped or unstabilized 1: MPLL stability				R
b4	Reserved	-	Read at "0", write at "0"				R
b3	XTALSTBF	XTAL stability sign bit	0: XTAL stopped or unstabilized 1: XTAL Stabilization				R
b2~b1	Reserved	-	Read at "0", write at "0"				R
b0	HRCSTBF	HRC stable sign bit	0: HRC stopped or unstabilized 1: HRC stability				R

4.11.20 Timer Switch Register of CMU 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 at "0", write at "0"				R/W
b2-b0	CKSW[2:0]	System timer switchover	000: Select HRC clock as system clock 001: Select the MRC clock as the system clock 010: Select the LRC clock as the system clock 011: Select the XTAL clock as the system clock 100: Select the XTAL32 clock as the system clock 101: Select MPLL as the system clock 110: Setting prohibited 111: Setting prohibited				R/W
			Note: 1. Switch the target timer to ensure the clock is stable. 2. Refer to the [Timer Switchover] chapter for the process 3. When the PWC_STPMCR.CKSMRCbit is 1, after the stop mode wakes up, this register initializes and the system timer selects the MRC clock.				

4.11.21 CMU Clock Divider Configuration Register (CMU_SCFGREG)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	HCLKS[2:0]			-	EXCKS[2:0]			-	PCLK4S[2:0]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	PCLK3S[2:0]			-	PCLK2S[2:0]			-	PCLK1S[2:0]			-	PCLK0S[2:0]		
Bit	Marking	Place name	Function	Read and write											
b31~b27	Reserved	-	Read at "0", write at "0"	R/W											
b26~24	HCLKS[2:0]	HCLK clock frequency division selection bit	000: 1 frequency division of system clock 001: 2-Division Frequency of System Clock 010: 4-division frequency of the system clock 011: 8 frequency division of system clock 100: 16-division frequency of system clock 101: 32 frequency division of system clock 110: 64-division frequency of system clock 111: Setting prohibited Note: When the PWC_STPMCR.CKSMRC bit is 1, this register is initialized after wake-up from stop mode, and HCLK is divided by 1 of the system clock.	R/W											
b23	Reserved	-	Read at "0", write at "0"	R/W											
b22~20	EXCKS[2:0]	ExMC clock frequency division selection bit	000: 1 frequency division of system clock 001: 2-Division Frequency of System Clock 010: 4-division frequency of the system clock 011: 8 frequency division of system clock 100: 16-division frequency of system clock 101: 32 frequency division of system clock 110: 64-division frequency of system clock 111: Setting prohibited Note: When the PWC_STPMCR.CKSMRC bit is 1, this register is initialized after wake-up from stop mode, and EXCLK is divided by 1 of the system clock.	R/W											
b19	Reserved	-	Read at "0", write at "0"	R/W											
b18~16	PCLK4S[2:0]	PCLK4 clock frequency division selection bit	000: 1 frequency division of system clock 001: 2-Division Frequency of System Clock 010: 4-division frequency of the system clock 011: 8 frequency division of system clock 100: 16-division frequency of system clock 101: 32 frequency division of system clock 110: 64-division frequency of system clock 111: Setting prohibited Note: When the PWC_STPMCR.CKSMRC bit is 1, this register is initialized after wake-up from stop mode, and PCLK4 is divided by 1 of the system clock.	R/W											
b15	Reserved	-	Read at "0", write at "0"	R/W											
b14~12	PCLK3S[2:0]	PCLK3 clock frequency division selection bit	000: 1 frequency division of system clock 001: 2-Division Frequency of System Clock 010: 4-division frequency of the system clock 011: 8 frequency division of system clock 100: 16-division frequency of system clock 101: 32 frequency division of system clock 110: 64-division frequency of system clock 111: Setting prohibited Note: When the PWC_STPMCR.CKSMRC bit is 1, this register is initialized after wake-up from stop mode, and PCLK3 is divided by 1 of the system clock.	R/W											
b11	Reserved	-	Read at "0", write at "0"	R/W											
b10~8	PCLK2S[2:0]	PCLK2 clock frequency division selection bit	000: 1 frequency division of system clock 001: 2-Division Frequency of System Clock 010: 4-division frequency of the system clock 011: 8 frequency division of system clock 100: 16-division frequency of system clock 101: 32 frequency division of system clock 110: 64-division frequency of system clock 111: Setting prohibited	R/W											

<p>Note: When the PWC_STPMCR.CKSMRC bit is 1, this register is initialized after wake-up from stop mode, and PCLK2 is divided by 1 of the system clock.</p>				
b7	Reserved	-	Read at "0", write at "0"	R/W
b6~4	PCLK1S[2:0]	PCLK1 clock frequency division selection bit	000: 1 frequency division of system clock 001: 2-Division Frequency of System Clock 010: 4-division frequency of the system clock 011: 8 frequency division of system clock 100: 16-division frequency of system clock 101: 32 frequency division of system clock 110: 64-division frequency of system clock 111: Setting prohibited <p>Note: When the PWC_STPMCR.CKSMRC bit is 1, this register is initialized after wake-up from stop mode, and PCLK1 is divided by 1 of the system clock.</p>	R/W
b3	Reserved	-	Read at "0", write at "0"	R/W
b2~0	PCLK0S[2:0]	PCLK0 clock frequency division selection bit	000: 1 frequency division of system clock 001: 2-Division Frequency of System Clock 010: 4-division frequency of the system clock 011: 8 frequency division of system clock 100: 16-division frequency of system clock 101: 32 frequency division of system clock 110: 64-division frequency of system clock 111: Setting prohibited <p>Note: When the PWC_STPMCR.CKSMRC bit is 1, this register is initialized after wake-up from stop mode, and PCLK0 is divided by 1 of the system clock.</p>	R/W

4.11.22 CMU USBFS Clock Configuration Register (CMU_UFSCKCFGREG)

Reset value: 0x40

b7	b6	b5	b4	b3	b2	b1	b0
				-	-	-	-
Bit	Marking	Place name	Function				
b7~b4	USBCKS[3:0]	48MHz clock source selection for USB-FS	0010: system clock divided by 2 0011: system clock divided by 3 0100: System clock divided by 4 1000: MPLL/P 1001: MPLL/Q 1010: MPLL/R 1011: UPLL/P 1100: UPLL/Q 1101: UPLL/R Other prohibitions Note: 1. When the switching target clock source is MPLL/UPLL, ensure that the MPLL/UPLL clock is in a stable state. 2. When the system clock selects MPLL, it is necessary to set USB, CAN, QSPI, SPI, general-purpose timer, FCM, ADC, and DAC to the module stop state, and then write the CMU_SCFGR register to switch the clock frequency division. After writing the CMU_USBCCFGREG register, the software waits for the system to stabilize for 30us. 3. When the PWC_STPMCR.CKSMRC bit is 1, after wake-up from stop mode, this register is initialized, and USBCLK is divided by 4 of the system clock.				R/W
b3~b0	Reserved	-	Read at "0", write at "0"				R/W

4.11.23 CMU AD/TRNG 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 at "0", write at "0"	R/W
b3~b0	PERICKSEL[3:0]	AD/TRNG clock source selection	0000: PCLK2/PCLK4 set by CMU_SCFGR 1000: MPLL/P 1001: MPLL/Q 1010: MPLL/R 1011: UPLL/P 1100: UPLL/Q 1101: UPLL/R In addition, the setting is prohibited.	R/W

Note:

- When the switching target clock source is MPLL/UPLL, ensure that the MPLL/UPLL clock is in a stable state.

4.11.24 CMU I2S Clock Configuration Register (CMU_I2SCKSEL)

Reset value: 0xBBB

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
I2S4CKSELc				I2S3CKSEL[3:0]				I2S2CKSEL[3:0]				I2S1CKSEL[3:0]			

Bit	Marking	Place name	Function	Read and write
b15~b12	I2S4CKSEL	I2S clock source selection	0000: PCLK3 set by CMU_SCFGR 1000: MPLL/P 1001: MPLL/Q 1010: MPLL/R 1011: UPLL/P 1100: UPLL/Q 1101: UPLL/R Other settings are prohibited. Note: There is no clock for settings other than the above	R/W
b11~b8	I2S3CKSEL	I2S clock source selection	0000: PCLK3 set by CMU_SCFGR 1000: MPLL/P 1001: MPLL/Q 1010: MPLL/R 1011: UPLL/P 1100: UPLL/Q 1101: UPLL/R Other settings are prohibited. Note: There is no clock for settings other than the above	R/W
b7~b4	I2S2CKSEL	I2S clock source selection	0000: PCLK3 set by CMU_SCFGR 1000: MPLL/P 1001: MPLL/Q 1010: MPLL/R 1011: UPLL/P 1100: UPLL/Q 1101: UPLL/R Other settings are prohibited. Note: There is no clock for settings other than the above	R/W
b3~b0	I2S1CKSEL	I2S clock source selection	0000: PCLK3 set by CMU_SCFGR 1000: MPLL/P 1001: MPLL/Q 1010: MPLL/R 1011: UPLL/P 1100: UPLL/Q 1101: UPLL/R Other settings are prohibited. Note: There is no clock for settings other than the above	R/W

Note:

- When the switching target clock source is MPLL/UPLL, it is necessary to ensure that the MPLL/UPLL oscillates in a stable state.
- When MPLL/UPLL is selected as the target clock source, refer to the detailed method of configuring MPLL/UPLL Clock Controller (CMU).

4.11.25 CMU Debug Clock Configuration Register (CMU_TPIUCKCFGR)

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 bit	0: Prohibited 1: Permissible				R/W
b6~b2	-	-	Read at "0", write at "0"				R/W
b1~0	TPIUCKS[1:0]	TPIU clock frequency division selection bit	00: 1 frequency division 01: 2 frequency division 10: 4 frequency division Other prohibitions				R/W

4.11.26 CMU MCO1 Configuration Register (CMU_MCO1CFGR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
MCO1EN		MCO1DIV[2:0]				MCO1SEL[3:0]	
Bit	Marking	Place name	Function				Read and write
b7	MCO1EN	MCO_1 Output License	0: Disable MCO_1 Output 1: Allow MCO_1 Output				R/W
b6~b4	MCO1DIV[2:0]	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	MCO1SEL[3:0]	MCO_1timer selection	0000: HRC clock 0001: MRC clock 0010: LRC clock 0011: XTAL clock 0100: XTAL32 clock 0110: MPLLP 0111: UPLL 1000: MPLLQ 1001: UPLLQ 1011: System clock Other prohibitions.				R/W

4.11.27 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 export permission		0: disable MCO_2 output 1: Enable MCO_2 output			R/W
b6~b4	MCO2DIV[2:0]	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	MCO2SEL[3:0]	MCO_2 clock source selection		0000: HRC clock 0001: MRC clock 0010: LRC clock 0011: XTAL clock 0100: XTAL32 clock 0110: MPLLP 0111: UPLL 1000: MPLLQ 1001: UPLLQ 1011: System clock Other prohibitions.			R/W

4.11.28 FCM Lower Limit Compare Value Register (FCM_LVR)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read at "0", write at "0"	R/W
b15~b0	LVR[15:0]	lower limit comparison value	A 0 in the START bit configures this register.	R/W

4.11.29 FCM Upper Limit Comparison Value Register (FCM_UVR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UVR[15:0]															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	Read at "0", write at "0"										R/W		
b15~b0	UVR[15:0]	Upper limit comparison value	A 0 in the START bit configures this register.										R/W		

4.11.30 FCM Counter Value Register (FCM_CNTR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNTR[15:0]															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	Read at "0", write at "0"										R/W		
b15~b0	CNTR[15:0]	Counter value	Save the counter value to this register when a valid edge selected by the EDGES bit of the base clock is detected (except for the first valid edge after START=1)										R		

4.11.31 FCM Start Stop Register (FCM_STR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Bit	Marking	Place name	Function										Read and write		
b31~b1	Reserved	-	Read at "0", write at "0"										R/W		
b0	START	Frequency measurement start bit	0: Stop frequency measurement 1: Frequency measurement starts										R/W		

4.11.32 FCM Measurement Object Control Register (FCM_MCCR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-															
Bit	Marking		Place name		Function										Read and write
b31~b8	Reserved		-		Read at "0", write at "0"										R/W
b7~b4	MCKS[3:0]	Measurement object clock selection bit		0000: XTAL 0001: XTAL32 0010: HRC 0011: LRC 0100: SWDTRC 0101: PCLK1 0110: UPLL 0111: MRC 1000: MPLLP 1001: RTCLRC Others: Prohibiting											R/W
b3~b2	Reserved		-		Read at "0", write at "0"										R/W
b1~b0	MDIVS[1:0]	Measurement object frequency division selection		00: no frequency division 01: 4 frequency division 10: 8 frequency division 11: 32 frequency division											R/W

4.11.33 FCM Measurement Reference Control Register (FCM_RCCR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16						
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0						
EXR EFE	-	EDGES[1:0]	-	-	DNFS[1:0]	INE XS	RCKS[3:0]	-	RDIVS[1:0]												
<hr/>																					
Bit	Marking		Place name			Function								Read and write							
b31~b16	Reserved		-			Read at "0", write at "0"								R/W							
b15	EXREFE		External pininput reference clock FCMREF enable bit			0: Disable external pininput reference clock FCMREF 1: Allow external pins to input reference clock FCMREF								R/W							
b14	Reserved		-			Read at "0", write at "0"								R/W							
b13~b12	EDGES[1:0]		Measurement reference edge selection bit			00: Rising edge 01: Falling edge 10: rising and falling edge 11: Setting prohibited								R/W							
b11~b10	Reserved		-			Read at "0", write at "0"								R/W							
b9~b8	DNFS[1:0]		Digital filter function selection bit			00: no filter function 01: The clock selected by the MCKS bit is used as the filter clock 10: The frequency of the clock selected by the MCKS bit is divided by 4 as the filter clock 11: 16 frequency division of the clock selected by the MCKS bit as the filter clock								R/W							
b7	INEXS		Measurement reference, internal clock and terminal selection bits			0: External pininput reference clock FCMREF 1: The clock selected by the RCKS selection bit								R/W							
b6~b3	RCKS[3:0]		Measurement reference clock selection bit			0000: XTAL 0001: XTAL32 0010: HRC 0011: LRC 0100: SWDTRC 0101: PCLK1 0110: UPLL 0111: MRC 1000: MPLLP 1001: RTCLRC Other: Set Prohibitions								R/W							
b2	Reserved		-			Read at "0", write at "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								R/W							

4.11.34 FCM Interrupt Reset Control Register (FCM_RIER)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
<hr/>															
Bit		Marking				Place name				Function				Read and write	
b31~b8		Reserved				-				Read at "0", write at "0"				R/W	
b7	ERRE				Frequency abnormal reset enable bit				0: Prohibited 1: Permissible				R/W		
b6~b5		Reserved				-				Read at "0", write at "0"				R/W	
b4	ERRINTRS				Frequency exception interrupt reset selection bit				0: Frequency abnormal interrupt 1: Reset when frequency abnormality occurs				R/W		
b3	Reserved				-				Read at "0", write at "0"				R/W		
b2	OVFIE				Counter overflow interrupt enable bit				0: Disable counter overflow interrupt 1: Enable counter overflow interrupt				R/W		
b1	MENDIE				End of measurement interrupt enable bit				0: disable interrupt at the end of measurement 1: Enable interrupt at end of measurement				R/W		
b0	ERRIE				Frequency exception interrupt enable bit				0: Disable frequency exception interrupt 1: Allow frequency exception to interrupt				R/W		

4.11.35 FCM Flags Register (FCM_SR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
<hr/>															
Bit		Marking				Place name				Function				Read and write	
b31~b8		Reserved				-				Read at "0", write at "0"				R/W	
b2	OVF				counter overflow flag				0: The counter has not overflowed 1: Counter overflow				R		
b1	MENDF				Measurement end flag				0: measuring 1: End of measurement				R		
b0	ERRF				Frequency abnormal flag bit				0: No frequency exception occurs 1: Abnormal occurrence frequency				R		

4.11.36 FCM Flag Clear Register (FCM_CLR)

Reset value: 0x0000_0000

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

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

5 Power Control (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 control logic (PWCL) and power voltage detection unit (PWD).

The operating voltage (VCC) of the chip is 1.8 V to 3.6 V. A voltage regulator (LDO) supplies power to the VDD and VDDR domains, and a VDDR voltage regulator (RLDO) supplies power to the VDDR domain power-down mode. The chip provides three operating modes of super high speed, high speed and super low speed through the power control logic (PWCL), and three low power modes such as sleep, stop and power down.

The power voltage detection unit (PWD) provides functions such as power-on reset (POR), power-down reset (PDR), brown-out reset (BOR), programmable voltage detection 1 (PWD1), and programmable voltage detection 2 (PWD2). Among them, POR, PDR, and BOR control the reset action of the chip by detecting the VCC voltage. PWD1 detects the VCC voltage and resets or interrupts the chip according to the register settings. PWD2 detects VCC voltage or external input detection voltage, and generates reset or interrupt according to register selection.

After the chip enters power-down mode, the VDDR area can maintain power through RLDO to ensure that the real-time clock module (RTC) and wake-up timer (WKT) can continue to operate, and maintain the data of 4KB low-power SRAM (Ret-SRAM). The analog blocks are equipped with dedicated supply pins for improved analog performance.

5.2 Distribution of Power

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

The VCC domain is powered by the VCC/VSS pin, which is composed of power control logic (PWCL), power voltage detection unit (PWD), IO level holding circuit, voltage regulator (LDO), VDDR domain voltage regulator (RLDO) and oscillator circuit configuration. The oscillator circuit includes an external high-speed oscillator (XTAL), an external low-speed oscillator (XTAL32), an internal low-speed oscillator (LRC), etc.

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

The VDDR domain is composed of 4KB holding RAM (Ret-SRAM), real-time clock (RTC), and wake-up timer (WKT). Power is supplied by RLDO in power-down mode, and by LDO in modes other than power-down mode. In power-down mode, Ret-SRAM can keep data, real-time clock RTC and wake-

up timer WKTM can continue to operate. When the function of the VDDR domain is not needed, you can cut off the power supply of the VDDR domain power-down mode by setting PWC_PWRC0.VVDRSD to further reduce power consumption.

The analog power domain is mainly composed of a digital-to-analog converter (ADC), a comparator (CMP), a programmable gain amplifier (PGA), and input and output pins of the analog system, and is powered by the AVCC/AVSS pin. In order to provide high-precision analog performance, the analog area is equipped with an independent power supply. In order to ensure that the ADC has higher precision, the reference voltage VREFH of the ADC uses a dedicated pin.

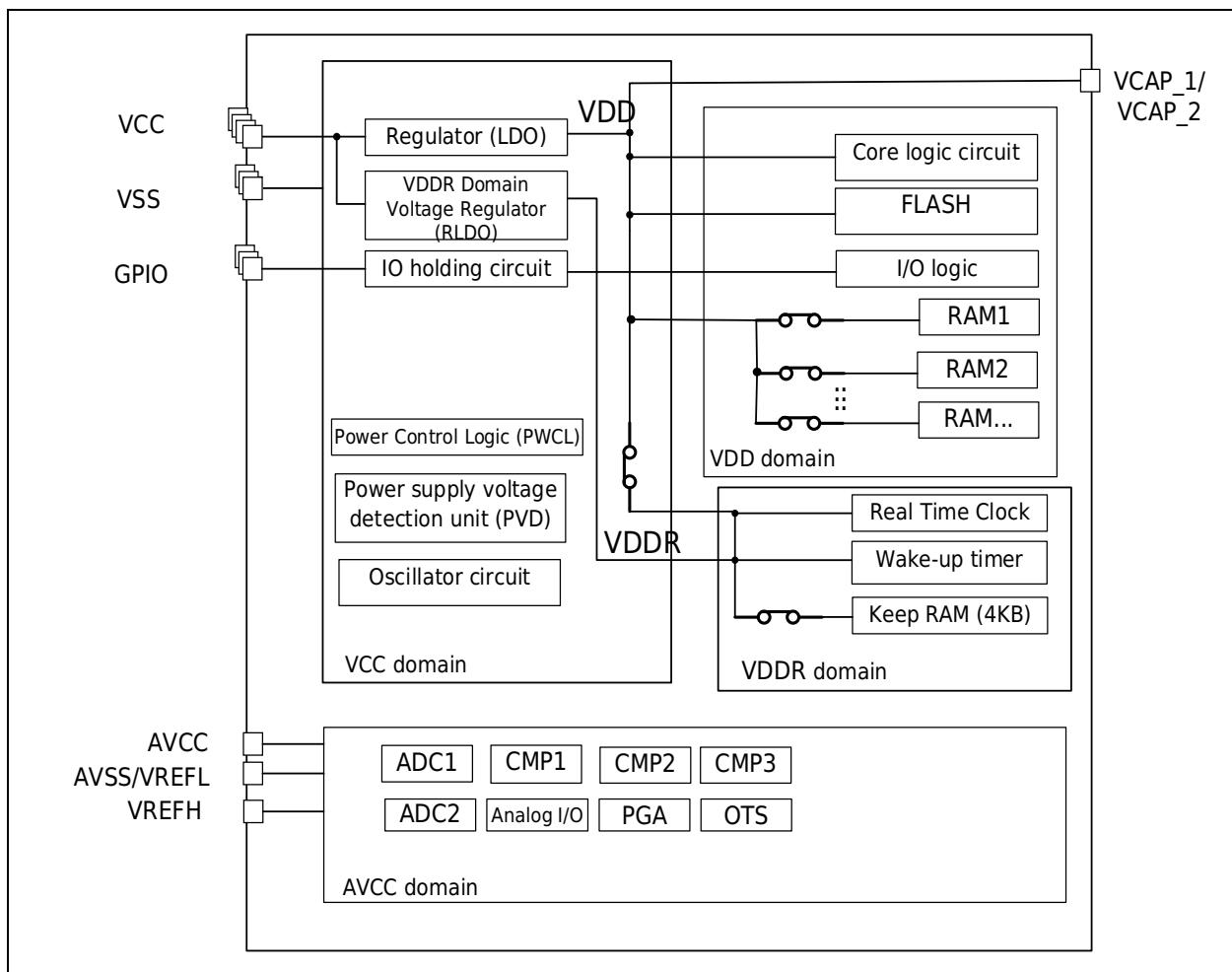


Figure 5-1 Power supply diagram

5.3 Power Voltage Detection Unit (PVD) Description

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

5.3.1 Description of Action of Power-on/Power-off Reset

The chip is integrated with power-on reset circuit and power-off reset circuit. The power-on reset and power-off reset waveforms are as shown Figure 5-2. When VCC is higher than the specified threshold V_{POR} , after the T_{RSTPOR} time, the chip releases the power-on reset state, and the CPU starts to execute code. When VCC is lower than V_{PDR} , the chip remains in reset state. When power-on reset is used, the reset pin NRST must be 1. If the reset pin is pulled down, the chip will be reset and started by pin reset.

For details of parameters such as V_{POR} , V_{PDR} , and T_{RSTPOR} , please refer to the **electrical characteristics in the Data Manual**.

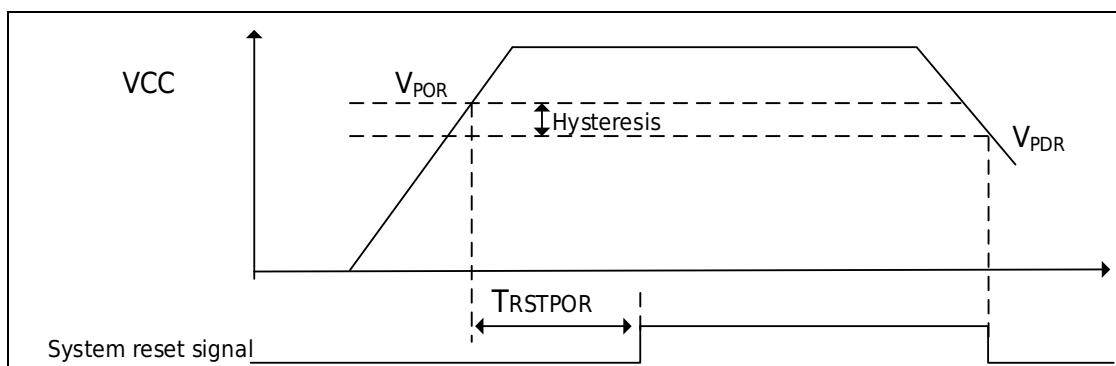


Figure 5-2 Power-on Rset, power-off Reset Waveforms

5.3.2 Brown-out Reset (BOR) Description

During power-up, until VCC is higher than BOR, a brown-out reset (BOR) will put the chip in reset.

The VBOR threshold is configured through the BORLEV, BORDIS of the initialization configuration bits (ICG). When BORDIS=0, the BOR detection voltage can be selected from 4 thresholds. When BORDIS is configured as 1, the chip performs reset control through power-on reset and power-off reset.

BORDIS	BORLEV	Note
1	XX	BOR invalid
0	00	BOR Valid, Select BOR Threshold 0 (VBOR0)
0	01	BOR Valid, Select BOR Threshold 1 (VBOR1)
0	10	BOR Valid, Select BOR Threshold 2 (VBOR2)
0	11	BOR Valid, Select BOR Threshold 3 (VBOR3)

Table 5-1 BOR configuration

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

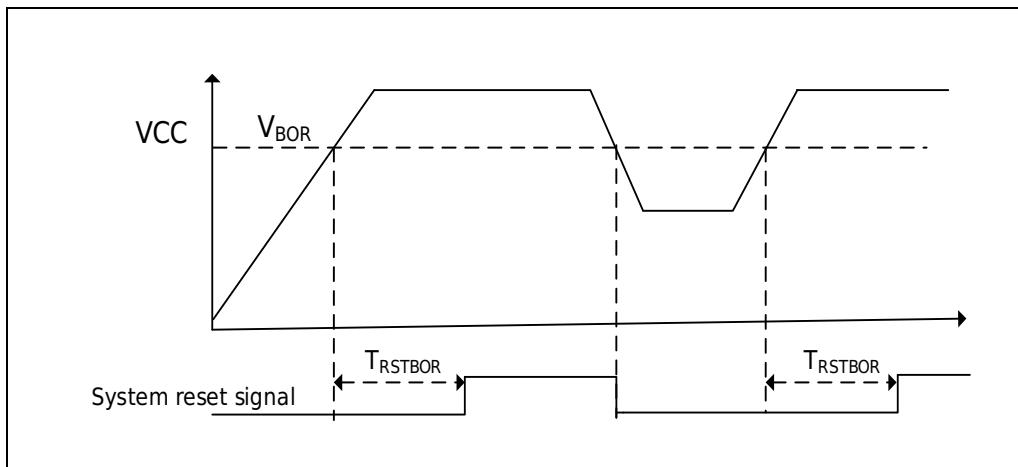


Figure 5-3 Brownout Reset Waveform

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 power supply voltage passes the detection threshold. Each detection circuit can be programmed and configured separately.

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

The main features of the programmable voltage detection are shown Table 5-2 below.

Table 5-2 PVD1/PVD2 Characteristics

Item	PVD1	PVD2
Test object	Whether VCC has passed the threshold voltage point (VPVD1) during the falling/rising process	<ol style="list-style-type: none">When PWC_PVDLCR.PVD2LVL[2:0] is set to a value other than 111, whether VCC has passed the threshold voltage point (VPVD2) during the falling/rising processWhen PWC_PVDLCR.PVD2LVL[2:0]=111, whether the drop/rise process of the external input voltage passes the threshold voltage point (VPVD2)
Detection voltage point	Configured by PVD1LVL[2:0]	Configured by PVD2LVL[2:0]
Reduction	Reset: VCC<VPVD1; Reset release: VCC>VPVD1 after a certain reset processing time.	Reset: VCC<VPVD2; Reset release: VCC>VPVD2 after a certain reset processing time.
Interrupt	Configured as voltage detection 1 interrupt or non-maskable interrupt VCC drops past the threshold voltage point (VPVD1)	Configured as voltage detection 2 interrupt or non-maskable interrupt VCC drops past the threshold voltage point (VPVD2)
Filtering function	Digital filtering	Digital filtering
Peripheral circuit trigger function	VCC drops past the threshold voltage point (VPVD1)	VCC drops past the threshold voltage point (VPVD2)

5.3.4 PVD1, PVD2 Interrupt/Reset Block Diagram

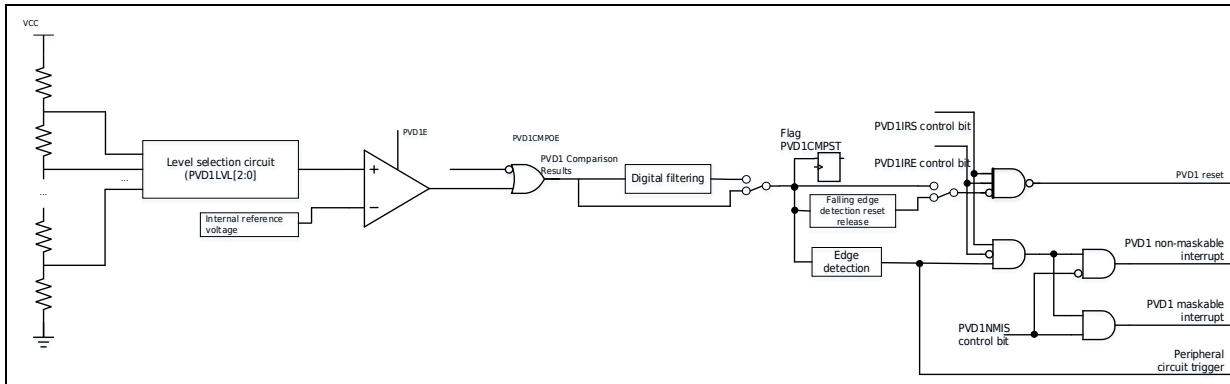


Figure 5-4 PVD1 Interrupt/Reset Block Diagram

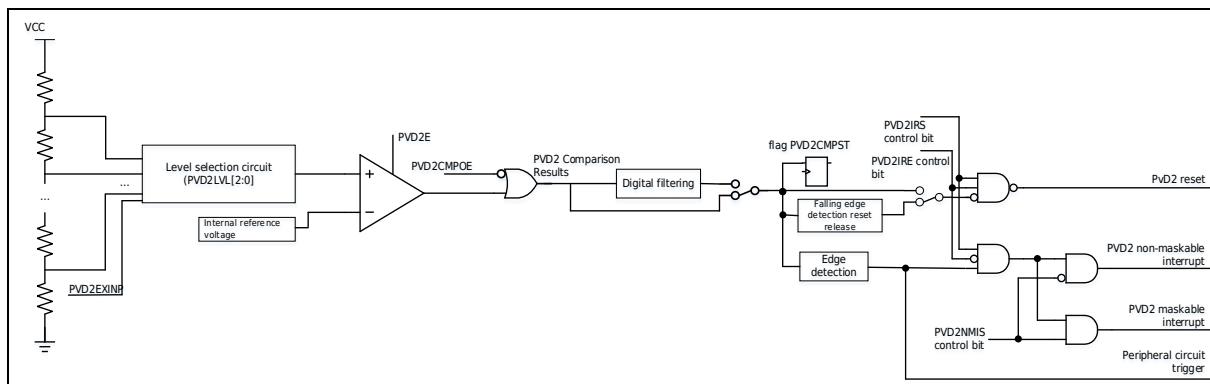


Figure 5-5 Interrupt/Reset Block Diagram

5.3.5 Input/Output Pin

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

5.3.6 PVD1 Interrupt and Reset

When using the PVD1 circuit in stop mode or power-down mode, 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) PVD1IRS is set to 0, and PVD1 is selected to generate an interrupt; when the reset function is selected, power-down mode cannot be entered.

The figure below is the running timing diagram of the voltage monitoring 1 interrupt. PVD1DETFLG needs to be cleared before the interrupt can occur again.

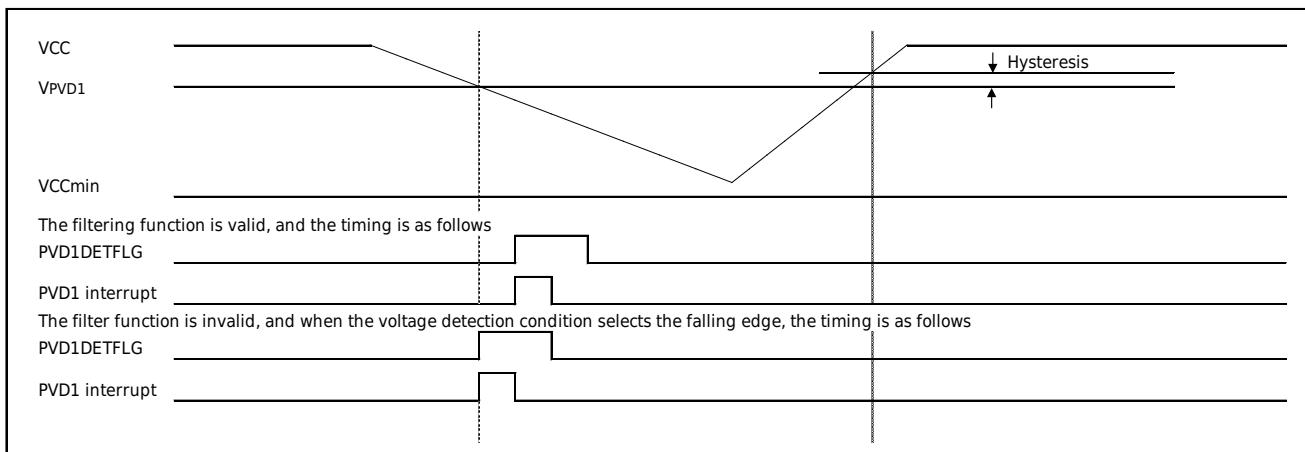


Figure 5-6 Power Monitor 1 Interrupt Timing Diagram

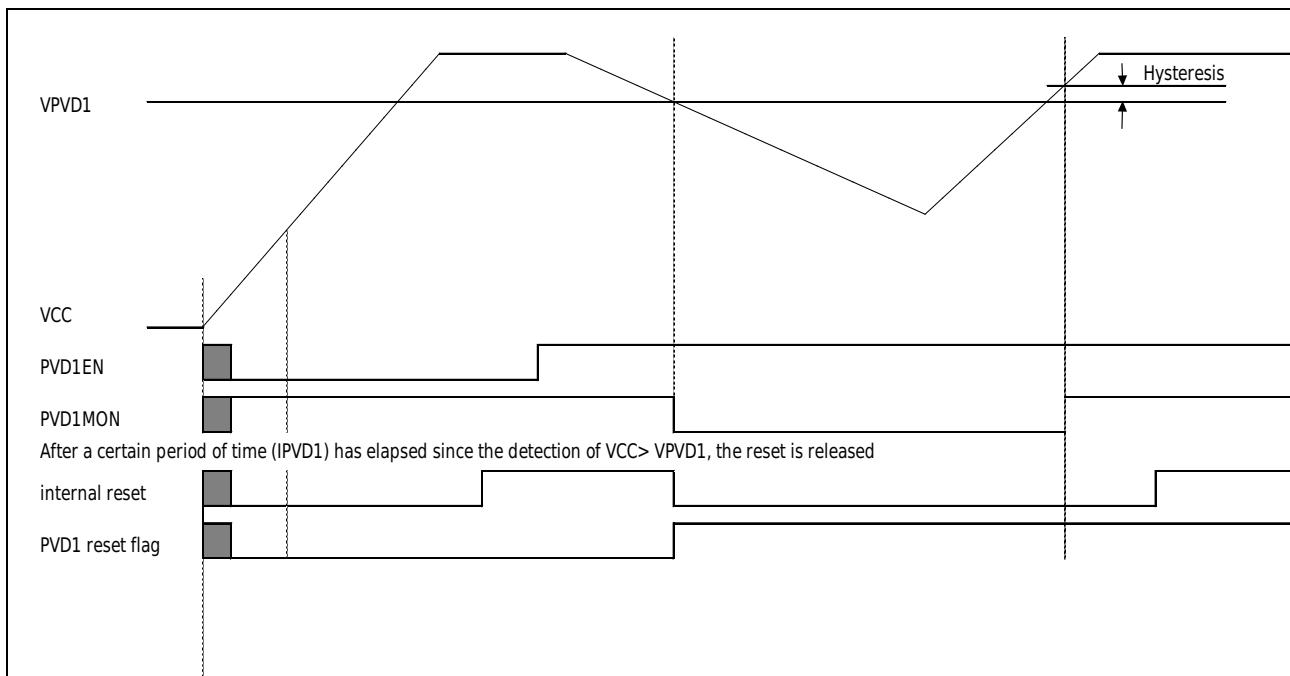


Figure 5-7 Power Monitor 1 Reset Timing Diagram

5.3.7 PVD2 Interrupt and Reset

When using the PVD2 circuit in stop mode or power-down mode, 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) PVD2IRS is set to 0, and PVD2 is selected to generate an interrupt; when the reset function is selected, power-down mode cannot be entered.

The figure below is the running timing diagram of the voltage monitoring 2 interrupt. PVD2DETFLG needs to be cleared before the interrupt can occur again.

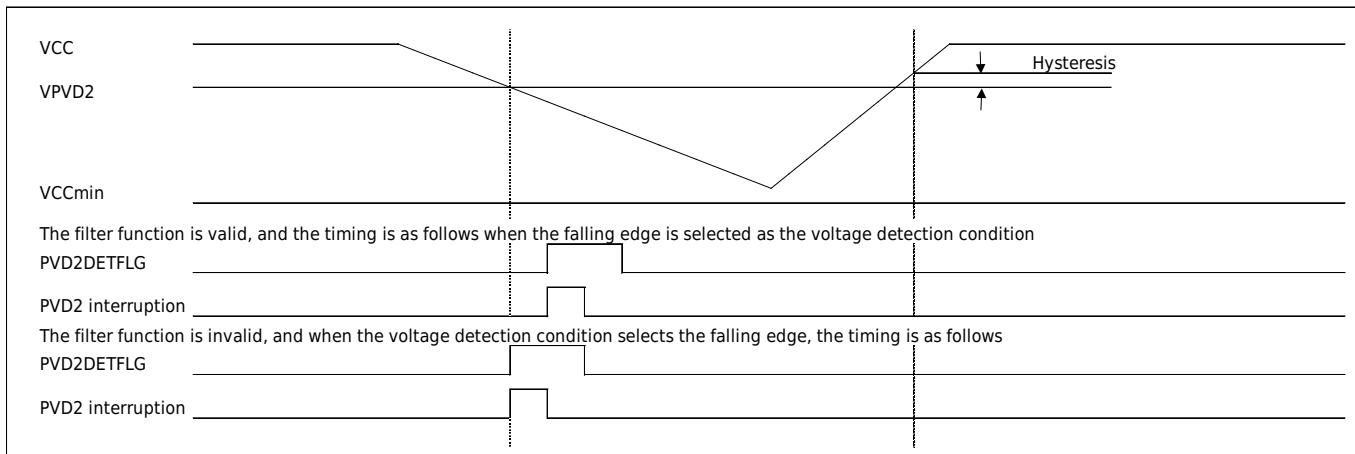


Figure 5-8 Power Supply Monitoring 2 Interrupt Operation Timing Diagram

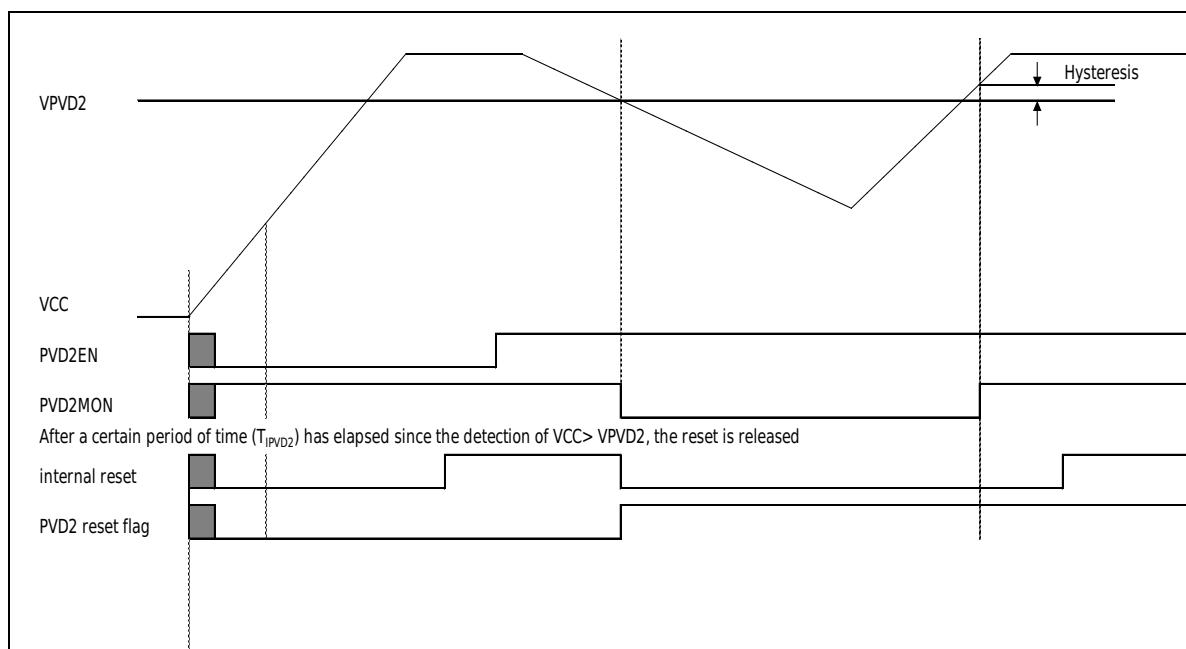


Figure 5-9 Power supply monitoring 2 reset operation sequence diagram

5.3.8 Internal Voltage Sampling and Detection Function

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

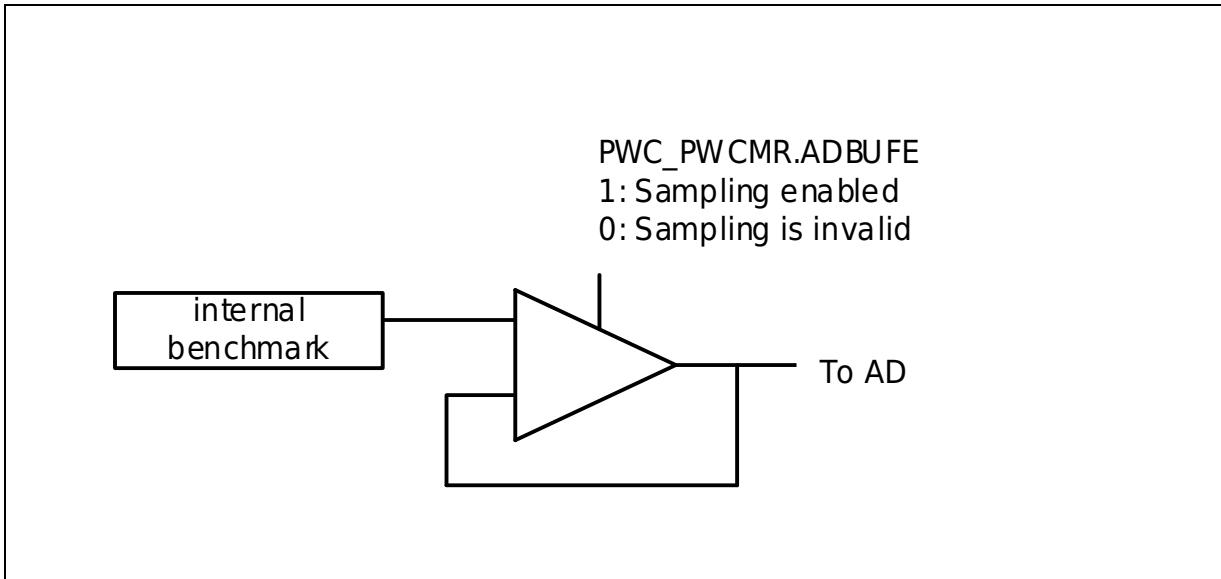


Figure 5-10 Internal Voltage Sampling Diagram

■ Reference Voltage Measurement Path

When using the reference voltage measurement channel, you need to select the reference voltage measurement channel according to the following steps.

1. PWC_PWCMR.ADBUFE=1, enable internal voltage measurement function
2. Configure the register according to Table 16-2 in the ADC chapter to enable the ADC to gate the internal voltage measurement channel
3. Use ADC to measure internal reference voltage after waiting 50uS

5.4 Action Mode and Low Power Mode

After the system reset or power-on reset, all the power fields of the chip are in the power supply state, and the chip enters the speed running mode. In run mode, the CPU provides the clock through HCLK and executes program code. The chip provides three operating modes: ultra-high-speed operation mode, high-speed operation mode, and ultra-low-speed operation mode. The operating modes in which the chip can be configured are asTable 5-3shown.

In order to save power consumption when the CPU does not need to run, the system provides three low-power modes: sleep mode, stop mode, and power-down mode. The low power consumption modes that the chip can be configured are asTable 5-4shown. In sleep mode, the Cortex™-M4F core of the chip stops operating, and the peripherals keep running; in stop mode, the peripherals and CPU of the chip stop operating; in power-down mode, the power supply of VDD domain is turned off, and the VDD domain Peripherals stop operating. The real-time clock and wake-up timer in the VDDR domain can operate in low-power mode, and the SRAM can keep data; when the real-time clock, wake-up timer and SRAM in the VDDR domain are not used, the VDDR domain can be set to be closed. The voltage regulator RLDO can further reduce power consumption after entering power-down mode.

Users can choose operating mode and low power mode based on the application to find the best balance between low power consumption, short start-up time, 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 asTable 5-5shown.

Table 5-3 Operation Mode

Operating mode	Note
Ultra high Speed Run Mode	Main frequency below 200MHz
High Speed Run Mode	Main frequency below 168MHz
Ultra low Speed Run Mode	Main frequency below 8MHz

Table 5-4 Low Power Modes

Pattern	Note	
Sleep Mode	CPU clock stopped, peripherals keep running	
Stop Mode	The clocks of the chip peripherals and the CPU are stopped	
Power Down Mode	PDMD1	VDD domain power down
	PDMD2	The voltage detection unit is invalid except the VDD domain is powered off
	PDMD3	In addition to VDD domain power-off, VDDR domain power-off, power-on reset circuit enters low power consumption mode, and voltage detection unit (PVD) is invalid. Compared with power-down mode 4, other chips except PWC_PDWKF0/PWC_PDWKF1/RSTF0 when power-down wake up Complete reset.

	PDMD4	In addition to the VDD domain power-off, the VDDR domain power-off, the power-on reset circuit enters the low power consumption mode, and the voltage detection unit (PVD) is invalid
--	-------	---

Table 5-5 Operating conditions of low power consumption mode and status of each module in low power consumption mode

Item	Sleep mode	Stop mode	Power Down Mode
Entry	PWC_STPMCR.STOP=0 PWC_PWR0.PWDN=0, WFI	PWC_STPMCR.STOP=1 PWC_PWR0.PWDN=0, WFI	PWC_STPMCR.STOP=1 PWC_PWR0.PWDN=1, WFI
Release	To interrupt or reset arbitrarily.	Interrupt or reset that can be used in stop mode	Wake-up event or reset available in power-down mode
External high-speed oscillator	Work can be done	Stop	Stop
External low-speed oscillator	Work can be done	Work can be done	Work can be done
Internal high-speed oscillator	Work can be done	Stop	power down
internal medium speed oscillator	Work can be done	Stop	power down
Internal low-speed oscillator	Work can be done	Work can be done	Work can be done
WDT dedicated clock oscillator	Work can be done	Work can be done	power down
PLL	Work can be done	Stop	power down
PLLM	Work can be done	Stop	power down
CPU	To stop (hold).	To stop (hold).	power down
RAM	Work can be done Can be set to work, power down	To stop (hold). According to the setting before entering standby, it can maintain power-off or sleep	power down
FLASH	Work can be done	To stop (hold).	power down, content remains
DMA	Work can be done	To stop (hold).	power down
Voltage regulator	Work Drive adjustable	Work Drive adjustable	Stop
Power-on reset circuit	Work	Work	Work The accuracy of the reset circuit in power-down mode 1 and power-down mode 2 can be guaranteed, and the power-on reset voltage in power-down mode 3 and power-down mode 4 is not guaranteed
Brown-out Reset BOR	Work can be done	Work can be done	Power-down mode 1 work can be set Stop in power-down mode 2/3/4
Voltage detection module PVD	Work can be done	Work can be done	Power-down mode 1 work can be set Stop in power-down mode 2/3/4
WDT	Work can be done	To stop (hold).	power down
SWDT	Work can be done	Work can be done	power down
RTC	Work can be done	Work can be done	Power-down mode 1/2 work can be set

Item	Sleep mode	Stop mode	Power Down Mode
			Power down in power down mode 3/4
USB-FS	Work can be done	To stop (hold).	power down
Timer0	Work can be done	Work can be done	power down
Ret-SRAM	Work can be done Can be set to work, power down, sleep	To stop (hold). Can be set to power down, sleep	Stop (hold) in power-down mode 1/2 Can be set to power down, sleep Power-down mode 3/4 power-down
WKTM	Work can be done	Work can be done	Power-down mode 1/2 work can be set Power down in power down mode 3/4
Other peripheral modules	Work can be done	To stop (hold).	power down
AD	Work can be done	Stop	power down
DA	Work can be done	Work can be done	power down
PGA	Work can be done	Work can be done	power down
CMP	Work can be done	Work can be done	power down
PA0-PA10, PB0-PB2, PB5-PB10, PB12-PB15, PC0-PC13, PD0-PD15, PE0-PE15, PH2	Work can be done	Hold on	Hold or high resistance
PC14-PC15	Work can be done	When used as an external low-speed oscillator pin, keep the oscillator active; when set as GPIO or other peripheral functions, please set to keep the two pins at the same level	When set to GPIO or other peripheral functions, the state 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	Work can be done	When used as an external high-speed oscillator, oscillator stops concussion 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, oscillator stops concussion and the pin state keeps the state before entering STOP mode. When GPIO or other peripheral functions are set, keep the status before STOP
NRST reset pin	The chip is pulled out of the circuit to the VCC	Outside the chip is pulled up to VCC through resistance	Outside the chip is pulled up to VCC through resistance
PA11-PA12	Work can be done	Maintain; Since redundant current will be generated when the level of this pin is pulled high, the pull-up is prohibited when entering STOP mode.	hold or high resistance; Since the level of this pin will generate redundant current when it is pulled high, the pull-up is prohibited when entering power-down mode.
PB11/MD	Work can be done	Maintain;	Maintain; Outside the chip is pulled up to VCC through resistance
PA13-PA15, PB3,PB4	Work may be set up; The built-in pull-up circuit works as a JTAG function	Maintain; The built-in pull-up circuit works as a JTAG function	Maintain; The built-in pull-up circuit works as a JTAG function

5.4.1 Operating Mode

The chip has three operating modes: ultra-high speed, 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. AsTable 5-6shown, according to the DVS and DDAS bits of PWC_PWRC2, the chip can work in the corresponding operation mode. When the chip selection action is in ultra-low speed mode, FLASH and RAM also need to be set to work at low core voltage, so register bits EFM_FRMC.LVM=1, PWC_RAMOPM=0x9062 need to be set.

Table 5-6 Operating Mode Description

Operating mode	Frequency Range	Register setting	
		PWC_PWRC2.DVS	PWC_PWRC2.DDAS
ultra-high-speed operation mode	≤200MHz	00	1111
high speed mode	≤ 168MHz or below	11	1111
Ultra-low speed operation mode*	≤ 8MHz or below	10	1000

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

The switching between ultra-high-speed operation mode, high-speed operation mode, and ultra-low-speed operation mode needs to follow the following process 1~process 6.

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 clock sources and modules that are not needed in ultra-low speed mode, and confirm that the FLASH is not in the programming or erasing state
- 3) Set FLASH FRMC.LVM=1, RAM operation mode register PWC_RAMOPM is set to 0x9062
- 4) Confirm FRMC.LVM=1, PWC_RAMOPM=0x9062
- 5) Set PWC_PWRC2.DDAS[3:0] to 1000; PWC_PWRC2.DVS[1:0] to 10
- 6) Write PWC_MDSWCR=0x10
- 7) Wait for TSWMD1 (30uS)
- 8) 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] to 1111; PWC_PWRC2. DVS[1:0] to 11 according to system frequency requirements
- 2) Write PWC_MDSWCR=0x10
- 3) Waiting for TSWMD 2 (30uS)
- 4) Set FLASH FRMC.LVM=0, RAM operation mode register PWC_RAMOPM is set to 0x8043

5) Confirm FRMC.LVM=0, PWC_RAMOPM=0x8043

6) The chip operates in high-speed mode

3. Switching from high-speed mode to ultra-high-speed mode

1) Set PWC_PWRC2.DDAS[3:0] to 1111; PWC_PWRC2.DVS[1:0] to 00

2) Write PWC_MDSWCR = 0x10

3) Wait for T SWMD2 (30uS)

4) The chip operates in ultra-high speed mode

4. Switching from super high speed mode to high speed mode

1) Set the clock source used in high-speed mode to ensure that the clock source meets the frequency requirements in high-speed mode

2) Set PWC_PWRC2.DDAS[3:0] to 1111; PWC_PWRC2.DVS[1:0] to 11

3) Write PWC_MDSWCR = 0x10

4) Waiting for TSWMD 2 (30uS)

5) The chip operates in high-speed mode

5. Switch from super low speed mode to super high speed mode

1) Set PWC_PWRC2.DDAS[3:0] to 1111; PWC_PWRC2.DVS[1:0] to 00

2) Write PWC_MDSWCR = 0x10

3) Waiting for TSWMD 2 (30uS)

4) Set FLASH FRMC.LVM=0, RAM operation mode register PWC_RAMOPM is set to 0x8043

5) Confirm FRMC.LVM=0, PWC_RAMOPM=0x8043

6) The chip operates in ultra-high speed mode

6. Switch from super high speed mode to super 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 clock sources and modules that are not needed in ultra-low speed mode, and confirm that the FLASH is not in the programming or erasing state

3) Set FLASH FRMC.LVM=1, RAM operation mode register PWC_RAMOPM is set to 0x9062

4) Confirm FRMC.LVM=1, RAMOPT=0x9062

5) Set PWC_PWRC2.DDAS[3:0] to 1000; PWC_PWRC2.DVS[1:0] to 10

6) Write PWC_MDSWCR = 0x10

7) Wait for TSWMD1 (30uS)

8) The chip operates in ultra-low speed mode

5.4.2 Sleep Mode

In sleep mode, the CPU stops running and its internal register remains in sleep mode. The operating status of peripherals and other system modules other than the watchdog and dedicated watchdog does not change.

When the ICG is set to start automatically, if the WDTSLPOFF bit of ICG is 1, the watchdog stops counting in sleep mode; if the WDTSLPOFF bit is 0, the watchdog continues counting in sleep mode. If the ICG is not set to start automatically, start the watchdog by software startup, 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 is in sleep mode does not stop counting.

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

- Enter sleep mode
 - When PWC_STPMCR.STOP=0, WFI instructions can be executed to enter sleep mode.
- Quit sleep mode
 - Any interrupt and reset can wake the chip from sleep mode. When waking up by interrupt, the chip enters interrupt processing program. The chip enters the reset state when the reset exits the sleep mode.

5.4.3 Stop Mode

In stop mode, the CPU, most peripherals, and timer all stop. The chip maintains CPU internal register and SRAM data, peripheral state and pin state. In the stop mode, because most of the timer stops working, the voltage regulator also reduces the drive capability, so the chip power consumption will be significantly reduced.

If the SWDTSLPOFF bit of ICG is set to 1, the dedicated watchdog stops counting in stop mode. If SWDTSLPOFF bit is 0, the dedicated watchdog continues counting in stop mode.

Before you execute the WFI command into stop mode, make sure that FLASH is not programmed or erased and that the oscillation stop monitoring function is invalid, otherwise the chip will enter sleep mode instead of 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" and write 1 to the ADC corresponding bit in the register PWC_FCG3, so that the ADC enters the module stop state, and then executes the WFI instruction to enter the stop mode.

When waking up from STOP mode, use the bits CKSMRC and FLNWT in the PWC_STPMCR register to select the clock after waking up and whether to wait for the FLASH to stabilize. CKSMRC is used to control the clock source after wake-up,

When CKSMRC=1, the wake-up system clock source is selected as MRC; when CKSMRC=0, the wake-up system clock remains the same as the clock source before entering STOP. FLNWT is used to control whether to wait for the FLASH to be stable after waking up. When FLNWT=0, it needs to wait for the FLASH to be stable when waking up; when FLNWT=1, it does not need to wait for the FLASH to be stable when waking up. FLNWT must be set when the program is running on RAM, otherwise the chip's action after waking up from STOP is not guaranteed. When the program runs on RAM and enters STOP mode, select CKSMRC = 1, FLNWT = 1 to 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 have unguaranteed actions.

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

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

When the chip enters STOP mode after setting PWC_PWRC1.STPDAS=11 in ultra-low speed mode; if PWC_PWRC1.STPDAS=00 and enters STOP mode, the chip will consume more current in STOP mode.

When the stop mode is related through a non-mail interrupt, the corresponding bit of int_nmier need to enable the interrupt; when the state mode is Released Throud a Maskable Interrupt, The Corresponding bit of the int_wupenr register needed to be set to enable the wake-up permission of the interrupt. Before executing the WFI or WFE command, it is necessary to ensure that all interrupts not used for wake-up from stop mode have been disabled.

- Enter stop mode

When PWC_STPMCR.STOP=1, PWC_PWRC0.PWDN=0, execute the WFI command to enter the stop mode. Table 5-5 The status of the chip's peripherals and clock sources in stop mode is given.

- Unstop mode

The stop mode can be removed by resetting and terminating. The reset methods that can be used to release stop mode include pin reset, power-on reset, brown-out reset (BOR), programmable voltage monitoring 1/2 reset, and dedicated watchdog reset. The interrupt events that can be used to unblock the stop mode are as follows:

NMI pinterrupt, pinterrupt EIRQ0-15, voltage monitoring 1 interrupt,
Voltage monitoring 2 interrupt, dedicated watchdog underflow interrupt, real-time clock cycle
interrupt, alarm clock interrupt,

Wake-up timer interrupt, comparator interrupt, USART1 RX interrupt, Timer0 compare match interrupt

When the chip interrupts the stop mode, it starts the used timer before entering the stop mode.

After all of the timer stabilizes, the chip unstops the mode.

5.4.4 Power Down Mode

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

When the ICG is set to start automatically, if the SWDTSLPOFF bit of the ICG is 1, the dedicated watchdog will be the same as other modules in the VDD domain, the power will be cut off and no longer count. If the SWDTSLPOFF 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 ICG.

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

Before executing the WFI command 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 monitoring 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) Chips with only VCAP_1 pins can use 0.1uF or 0.22uF capacitors. When waking up from power-down mode, VCAP_1/VCAP_2 needs to be charged during the core voltage establishment process. On the one hand, the smaller VCAP_1/VCAP_2 total capacity can shorten the charging time and bring fast response capability 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 capacity of VCAP_1/VCAP_2 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 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 power-down mode.

By setting PWC_PWRC0.PDMDS[1:0], the power consumption in power-down mode can be further reduced. The sub-modes of power-down mode are Table 5-7 shown. In power-down mode 1, the voltage monitoring circuit can be used, and the power-on reset detection circuit is in the active state. Since there is no need to wait for the stability of the VCC domain reference voltage circuit, voltage monitoring circuit , and power-on reset detection circuit when waking up, low power

consumption is realized. At the same time, the wake-up time is the shortest. In power-down mode 2, the reference voltage circuit in the VCC domain and the voltage monitoring circuit stop working, and the power-on reset detection circuit is in an active state. When waking up, it needs to wait for the stabilization time of the reference voltage circuit in the VCC domain and the voltage monitoring circuit. In power-down mode 3, the VCC domain reference voltage circuit, voltage monitoring circuit, and power-on reset detection circuit all stop working. It is necessary to wait for these circuits to stabilize when waking up. Therefore, while achieving the lowest power consumption, the wake-up time is shorter than power-down mode 2, and power-down mode 1 long. Power-down mode 4 has the same circuit that stops working in 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 [Electrical Characteristics - Low Power Mode].

The VDDR domain can work in power-down mode 1 and power-down mode 2, so the real-time clock module, wake-up timer can continue to run, and can be used to wake up power-down mode. Ret-SRAM can still retain data in power-down mode. If the real-time clock, wake-up timer, and Ret-SRAM are not needed in power-down mode, you can set PWC_PWRC0.VVDRSD to turn off the low-power voltage regulator to further reduce power consumption. In power-down mode 3 and power-down mode 4, the VDDR domain is also powered down.

Table 5-7 Power-down Mode Submodes

Power Down Mode	PDMD[1:0]	Power consumption	wake up time	Note
Power down mode 1	00	I _{PD1}	T _{PD1}	VCC domain supply voltage detection unit is valid
Power down mode 2	10	I _{PD2}	T _{PD2}	VCC domain POR, PDR detection circuit is valid, BOR, PVD1, PVD2 are invalid
Power down mode 3	01	I _{PD3}	T _{PD3}	VCC domain POR, PDR, BOR, PVD1, PVD2 invalid VDDR Domain Power Down
Power down mode 4	11	I _{PD4}	T _{PD4}	VCC domain POR, PDR, BOR, PVD1, PVD2 invalid VDDR Domain Power Down

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

- enter power-down mode

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

- Release power-down mode

Power-down mode can be released by power-down mode wake-up event or reset. The resets that can be used to wake up the power-down mode include pin reset, power-on reset and voltage monitoring 0 reset. Events that can be used to wake up from power-down mode include:

NMI wake-up event, WKUPn_0/1/2/3 (n=0/1/2/3) wake-up event, real-time clock alarm and timing events, voltage monitoring 1 wake-up event, voltage monitoring 2 wake-up event, wake-up timer wake-up event

After waking up from power-down mode 1 and power-down mode 2, reset the chip and execute the program again. The wake-up event can be queried through the power-down wake-up flag bit, and the reset flag bit can be queried through RSTF0.PDRF.

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

In power-down mode 4, POR, PDR, BOR, PVD1, and PVD2 circuits are all in an invalid state, and the program is re-executed after the chip is reset; the reset flag bit can be queried through RSTF0.PDRF. When waking up from power-down mode 4, the RTC/WKTM in the VDDR domain is reset, and the data in Ret-SRAM cannot be guaranteed.

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

When waking up from power-down mode, the VDD domain will re-supply power, the system executes power-down wake-up reset, and the internal medium-speed oscillator works as clock.

When waking up from power-down mode Registers that are not reset the following table.

Power Down Mode	Registers that are not reset
Power down mode 1 Power down mode 2 Power down mode 4	PWC_PWRC0 PWC_PWRC1 PWC_PWRC3 PWC_PDWKE0 PWC_PDWKE1 PWC_PDWKE2 PWC_PDWKES PWC_PDWKF0 PWC_PDWKF1 PWC_PWCMR PWC_XTAL32CS PWC_PVDCR0 PWC_PVDCR1 PWC_PVDFCR PWC_PVDLCR PWC_PVDICR PWC_PVDDSR
Power down mode 3	PWC_PDWKF0 PWC_PDWKF1 RSTF0

■ Pin status after releasing power-down mode

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

- WKTM pin is used for power-down wake-up

The chip has a built-in counter WKTM for power-down wake-up. This counter can choose the internal low-speed oscillator, external low-speed oscillator, and 64Hz internal clock signal as the clock source. The 64Hz internal clock signal uses the external low-speed oscillator as the clock source for RTC. And it is valid when RTC is active. The counter is an up-counting counter. After WKTC0.WKTCE is set, the counter starts counting. When the count value is equal to the set value of WKTCMP[11:0], the counting stops and a wake-up event is generated to wake up the chip from power-down mode. When using WKTM again, you need to reset WKTC0.WKTCE and set it again.

- PTWK power-down mode wake-up event

PTWK is used to wake up the chip in power-down mode. Through software setting, each PTWK event can be selected from 4 pins, and can choose rising or falling trigger edges, with independent flag bits. The structural block diagram of PTWKn is shown in Figure 5-11 PTWKn structure block diagram the figure.

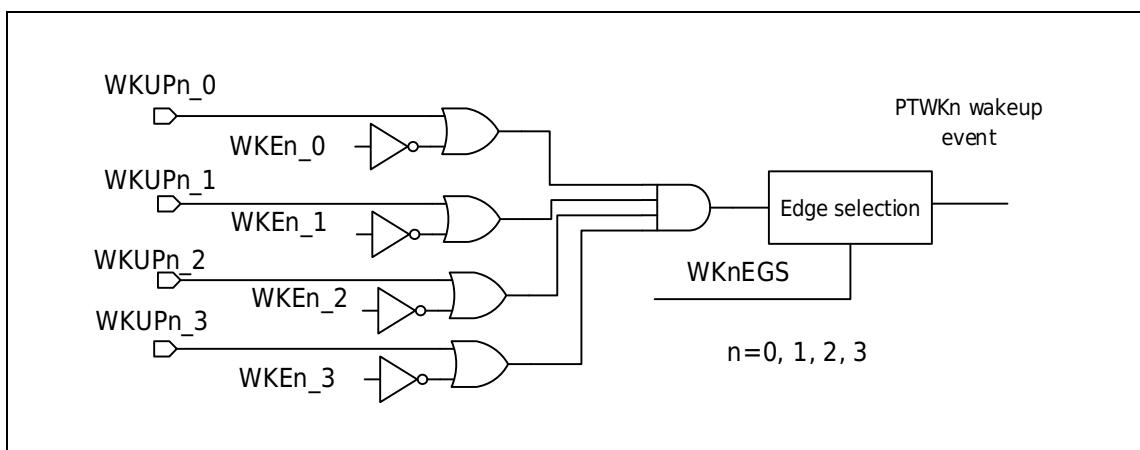


Figure 5-11 PTWKn structure block diagram

- Operation description of VDDR domain power-down mode

The VDDR domain continues to supply power through RLDO after the chip enters power-down mode 1 or power-down mode 2, so RTC /WKTM/Ret-SRAM can continue to operate or maintain data. The VDDR domain is powered by the LDO after power-down wake-up. In power-down mode 3 or power-down mode 4, the VDDR domain stops supplying power.

5.5 Methods of Reducing Power Consumption

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

1. Set the most suitable operation mode
2. Reduce system clock speed
3. Turn off unused timer
4. Set the function clock control register PWC_FCGn (n=0/1/2/3) to turn off unused functions
5. Power off the RAM

5.5.1 Reduce System Clock Speed

In run mode, the speed of system clock (HCLK), external bus clock (EXCLK), peripheral clock PCLK0/ PCLK1/ PCLK2/ PCLK3/ PCLK4 can be reduced by programming the prescaler register. 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)].

5.5.2 Turn-off Unused Timer

The system clock of the chip has six timers:

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

SWDT has an independent dedicated internal low-speed oscillator (SWDTLRC); RTC can choose an external low-speed oscillator or an internal low-speed oscillator as a clock source. In addition, the chip is also configured with a UPLL clock source used by I2S.

For each timer, you can turn it off separately when not in use, reducing system power. Both HRC and PLL are equipped with independent power supply circuits. After the HRC is turned off, the power of the HRC can be cut off by setting the PWC_PWRC1.VHRCSD bit to further reduce power consumption; bit cuts off power to the UPLL and MPLL , further reducing power consumption.

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

5.5.3 Functional Clock Stop

The peripheral module of the chip is provided with the function clock stop function. By using the register corresponding position bit, the module which does not need to be used can be stopped, and the clock of the corresponding module is also stopped to reduce the power consumption. In the module stop state, the register inside the module will remain the state before it stops.

5.5.4 Turn Off Unused RAM

Each RAM module in the chip is configured with a function clock stop bit and a power-down control bit. By setting the module stop bit, it stops providing clocks to RAMs that do not need to be used, 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 It is the corresponding relationship between the RAM module and the power-down control bit. By setting the corresponding RAMPDCn (n=0-8) and PW_PWRC0.RETRAMSD bits of the PWC_RAMPC0 register, the corresponding RAM can be powered down.

Table 5-8 RAM Module and RAM Power-down Control Bits

RAM module	Note	Power down control bit
SRAM1	0x2000_0000~0x2000_FFFF RAM for address space	PWC_RAMPC0.RAMPDC0
SRAM2	0x2001_0000~0x2001_FFFF RAM for address space	PWC_RAMPC0.RAMPDC1
SRAM3	0x2002_0000~0x2002_6FFF RAM for address space	PWC_RAMPC0.RAMPDC2
SRAMH	0x1FFF_8000~0x1FFF_FFFF RAM for address space	PWC_RAMPC0.RAMPDC3
USBFS	RAM for USBFS FIFO	PWC_RAMPC0.RAMPDC4
SDIO0RAM	RAM for SDIO0	PWC_RAMPC0.RAMPDC5
SDIO1RAM	RAM for SDIO1	PWC_RAMPC0.RAMPDC6
CANRAM	RAM for CAN	PWC_RAMPC0.RAMPDC7
CACHERAM	RAM for Cache	PWC_RAMPC0.RAMPDC8
Ret-SRAM	0x200F_0000~0x200F_0FFF RAM for address space	PWC_PWRCC0.RETRAMSD

5.6 Register Protection Function

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

Table 5-9 Register Protection List

Protected register bit	Protected register
PWC_FPRC.FPRCB0	CMU_XTALCFGR, CMU_XTALSTBCR, CMU_XTALCR, CMU_XTALSTDRCR, CMU_XTALSTDSCR, CMU_HRCTRM, CMU_HRCCR, CMU_MRCTRMR, CMU_MRCCR, CMU_PLLCFGR, CMU_PLLCR, CMU_UPLLCFGR, CMU_UPLLCR, CMU_OSCSTBSR, CMU_CKSWR, CMU_SCFGR, CMU_UFSCKCFGR, CMU_TPIUCKCFGR, CMU_MCO1CFG, CMU_MCO2CFG, CMU_XTAL32CR, CMU_XTALC32CFG, CMU_XTAL32NFR, CMU_LRCCR, CMU_LRCTRMR, PWC_XTAL32CS
PWC_FPRC.FPRCB1	PWC_PWRC0, PWC_PWRC1, PWC_PWRC2, PWC_PWRC3, PWC_PDWKE0, PWC_PDWKE1, PWC_PDWKE2, PWC_PDWKES, PWC_PDWKF0, PWC_PDWKF1, PWC_PWCMR, CMU_PERICKSEL, CMU_I2SCKSEL, PWC_MDSWCR, PWC_STPMCR, PWC_RAMPC0, PWC_RAMOPM, RMU_RSTF0
PWC_FPRC.FPRCB3	PWC_PVDCR0, PWC_PVDCR1, PWC_PVDFCR, PWC_PVDLCR, PWC_PVDICR, PWC_PVDDSR

Protected register bit	Protected register
PWC_FCG0PRC.B0	PWC_FCG0

5.7 Register Description

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

BASE ADDR:0x4005_4400

Table 5-10 Register Summary

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	0x01	8	0x00
Power Mode Control Register 2	PWC_PWRC2	0x02	8	0xFF
Power Mode Control Register 3	PWC_PWRC3	0x03	8	0x07
Power-down wake-up enable register 0	PWC_PDWKE0	0x04	8	0x00
Power-down wake-up enable register 1	PWC_PDWKE1	0x05	8	0x00
Power-down wake-up enable register 2	PWC_PDWKE2	0x06	8	0x00
Power-down wake-up event edge select register	PWC_PDWKES	0x07	8	0x00
Power-down wake-up flag register 0	PWC_PDWKF0	0x08	8	0x00
Power-down wake-up flag register 1	PWC_PDWKF1	0x09	8	0x00
Power Monitoring Register	PWC_PWCMR	0x0A	8	0x00
Mode Switch Control Register	PWC_MDSWCR	0x0F	8	0x00
PVD Control Register 0	PWC_PVDCR0	0x12	8	0x00
PVD Control Register 1	PWC_PVDCR1	0x13	8	0x00
PVD filter control register	PWC_PVDFCR	0x14	8	0x11
PVD level control register	PWC_PVDLCR	0x15	8	0x00
XTAL32 Current Control Register	PWC_XTAL32CS	0x2B	8	0x02
BASE ADDR:0x4005_4000				
Register name	Symbol	Offset address	Bit width	Reset value
STOP mode wake-up control register	PWC_STPMCR	0x0C	16	0x4000
RAM Power Control Register 0	PWC_RAMPC0	0x14	32	0x0000_0000
RAM operating condition register	PWC_RAMOPM	0x18	16	0x8043
PVD Interrupt Control Register	PWC_PVIDCR	0xE0	8	0x00
PVD Detection Status Register	PWC_PVDDSR	0xE1	8	0x11
Function protection control register	PWC_FPRC	0x3FE	16	0x0000
BASE ADDR:0x4004_C400				
Register name	Symbol	Offset address	Bit width	Reset value
Wake-up Timer Control Register	PWC_WKTCR	0x00	16	0x0000
BASE ADDR:0x4004_8000				
Register name	Symbol	Offset address	Bit width	Reset value
Function Clock Control 0	PWC_FCG0	0x00	32	0xFFFF_FAEE
Function Clock Control 1	PWC_FCG1	0x04	32	0xFFFF_FFFF

Function Clock Control 2	PWC_FCG2	0x08	32	0xFFFF_FFFF
Function Clock Control 3	PWC_FCG3	0x0C	32	0xFFFF_FFFF
PWC_FCG0 protection control	PWC_FCG0PC	0x10	32	0x0000_0000

5.7.1 Power Mode Control Register 0 (PWC_PWRC0)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
PWDN	-	IORTN[1:0]		RETRAMSD	VVDRSD		PDMDS[1:0]
Bit	Marking	Place name	Function				Read and write
b7	PWDN	Power-down mode control bit	0: Disable power-down mode 1: Enable power-down mode				R/W
b6	Reserved	-	Read as "0", write as "0"				R/W
b5-b4	IORTN[1:0]	IO remains in control in power-down mode	00 : IO hold state in power-down mode , and the hardware releases IO hold state after power-down wake-up 01: IO keeps state in power-down mode, set IORTN[1:0] to 00b, release IO holding status 1x: IO is high impedance in power-down mode and after power-down wake-up				R/W
b3	RETRAMSD	Maintain RAM Power Down Control	0: Ret-SRAM does not power down 1: Ret-SRAM power down				R/W
b2	VVDRSD	VDR LDO control	0: Use RLDOs 1: close RLDO When the RLDO is turned off and the chip enters power-down mode the power to the RTC/wake-up timer/retentive RAM is turned off and thus cannot be used.				R/W
b1-b0	PDMDS[1:0]	Power-down mode selection control	PDMDMS[1:0] 00: power down mode 1 01: power down mode 2 10: power down mode 3 11: Power down mode 4				R/W

5.7.2 Power Mode Control Register 1 (PWC_PWRC1)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
STPDAS[1:0]		-	-	-	-	VHRCSD	VPLLSD
Bit	Marking	Place name	Function				Read and write
b7-b6	STPDAS[1:0]	STOP mode LDO driver selection	00: Drive capacity set when entering STOP mode in super high speed mode and high speed mode 11: Drive capacity set when ultra-low speed mode enters STOP mode 00\01: Setting is prohibited.				R/W
b5-b2	Reserved	-	Read as "0000", write as "0000"				R/W
b1	VHRCSD	HRC power off	0: HRC power enable 1: HRC power off When HRC is not in use, set VHRCSD and turn off the power supply for HRC to further reduce power consumption. After VHRCSD is cleared, you need to wait 25uS before starting the HRC module				R/W
b0	VPLLSD	PLL power off	0: PLL power enable 1: PLL power off After both UPLL and MPLL are turned off and wait for 50us, set VPLLSD and turn off the PLL power supply to further reduce power consumption. After VPLLSD is cleared, you need to wait 25uS before starting the PLL module.				R/W

5.7.3 Power Mode Control Register 2 (PWC_PWRC2)

Reset value: 0xFF

b7	b6	b5	b4	b3	b2	b1	b0
-	-	DVS[1:0]		DDAS[3:0]			
Bit	Marking	Place name	Function				Read and write
b7	Reserved	-	Read as "1", write as "1"				R/W
b6	Reserved	-	Read as "1", write as "1"				R/W
b5-b4	DVS[1:0]	Voltage selection in action mode	00: Select ultra-high speed operation mode voltage 01: Prohibiting 10: Select ultra-low speed action voltage 11: Select high-speed operation mode voltage				R/W
b3-b0	DDAS[3:0]	Power drive selection	1111: Ultra-high-speed operation mode, high-speed operation mode drive capability selection 1110: Prohibiting 1001: Prohibiting 1000: Drive capacity selection for ultra-low speed operation mode Others: Prohibiting				R/W

5.7.4 Power Mode Control Register 3 (PWC_PWRC3)

Reset value: 0x07

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-		PDTs	-	-	-
Bit	Marking	Place name	Function				Read and write
b7-b3	Reserved	-	Reserved bit, write "00000" when writing.				R/W
b2	PDTs	Power-down wake-up time control bit	0: The total capacity of VCAP_1/VCAP_2 is two 0.1uF or one 0.22uF 1: The total capacity of VCAP_1/VCAP_2 is two 0.047uF or one 0.1uF				R/W
b1	Reserved	-	Read as "1", write as "1"				R/W
b0	Reserved	-	Read as "1", write as "1"				R/W

5.7.5 Power-down Wake-up Enable Register 0 (PWC_PDWKE0)

Reset value: 0x00

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

Bit	Marking	Place name	Function	Read and write
b7	WKE13	WKUP1_3 wakeup event enable	0: WKUP1_3 wakeup event is invalid 1: WKUP1_3 wakeup event enable	R/W
b6	WKE12	WKUP1_2 wakeup event enable	0: WKUP1_2 wakeup event is invalid 1: WKUP1_2 wake-up event enable	R/W
b5	WKE11	WKUP1_1 wakeup event enable	0: WKUP1_1 wakeup event is invalid 1: WKUP1_1 wake-up event enable	R/W
b4	WKE10	WKUP1_0 wakeup event enable	0: WKUP1_0 wakeup event is invalid 1: WKUP1_0 wake-up event enable	R/W
b3	WKE03	WKUP0_3 wakeup event enable	0: WKUP0_3 wakeup event is invalid 1: WKUP0_3 wake-up event enable	R/W
b2	WKE02	WKUP0_2 wakeup event enable	0: WKUP0_2 wakeup event is invalid 1: WKUP0_2 wake-up event enable	R/W
b1	WKE01	WKUP0_1 wakeup event enable	0: WKUP0_1 wakeup event is invalid 1: WKUP0_1 wake-up event enable	R/W
b0	WKE00	WKUP0_0 wakeup event enable	0: WKUP0_0 wakeup event is invalid 1: WKUP0_0 wake-up event enable	R/W

5.7.6 Power-down Wake-up Enable Register 1 (PWC_PDWKE1)

Reset value: 0x00

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

Bit	Marking	Place name	Function	Read and write
b7	WKE33	WKUP3_3 wakeup event enable	0: WKUP3_3 wakeup event is invalid 1: WKUP3_3 wake-up event enable	R/W
b6	WKE32	WKUP3_2 wakeup event enable	0: WKUP3_2 wakeup event is invalid 1: WKUP3_2 wake-up event enable	R/W
b5	WKE31	WKUP3_1 wakeup event enable	0: WKUP3_1 wakeup event is invalid 1: WKUP3_1 wake-up event enable	R/W
b4	WKE30	WKUP3_0 wakeup event enable	0: WKUP3_0 wakeup event is invalid 1: WKUP3_0 wake-up event enable	R/W
b3	WKE23	WKUP2_3 wakeup event enable	0: WKUP2_3 wakeup event is invalid 1: WKUP2_3 wake-up event enable	R/W
b2	WKE22	WKUP2_2 wakeup event enable	0: WKUP2_2 wakeup event is invalid 1: WKUP2_2 wake-up event enable	R/W
b1	WKE21	WKUP2_1 wakeup event enable	0: WKUP2_1 wakeup event is invalid 1: WKUP2_1 wake-up event enable	R/W
b0	WKE20	WKUP2_0 wakeup event enable	0: WKUP2_0 wakeup event is invalid 1: WKUP2_0 wake-up event enable	R/W

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

Reset value: 0x00

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

5.7.8 Power-down Wake-up Event Edge Selection Register (PWC_PDWKES)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	NMIEGS	VD2EGS	VD1EGS	WK3EGS	WK2EGS	WK1EGS	WK0EGS
Bit	Marking	Place name			Function	Read and write	
b7	Reserved	-			Read as "0", write as "0"	R/W	
b6	NMIEGS	NMI wakeup event edge selection			0: Falling edge 1: Rising edge	R/W	
b5	VD2EGS	VD2 edge selection			0: VCC<VPVD2 1: VCC> VPVD2	R/W	
b4	VD1EGS	VD1 edge select			0: VCC<VPVD2 1: VCC> VPVD2	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	WK0EGS	PTWK0 edge select			0: Falling edge 1: Rising edge	R/W	

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

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	NMIWKF	PVD2WKF	PVD1WKF	PTWK3F	PTWK2F	PTWK1F	PTWK0F
Bit	Marking	Place name	Function				Read and write
b7	Reserved	-	Read as "1", write as "1"				R/W
b6	NMIWKF	NMI wakeup flag	0: No NMI pin wake-up event occurs 1: NMI pin wake-up event occurs After power-down wake-up, you need to write zero to clear this bit.				R/W
b5	PVD2WKF	PVD2 wakeup flag bit	0: No PVD2 wakeup event occurred 1: A PVD2 wakeup event occurs After power-down wake-up, you need to write zero to clear this bit.				R/W
b4	PVD1WKF	PVD1 wake-up flag bit	0: No PVD1 wake-up event occurred 1: A PVD1 wakeup event occurs After power-down wake-up, you need to write zero to clear this bit.				R/W
b3	PTWK3F	PTWK3 wake-up flag bit	0: No PTWK3 wakeup event occurs 1: A PTWK3 wakeup event occurs After power-down wake-up, you need to write zero to clear this bit.				R/W
b2	PTWK2F	PTWK2 wake-up flag bit	0: No PTWK2 wakeup event occurs 1: A PTWK2 wakeup event occurs After power-down wake-up, you need to write zero to clear this bit.				R/W
b1	PTWK1F	PTWK1 wake-up flag bit	0: No PTWK1 wakeup event occurs 1: A PTWK1 wakeup event occurs After power-down wake-up, you need to write zero to clear this bit.				R/W
b0	PTWK0F	PTWK0 wake-up flag bit	0: No PTWK0 wakeup event occurs 1: A PTWK0 wakeup event occurs After power-down wake-up, you need to write zero to clear this bit.				R/W

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

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
WKTMWKF	-	RTCALMWKF	RTCPRDWKF	-	-	-	
Bit	Marking	Place name	Function			Read and write	
b7	WKTMWKF	WKTM wakeup flag bit	0: No WKTM wakeup event occurs 1: A WKTM wakeup event occurs After power-down wake-up, you need to write zero to clear this bit.			R/W	
b6	Reserved	-	Reserved bit, write "0" when writing.			R/W	
b5	RTCALMWKF	RTC alarm clock wake-up flag bit	0: No RTC alarm wakeup event occurs 1: RTC alarm clock wake-up event occurs After power-down wake-up, you need to write zero to clear this bit.			R/W	
b4	RTCPRDWKF	RTC cycle wake-up flag bit	0: No RTC cycle wake-up event occurs 1: RTC cycle wake-up event occurs After power-down wake-up, you need to write zero to clear this bit.			R/W	
b3	Reserved	-	Reserved bit, write "0" when writing.			R/W	
b2	Reserved	-	Reserved bit, write "0" when writing.			R/W	
b1	Reserved	-	Reserved bit, write "0" when writing.			R/W	
b0	Reserved	-	Reserved bit, write "0" when writing.			R/W	

5.7.11 Power Monitor Control Register (PWC_PWCMR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
ADBUFE	-	-	-	-	-	-	-
Bit	Marking	Place name	Function			Read and write	
b7	ADBUFE	ADBUF enable	When using AD to measure the internal voltage of the chip, you need to set this bit to 1 0: Invalid 1: Valid			R/W	
b6	Reserved	-	Read as "0", write as "0"			R/W	
b5	Reserved	-	Read as "0", write as "0"			R/W	
b4	Reserved	-	Read as "0", write as "0"			R/W	
b2	Reserved	-	Read as "0", write as "0"			R/W	
b1	Reserved	-	Read as "0", write as "0"			R/W	
b0	Reserved	-	Read as "0", write as "0"			R/W	

5.7.12 Mode Switch Control Register (PWC_MDSWCR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
MDSWC[7:0]							
Bit	Marking	Place name	Function				Read and write
b7-b0	MDSWC[7:0]	Mode switching enable	When switching the action mode, after setting PWRC2, you need to set MDSWC[7:0] to 0x10 to take effect. Only 0x10 can be written in, other values are forbidden to be written, and the readout is 0x00.				R/W

5.7.13 Function Clock Control 0 (PWC_FCG0)

Reset value: 0xFFFF_FAEE

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

Bit	Marking	Place name	Function	Read and write
b31	KEY	KEYSCAN function control	0: The KEYSAN function of the keyboard scanning control module is enabled 1: The KEYSAN function of the keyboard scanning control module is invalid	R/W
b30	Reserved	-	Read as "1", write as "1"	R/W
b29	Reserved	-	Read as "1", write as "1"	R/W
b28	Reserved	-	Read as "1", write as "1"	R/W
b27	DCU4	DCU4 Functional Control	0: Digital computing unit DCU3 function enable 1: Digital Computing Unit DCU3 function invalid	R/W
b26	DCU3	DCU3 Functional Control	0: Digital computing unit DCU2 function enable 1: Digital Computing Unit DCU2 function invalid	R/W
b25	DCU2	DCU2 Functional Control	0: Digital computing unit DCU1 function enable 1: Digital Computing Unit DCU1 function invalid	R/W
b24	DCU1	DCU1 Functional Control	0: Digital computing unit DCU0 function enable 1: Digital Computing Unit DCU0 function invalid	R/W
b23	CRC	CRC function control	0: CRC function enabled 1: Invalid CRC function	R/W
b22	TRNG	TRNG function control	0: The true random generator TRNG function in the encryption co-processing module CPM is enabled 1: The true random generator TRNG function in the encryption co-processing module CPM is invalid	R/W
b21	HASH	HASH function control	0: Enable the HASH function of the secure hash algorithm module in the encryption co-processing module CPM 1: The function of the secure hash algorithm module in the encryption co-processing module CPM is invalid	R/W
b20	AES	AES function control	0: Enable the AES function of the encryption and decryption algorithm processor in the encryption co-processing module CPM 1: The AES function of the encryption and decryption algorithm processor in the encryption co-processing module CPM is invalid	R/W
b19	Reserved	-	Read as "1", write as "1"	R/W
b18	Reserved	-	Read as "1", write as "1"	R/W
b17	AOS	Peripheral circuit trigger function control	0: Peripheral circuit trigger function enable 1: The peripheral circuit trigger function is invalid	R/W
b16	FCM	FCM function control	0: The clock frequency measurement module FCM function in the clock controller CMU is enabled 1: The function of the clock frequency measurement module FCM in the clock controller CMU is invalid	R/W
b15	DMA2	DMA2 Functional Control	0: DMA controller DMA2 function enable 1: DMA controller DMA2 function is invalid	R/W
b14	DMA1	DMA1 Functional Control	0: DMA controller DMA1 function enable 1: DMA controller DMA1 function is invalid	R/W
b13	Reserved	-	Read as "1", write as "1"	R/W
b12	Reserved	-	Read as "1", write as "1"	R/W
b11	Reserved	-	Read as "1", write as "1"	R/W
b10	SRAMRET	Ret_SRAM function control	0: Ret_SRAM function enable in built-in SRAM 1: The Ret_SRAM function in the built-in SRAM is invalid	R/W
b9	Reserved	-	Read as "1", write as "1"	R/W

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

5.7.14 Function Clock Control 1 (PWC_FCG1)

Reset value: 0xFFFF_FFFF

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

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "1", write as "1"	R/W
b30	Reserved	-	Read as "1", write as "1"	R/W
b29	Reserved	-	Read as "1", write as "1"	R/W
b28	Reserved	-	Read as "1", write as "1"	R/W
b27	USART4	USART4 Functional Control	0: Universal synchronous asynchronous transceiver USART4 function enable 1: Universal synchronous asynchronous transceiver USART4 function invalid	R/W
b26	USART3	USART3 Functional Control	0: Universal synchronous asynchronous transceiver USART3 function enable 1: Universal synchronous asynchronous transceiver USART3 function is invalid	R/W
b25	USART2	USART2 Functional Control	0: Universal synchronous asynchronous transceiver USART2 function enable 1: Universal synchronous asynchronous transceiver USART2 function is invalid	R/W
b24	USART1	USART1 Functional Control	0: Universal synchronous asynchronous transceiver USART1 function enable 1: Universal synchronous asynchronous transceiver USART1 function is invalid	R/W
b23	Reserved	-	Read as "1", write as "1"	R/W
b22	Reserved	-	Read as "1", write as "1"	R/W
b21	Reserved	-	Read as "1", write as "1"	R/W
b20	Reserved	-	Read as "1", write as "1"	R/W
b19	SPI4	SPI4 Functional Control	0: Serial peripheral interface SPI4 function enable 1: Serial Peripheral Interface SPI4 function invalid	R/W
b18	SPI3	SPI3 Functional Control	0: Serial peripheral interface SPI3 function enable 1: Serial Peripheral Interface SPI3 function invalid	R/W
b17	SPI2	SPI2 Functional Control	0: Serial peripheral interface SPI2 function enable 1: Serial Peripheral Interface SPI2 function invalid	R/W
b16	SPI1	SPI1 Functional Control	0: Serial peripheral interface SPI1 function enable 1: Serial Peripheral Interface SPI1 function invalid	R/W
b15	I2S4	I2S4 Functional Control	0: IC built-in audio bus module I2S4 function enable 1: Integrated circuit built-in audio bus module I2S4 function is invalid	R/W

b14	I2S3	I2S3 Functional Control	0: Integrated circuit built-in audio bus module I2S3 function enable 1: Integrated circuit built-in audio bus module I2S3 function invalid	R/W
b13	I2S2	I2S2 Functional Control	0: Integrated circuit built-in audio bus module I2S2 function enable 1: Integrated circuit built-in audio bus module I2S2 function invalid	R/W
b12	I2S1	I2S1 Functional Control	0: Integrated circuit built-in audio bus module I2S1 function enable 1: Integrated circuit built-in audio bus module I2S1 function invalid	R/W
b11	SDIOC2	SDIOC2 Functional Control	0: SDIO controller SDIOC2 function enable 1: SDIO controller SDIOC2 function invalid	R/W
b10	SDIOC1	SDIOC1 Functional Control	0: SDIO controller SDIOC1 function enable 1: SDIO controller SDIOC1 function invalid	R/W
b9	Reserved	-	Read as "1", write as "1"	R/W
b8	USBFS	USBFS function control	0: USB2.0 full speed module USBFS function enable 1: The USBFS function of the USB2.0 full-speed module is invalid	R/W
b7	Reserved	-	Read as "1", write as "1"	R/W
b6	I2C3	I2C3 Functional Control	0: Integrated circuit bus I2C3 function enable 1: IC bus I2C3 function invalid	R/W
b5	I2C2	I2C2 Functional Control	0: Integrated circuit bus I2C2 function enable 1: IC bus I2C2 function invalid	R/W
b4	I2C1	I2C1 Functional Control	0: Integrated circuit bus I2C1 function enable 1: IC bus I2C1 function invalid	R/W
b3	QSPI	QSPI function control	0: Four-wire serial peripheral interface QSPI function enable 1: Four-wire serial peripheral interface QSPI function is invalid	R/W
b2	Reserved	-	Read as "1", write as "1"	R/W
b1	Reserved	-	Read as "1", write as "1"	R/W
b0	CAN	CAN function control	0: Controller area network CAN function enable 1: The CAN function of the controller area network is invalid	R/W

5.7.15 Function Clock Control 2 (PWC_FCG2)

Reset value: 0xFFFF_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	TIMER 6_3	TIMER 6_2	TIMER 6_1
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EMB	-	-	-	-	TIMER 4_3	TIMER 4_2	TIMER 4_1	TIMER A_6	TIMER A_5	TIMER A_4	TIMER A_3	TIMER A_2	TIMER A_1	TIMER 0_2	TIMER 0_1

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "1", write as "1"	R/W
b30	Reserved	-	Read as "1", write as "1"	R/W
b29	Reserved	-	Read as "1", write as "1"	R/W
b28	Reserved	-	Read as "1", write as "1"	R/W
b27	Reserved	-	Read as "1", write as "1"	R/W
b26	Reserved	-	Read as "1", write as "1"	R/W
b25	Reserved	-	Read as "1", write as "1"	R/W
b24	Reserved	-	Read as "1", write as "1"	R/W
b23	Reserved	-	Read as "1", write as "1"	R/W
b22	Reserved	-	Read as "1", write as "1"	R/W
b21	Reserved	-	Read as "1", write as "1"	R/W
b20	Reserved	-	Read as "1", write as "1"	R/W
b19	Reserved	-	Read as "1", write as "1"	R/W
b18	TIMER6_3	TIMER6_3 Functional Control	0: TIMER6_3 function enabled 1: Invalid TIMER6_3 function	R/W
b17	TIMER6_2	TIMER6_2 Functional Control	0: TIMER6_2 function enabled 1: Invalid TIMER6_2 function	R/W
b16	TIMER6_1	TIMER6_1 Functional Control	0: TIMER6_1 function enabled 1: Invalid TIMER6_1 function	R/W
b15	EMB	EMB Functional Control	0: Enable the EMB function of the emergency brake module 1: Emergency brake module EMB function invalid	R/W
b14	Reserved	-	Read as "1", write as "1"	R/W
b13	Reserved	-	Read as "1", write as "1"	R/W
b12	Reserved	-	Read as "1", write as "1"	R/W
b11	Reserved	-	Read as "1", write as "1"	R/W
b10	TIMER4_3	TIMER4_3 Functional Control	0: TIMER4_3 function enabled 1: Invalid TIMER4_3 function	R/W
b9	TIMER4_2	TIMER4_2 Functional Control	0: TIMER4_2 function enabled 1: Invalid TIMER4_2 function	R/W
b8	TIMER4_1	TIMER4_1 Functional Control	0: TIMER4_1 function enabled 1: Invalid TIMER4_1 function	R/W
b7	TIMERA_6	TIMERA_6 Functional Control	0: TIMERA_6 function enabled 1: Invalid TIMERA_6 function	R/W
b6	TIMERA_5	TIMERA_5 Functional Control	0: TIMERA_5 function enabled 1: Invalid TIMERA_5 function	R/W
b5	TIMERA_4	TIMERA_4 Functional Control	0: TIMERA_4 function enabled 1: Invalid TIMERA_4 function	R/W
b4	TIMERA_3	TIMERA_3 Functional Control	0: TIMERA_3 function enabled 1: Invalid TIMERA_3 function	R/W
b3	TIMERA_2	TIMERA_2 Functional Control	0: TIMERA_2 function enabled 1: Invalid TIMERA_2 function	R/W

b2	TIMERA_1	TIMERA_1 Functional Control	0: TIMERA_1 function enabled 1: Invalid TIMERA_1 function	R/W
b1	TIMER0_2	TIMER0_2 Functional Control	0: TIMER0_2 function enabled 1: Invalid TIMER0_2 function	R/W
b0	TIMER0_1	TIMER0_1 Functional Control	0: TIMER0_1 function enabled 1: Invalid TIMER0_1 function	R/W

5.7.16 Function Clock Control 3 (PWC_FCG3)

Reset value: 0xFFFF_FFFF

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

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "1", write as "1"	R/W
b30	Reserved	-	Read as "1", write as "1"	R/W
b29	Reserved	-	Read as "1", write as "1"	R/W
b28	Reserved	-	Read as "1", write as "1"	R/W
b27	Reserved	-	Read as "1", write as "1"	R/W
b26	Reserved	-	Read as "1", write as "1"	R/W
b25	Reserved	-	Read as "1", write as "1"	R/W
b24	Reserved	-	Read as "1", write as "1"	R/W
b23	Reserved	-	Read as "1", write as "1"	R/W
b22	Reserved	-	Read as "1", write as "1"	R/W
b21	Reserved	-	Read as "1", write as "1"	R/W
b20	Reserved	-	Read as "1", write as "1"	R/W
b19	Reserved	-	Read as "1", write as "1"	R/W
b18	Reserved	-	Read as "1", write as "1"	R/W
b17	Reserved	-	Read as "1", write as "1"	R/W
b16	Reserved	-	Read as "1", write as "1"	R/W
b15	Reserved	-	Read as "1", write as "1"	R/W
b14	Reserved	-	Read as "1", write as "1"	R/W
b13	Reserved	-	Read as "1", write as "1"	R/W
b12	OTS	OTS function control	0: The temperature sensor OTS function is valid 1: The temperature sensor OTS function is invalid	R/W
b11	Reserved	-	Read as "1", write as "1"	R/W
b10	Reserved	-	Read as "1", write as "1"	R/W
b9	Reserved	-	Read as "1", write as "1"	R/W
b8	CMP	CMP function control	0: Voltage comparator CMP function enable 1: The voltage comparator CMP function is invalid	R/W
b7	Reserved	-	Read as "1", write as "1"	R/W
b6	Reserved	-	Read as "1", write as "1"	R/W

b5	Reserved	-	Read as "1", write as "1"	R/W
b4	Reserved	-	Read as "1", write as "1"	R/W
b3	Reserved	-	Read as "1", write as "1"	R/W
b2	Reserved	-	Read as "1", write as "1"	R/W
b1	ADC2	ADC2 Functional Control	0: ADC2 function of the analog-to-digital conversion module is enabled 1: The ADC2 function of the analog-to-digital conversion module is invalid	R/W
b0	ADC1	ADC1 Functional Control	0: ADC1 function of the analog-to-digital conversion module is enabled 1: The ADC1 function of the analog-to-digital conversion module is invalid	R/W

5.7.17 PWC_FCG0 Protection Control (PWC_FCG0PC)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	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	Protection bit	PWC_FCG0 write enable control bit 0: PWC_FCG0 write invalid 1: PWC_FCG0 write enable	R/W

5.7.18 Function Protection Control Register (PWC_FPRC)

Reset value: 0x0000

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

Bit	Marking	Place name	Function	Read and write
b15~b8	PWC_FPRCWE	PWC_FPRC register write enable	Write to 0xA5h to update the PWC_FPRC value, otherwise it is invalid for the lower 8-bit write value. Read at 0x00.	R/W
b7	Reserved	-	Reserved bit, write "0" when writing.	R/W
b6	Reserved	-	Reserved bit, write "0" when writing.	R/W
b5	Reserved	-	Reserved bit, write "0" when writing.	R/W
b4	Reserved	-	Reserved bit, write "0" when writing.	R/W
b3	FPRCB3	FPRC bit 3	Protection Register Bits, Protection Object Reference Table 5-9 0: Write protection 1: Write enable	R/W
b2	Reserved	-	Reserved bit, write "0" when writing.	R/W
b1	FPRCB1	FPRC bit 1	Protection Register Bits, Protection Object Reference Table 5-9 0: Write protection 1: Write enable	R/W
b0	FPRCB0	FPRC bit 0	Protection Register Bits, Protection Object Reference Table 5-9 0: Write protection 1: Write enable	R/W

5.7.19 STOP Mode Control Register (PWC_STPMCR)

Reset value: 0x4000

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

Bit	Marking	Place name	Function	Read and write
b15	STOP	STOP mode selection bit	0: STOP mode is invalid 1: STOP mode is valid	R/W
b14	Reserved	-	Read as "1", write as "1"	R/W
b13-b2	Reserved	-	Read as "0", write as "0"	R/W
b1	CKSMRC	Clock switch to MRC option	0: Maintain the system clock and frequency division before entering STOP mode 1: The system clock switches to MRC and the SCKCFGR register is initialized when waking up from STOP mode	R/W
b0	FLNWT	FLASH stable waiting control	0: Wait for FLASH to stabilize when waking up from STOP mode 1: Do not wait for FLASH to stabilize when waking up from STOP mode	R/W-

5.7.20 RAM Power Control Register 0 (PWC_RAMPC0)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b32-b23	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-b9	Reserved	-	Read as "0", write as "0"	R/W
b8	RAMPDC8	RAM power down control bit 8	0: cache uses RAM to act 1: The cache is powered off with RAM	R/W
b7	RAMPDC7	RAM power down control bit 7	0: can operate with RAM 1: Can use RAM to power down	R/W
b6	RAMPDC6	RAM power down control bit 6	0: sdio1 operates with RAM 1: sdio1 power down with RAM	R/W
b5	RAMPDC5	RAM power down control bit 5	0: sdio0 operates with RAM 1: sdio0 uses RAM to power down	R/W
b4	RAMPDC4	RAM power down control bit 4	0: usbfs operates with RAM 1: usbfs power down with RAM	R/W
b3	RAMPDC3	RAM power down control bit 3	0: 0x1FFF_8000~0x1FFF_FFFF space RAM (SRAMH) action 1: 0x1FFF_8000~0x1FFF_FFFF space RAM (SRAMH) power down	R/W
b2	RAMPDC2	RAM power down control bit 2	0: 0x2002_0000~0x2002_6FFF space RAM (SRAM3) action 1: 0x2002_0000~0x2002_6FFF space RAM (SRAM3) power down	R/W
b1	RAMPDC1	RAM power down control bit 1	0: 0x2001_0000~0x2001_FFFF space RAM (SRAM2) action 1: 0x2001_0000~0x2001_FFFF space RAM (SRAM2) power down	R/W
b0	RAMPDC0	RAM power down control bit 0	0: 0x2000_0000~0x2000_FFFF space RAM (SRAM1) action 1: 0x2000_0000~0x2000_FFFF space RAM (SRAM1) power down	R/W

5.7.21 RAM Operating Condition Register (PWC_RAMOPM)

Reset value: 0x8043

15	14	13	12	11	10	9	8
PWC_RAMOPM[15:8]							
7	6	5	4	3	2	1	0
PWC_RAMOPM[7:0]							
Bit	Marking	Place name	Function			Read and write	
b15-b0	PWC_RAMOPM[15:0]	The RAM action mode is selected as	When the chip works in ultra-high speed/high speed mode, PWC_RAMOPM is set to 0x8043. When the chip works in ultra-low speed mode, PWC_RAMOPM is set to 0x9062			R/W	

5.7.22 XTAL32 is Controlled by a Current Source (PWC_XTAL32CS)

Reset value: 0x02

b7	b6	b5	b4	b3	b2	b1	b0
CSDIS			-		-	-	
Bit	Marking	Place name	Function			Read and write	
7	CSDIS	Current source invalid control	0: current source is active 1: The current source is invalid When XTAL32/RTC/WTKM/Ret-SRAM are not needed, CSDIS can be set to reduce power consumption.			R/W	
b6-b2	Reserved	-	Read as "0", write as "0"			R/W	
b1	Reserved	-	Read as "1", write as "1"			R/W	
b0	Reserved	-	Read as "0", write as "0"			R/W	

5.7.23 Wake-up Timer Control Register (PWC_WKTCR)

Reset value: 0x0000

b15	b14-b13	b12	b11	-	b0
WKTCE	WKCKS[1:0]	WKOVF			WKTMCMP[11:0]

Bit	Marking	Place name	Function	Read and write
b15	WKTCE	WKTM enable	0: WKTM stop 1: WKTM count	R/W
b14-b13	WKCKS[1:0]	WKTM clock selection	00: 64Hz clock 01: XTAL32 10: LRC 11: reserved	R/W
b12	WKOVF	Timer comparison result flag	0: The counter is inconsistent with the WKTMCMP value 1: The counter is consistent with the WKTMCMP value	R/W
b11-b0	WKTMCMP[11:0]	WKTM compare bit	Comparison value of WKTM counter	R/W

5.7.24 PVD Control Register 0 (PWC_PVDCR0)

Reset value: 0x00

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

Bit	Marking	Place name	Function	Read and write
b7	Reserved	-	Read as "0", write as "0"	R/W
b6	PVD2EN	Voltage detection 2 permissible	0: Voltage detection 2 circuit is invalid 1: The voltage detection 2 circuit is valid	R/W
b5	PVD1EN	Voltage detection 1 permissible	0: Voltage detection 1 circuit is invalid 1: The voltage detection 1 circuit is valid	R/W
b4	Reserved	-	Read as "0", write as "0"	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	Reserved	-	Read as "0", write as "0"	R/W
b0	EXVCCINEN	External VCC voltage input enable	0: External VCC voltage input is invalid 1: External VCC voltage input is valid	R/W

5.7.25 PVD Control Register 1 (PWC_PVDCR1)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	PVD2CMPOE	PVD2IRS	PVD2IRE	-	PVD1CMPOE	PVD1IRS	PVD1IRE
Bit	Marking	Place name	Function				Read and write
b7	Reserved	-	Read as "0", write as "0"				R/W
b6	PVD2CMPOE	PVD2 comparison result output enable	0: Disable output of comparison result of comparator 2 1: Allow to output the comparison result of comparator 2				R/W
b5	PVD2IRS	PVD2 interrupt reset selection	0: PVD2 interrupt is generated when VCC or external input voltage drops through VPVD2 1: VCC or external input voltage drops and generates PVD reset through VPVD2 Note: When the PVD1IRS bit is "1" or the PVD2IRS bit is "1", you cannot enter the power-down mode. To enter the power-down mode, you must set the PVD1IRS bit to "0" and the PVD2IRS bit to "0".				R/W
b4	PVD2IRE	PVD2 interrupt reset enable	0: Prohibited 1: Permissible Note: Please write "1" to the PVD2IRE bit when the PVD2EN bit is "1" and the PVD2CMPOE bit is "1"				R/W
b3	Reserved	-	Read as "0", write as "0"				R/W
b2	PVD1CMPOE	PVD1 comparison result output enable	0: Disable output of comparison result of comparator 1 1: Allow to output the comparison result of comparator 1				R/W
b1	PVD1IRS	PVD1 interrupt reset selection	0: PVD1 interrupt is generated when VCC falls through VPVD1 1: VCC falls and passes through VPVD1 to generate PVD1 reset Note 1: When the PVD1IRS bit is "1" or the PVD2IRS bit is "1", you cannot enter the power-down mode. To enter the power-down mode, you must set the PVD1IRS bit to "0" and the PVD2IRS bit to "0".				R/W
b0	PVD1IRE	PVD1 interrupt reset enable	0: Prohibited 1: Permissible Note: Please write "1" to the PVD1IRE bit when the PVD1EN bit is "1" and the PVD1CMPOE bit is "1"				R/W

5.7.26 PVD Filter Control Register (PWC_PVDFCR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	PVD2NFCKS[1:0]		PVD2NFDIS	-	PVD1NFCKS[1:0]		PVD1NFDIS
<hr/>							
Bit	Marking	Place name	Function			Read and write	
b7	Reserved	-	Read as "0", write as "0"			R/W	
b6~b5	PVD2NFCKS	PVD2 digital filter sampling clock selection	00: 0.25 LRC cycle 01: 0.5 LRC cycle 10: 1 frequency division of LRC 11: 2-Division Frequency of LRC Note: This bit can only be rewritten when the PVD2NFDIS bit is "1"			R/W	
b4	PVD2NFDIS	PVD2 Digital Filter Shield	0: Digital filter is valid 1: Invalid digital filter			R/W	
b3	Reserved	-	Read as "0", write as "0"			R/W	
b2~b1	PVD1NFCKS	PVD1 digital filter sampling clock selection	00: 0.25 LRC cycle 01: 0.5 LRC cycle 10: 1 frequency division of LRC 11: 2-Division Frequency of LRC Note: This bit can only be rewritten when the PVD1NFDIS bit is "1"			R/W	
b0	PVD1NFDIS	PVD1 Digital Filter Shield	0: Digital filter is valid 1: Invalid digital filter			R/W	

5.7.27 PVD Level Control Register (PWC_PVDLCR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-		PVD2LVL[2:0]		-		PVD1LVL[2:0]	
Bit	Marking	Place name	Function				Read and write
b7	Reserved	-	Read as "0", write as "0"				R/W
			000: 2.1V 001: 2.3V 010: 2.5V 011: 2.6V 100: 2.7V 101: 2.8V 110: 2.9V				
b6~b4	PVD2LVL	PVD2 threshold voltage selection	111: 1.1V (only valid when PWC_PVDCR0.EXVCCINEN=1, please do not set this value in other cases) Note: The above-mentioned thresholds are the thresholds when the chip works in high-speed mode, ultra-low-speed mode, and stop mode. For the thresholds when the chip works in ultra-high-speed mode, please refer to the electrical characteristics.				R/W
b3	Reserved	-	Read as "0", write as "0"				R/W
			000: 2.0V 001: 2.1V 010: 2.3V 011: 2.5V 100: 2.6V 101: 2.7V 110: 2.8V 111: 2.9V				
b2~b0	PVD1LVL	PVD1 threshold voltage selection	Note: The above-mentioned thresholds are the thresholds when the chip works in high-speed mode, ultra-low-speed mode, and stop mode. For the thresholds when the chip works in ultra-high-speed mode, please refer to the electrical characteristics.				R/W

5.7.28 PVD Interrupt Control Register (PWC_PVDICR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
Bit	Marking	Place name	Function				Read and write
b7	Reserved	-	Read as "0", write as "0"				R/W
b6~b5	Reserved	-	Read as "0", write as "0"				R/W
b4	PVD2NMIS	PVD2 interrupt type selection	0: PVD2 interrupt as non-maskable interrupt 1: PVD2 interrupt as a maskable interrupt				R/W
b3	Reserved	-	Read as "0", write as "0"				R/W
b2~b1	Reserved	-	Read as "0", write as "0"				R/W
b0	PVD1NMIS	PVD1 interrupt type selection	0: PVD1 interrupt as non-maskable interrupt 1: PVD1 interrupt as a maskable interrupt				R/W

5.7.29 PVD Detection Status Register (PWC_PVDDSR)

Reset value: 0x00

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

6 Initialization Configuration (ICG)

6.1 Introduction

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

The initialization configuration register address list is as follows:

ICG_BASE_ADDR: 0x0000_0400

Table 6-1 Register Summary

Register name	Symbol	Offset address	Bit width	Reset value
Initialize Configuration Register 0	ICG0	0x000	32	Indefinite
Initialize Configuration Register 1	ICG1	0x004	32	Indefinite
Initialize Configuration Register 2	ICG2	0x008	32	Indefinite
Initialize Configuration Register 3	ICG3	0x00C	32	Indefinite
Initialize Configuration Register 4	ICG4	0x010	32	Indefinite
Initialize Configuration Register 5	ICG5	0x014	32	Indefinite
Initialize Configuration Register 6	ICG6	0x018	32	Indefinite
Initialize Configuration Register 7	ICG7	0x01C	32	Indefinite

6.2 Register Description

6.2.1 Initialization Configuration Register0 (ICG0)

Reset value:Indefinite

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
-	-	-	SWDT SLPO FF		SWDTWDPT[3:0]			SWDTCKS[3:0]			SWDTPERI[1: 0]		SWDTI TS		SWDTA UTS

Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	Functional reservation bit	R
b28	WDTSLPOFF	WDT counting stops 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]	Refresh Allowed Area Count%	WDT count value refresh allowed range 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 counter clock	0010: PCLK3/4 0110: PCLK3/64 0111: PCLK3/128 1000: PCLK3/256 1001: PCLK3/512 1010: PCLK3/1024 1011: PCLK3/2048 1101: PCLK3/8192 Other values: Reserved	R
b19~b18	WDTPERI[1:0]	WDT count overflow cycle	00: 256 cycle 01: 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 auto start	0: WDT starts automatically after reset (hardware starts) 1: WDT stops after reset	R
b15~b13	Reserved	-	Functional reservation bit	R
b12	SWDTSLPOFF	SWDT count stops in Sleep, Stop mode	0: SWDT counts do not stop in sleep, stop mode 1: SWDT count stopped in sleep, stop mode	R
b11~b8	SWDTWDPT[3:0]	Refresh Allowed Area Count%	SWDT Count Refresh Allowed Interval 0000: 0%~100% 0001: 0%~25% 0010: 25%~50% 0011: 0%~50% 0100: 50%~75% 0101: 0%~25%, 50%~75% 0110: 25%~75% 0111: 0%~75% 1000: 75%~100% 1001: 0%~25%, 75%~100% 1010: 25%~50%, 75%~100% 1011: 0%~50%, 75%~100%	R

			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: Reserved	R
b3~b2	SWDTPERI[1:0]	SWDT count overflow cycle	00: 256 cycle 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 automatically after reset (hardware starts) 1: SWDT stops after reset	R

6.2.2 Initialization Configuration register1 (ICG1)

Reset value: Indefinite

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
NMIIC GENA	NFEN	NMIE NR	NMIT RG	SMPCLK[1:0]	-	-	-	-	-	-	-	-	BOR DIS	BOR_lev[1:0]	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	HRCS TOP	-	-	-	-	-	-	HRCF REQS EL

Bit	Marking	Place name	Function	Read and write
b31	NMIICGENA	NMI Pin ICG Setting Enable	0: NMI Pin ICG Setting Enable 1: NMI Pin ICG Setting Forbidding	R
b30	NFEN	NMI digital filter enable	0: Disable digital filter function 1: Licensed digital filter function	R
b29	NMIENR	NMI Pinterrrupt Selection	0: Disable NMI Pinterrrupt 1: Licensed NMI Pinterrrupt	R
b28	NMITRG	NMI Pin Edge Trigger	0: Falling edge 1: Rising edge	R
b27~b26	SMPCLK[1:0]	Filtering sampling clock selection	0 0: PCLK3 0 1: PCLK3/8 1 0: PCLK3/32 1 1:: PCLK3/64	R
b25~b19	Reserved	-	Functional reservation bit	R
b18	BORDIS	BOR action selection	0: Allow BOR action 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 Bit	0: HRC oscillation 1: HRC stop	R
b7~b1	Reserved	-	Read as "1", write as "1"	R
b0	HRCFREQSEL	HRC frequency selection	0: 20MHz 1: 16MHz	R

6.2.3 Initialization Configuration Registers (ICGn) n = 2 ~ 3

Reset value: Indefinite

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

Bit	Marking	Place name	Function	Read and write
b31~b0	ICGn[31:0]	-	Functional reservation bit The user needs to set all 1 to make sure the chip works properly	R

6.2.4 Initialize Configuration Register 4 (ICG4)

Reset value: Indefinite

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

Bit	Marking	Place name	Function	Read and write
b31~b0	ICG4[31:0]	Initial configuration 4	Chip data security protection level 1 enable configuration. When the data is 0xAF180402, data security protection level 1 is enabled. When the data is other values, the data security protection level 1 is invalid.	R

6.2.5 Initialize Configuration Register 5 (ICG5)

Reset value: Indefinite

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

Bit	Marking	Place name	Function	Read and write
b31~b0	ICG5[31:0]	Initial configuration 5	Chip data security protection level 2 enable configuration. When the data is 0xA85173AE, data security protection level 2 is enabled. When the data is other values, data security protection level 2 is invalid.	R

6.2.6 Initialization Configuration Registers (ICGn) n = 6 ~ 7

Reset value: Indefinite

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ICGn[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ICGn[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b0	ICGn[31:0]	-	Functional reservation bit The user needs to set all 1 to make sure the chip works properly	R											

7 Embedded FLASH (EFM)

7.1 Introduction

The FLASH interface accesses the FLASH through the FLASH ICODE, DCODE and MCODE buses. This interface can perform programming, sector erase and full erase operations on FLASH; accelerate code execution through instruction prefetch and cache mechanism.

7.2 Main Characteristics

- Maximum 512KByte FLASH space
- ICODE bus 16Byte prefetch value
- Shared 64 buffers (1KByte) on ICODE and DCODE buses
- Provide 960Byte one-time programming area (OTP)
- Supports low-power read operations
- Support boot swap function
- Support Data Security Protection

7.3 Embedded FLASH

FLASH has the following main features:

- Capacity up to 512 KBytes (of which 32 Bytes are reserved for functions)
Divided into 64 sectors, each sector is 8KBytes.
- The OTP (One Time Program) area has a total of 1020Bytes, which is divided into a data area of 960Bytes and a latch area of 60Bytes.
- 128-bit data read.
- The programming unit is 4Bytes, and the erasing unit is 8KBytes.

In 512KB products, the FLASH address structure is shown in the figure below.

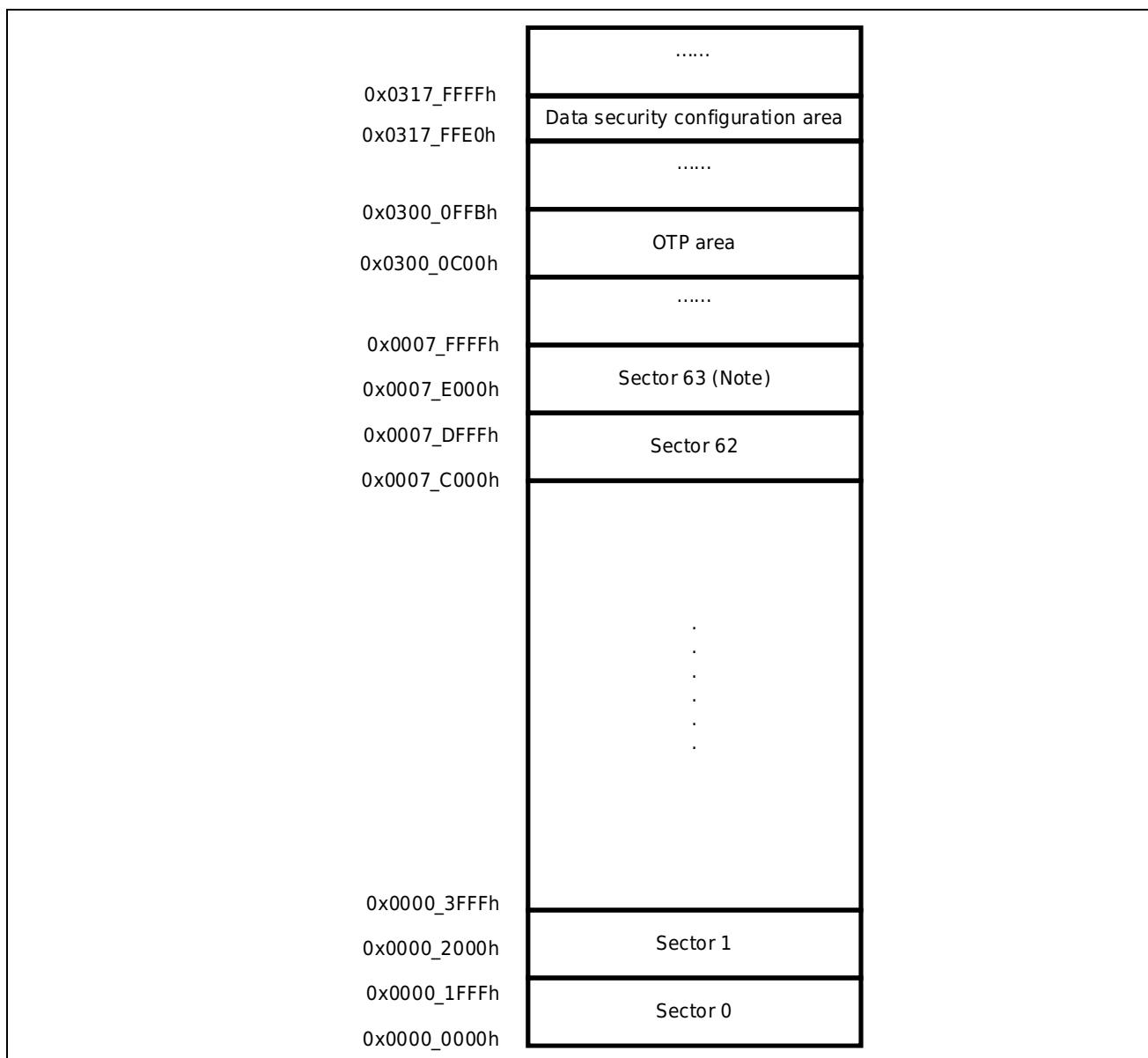


Figure 7-1 FLASH Address Structure (512KB product)

Note:

- The address 0x0007_FFE0~0x0007_FFFF in sector 63 is a function reserved address with a total of 32Bytes; programming, sector erasing, and full erasing these 32Bytes addresses, the FLASH data will not change, read these addresses , and the read data is all 1.

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

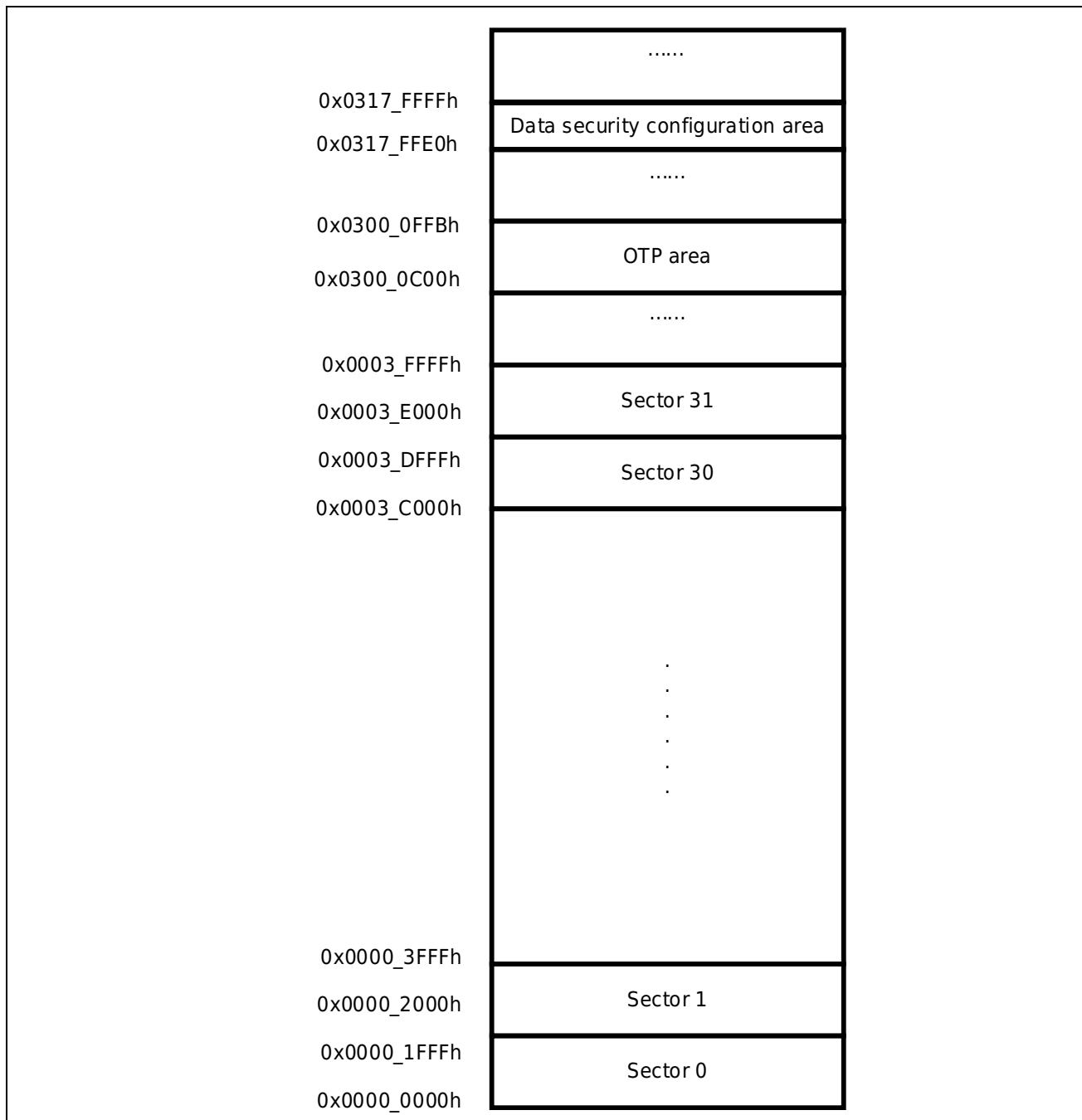


Figure 7-2 FLASH Address Structure (256KB product)

7.4 Read Interface

7.4.1 Relationship between CPU Clock and FLASH Read Time

To read FLASH data correctly, users need to correctly set the number of waiting periods (FLWT[3:0]) in FLASH read mode register (EFM_FRMC) according to the CPU action frequency.

After the system is reset, the CPU clock source is MRC (8MHz), and the FLASH read wait period is 0. It is recommended that the user modify the CPU main frequency and FLASH read wait period bits as follows. Please refer to the number of waiting cyclesTable 7-1.

CPU Frequency Increase Step:

1. Write the new read wait period setting value (FLWT[3:0]) into the register EFM_FRMC.
2. Read register EFM_FRMC and check whether the new waiting period is set successfully.
3. Increase the CPU clock frequency by setting the system clock source switching register CMU_CKSWR (CKSW[2:0]) or the system clock configuration register CMU_SCFG (HCLKS[2:0]).
4. Read the register CMU_CKSWR or CMU_SCFG to check whether the new setting is successful.

CPU Frequency Reduction Step:

1. Reduce the CPU clock frequency by setting the system clock source switching register CMU_CKSWR (CKSW[2:0]) or the system clock configuration register CMU_SCFG (HCLKS[2:0]).
2. Read the register CMU_CKSWR or CMU_SCFG to check whether the new setting is successful.
3. Write the new read wait period setting value (FLWT[3:0]) into the register EFM_FRMC.
4. Read register EFM_FRMC and check whether the new waiting period is set successfully.

7.4.2 FLASH Low Power Read

When the CPU clock frequency is lower than 2MHz, the user can set the register bit EFM_FRMC.SLPMD to enter the low-power read mode to reduce the current of the chip. After entering the low-power read mode, the programming and erasing operations of the FLASH will be ignored. In this mode, the FLASH is erased and written (programming and erasing, the same below), and the status bit EFM_FSR.COLERR is set .

Enter the ultra-low Power Read Step:

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

Exit the Ultra-low Power Read Steps:

1. Register bit EFM_FRMC.SLPMD is cleared.
2. According to Table 7-1the new read insertion wait period setting value (FLWT[3:0]), the

recommended value of the normal read mode is written into the register EFM_FRMC.

Table 7-1 Comparison Table of CPU Clock Frequency and FLASH Read Wait Period

CPU clock frequency (HCLK)	FRMC register bit FLWT[3:0] setting	
	Normal read mode (SLPMD=0)	Ultra Low Power Read Mode (SLPMD=1)
168MHz < F _{HCLK} ≤ 200MHz	FLWT[3:0]=4'b0101 Insert 5 waiting period	not support
132MHz < F _{HCLK} ≤ 168MHz	FLWT[3:0]=4'b0100 Insert 4 waiting period	not support
99MHz < F _{HCLK} ≤ 132MHz	FLWT[3:0]=4'b0011 Insert 3 waiting period	not support
66MHz < F _{HCLK} ≤ 99MHz	FLWT[3:0]=4'b0010 Insert 2 waiting period	not support
33MHz < F _{HCLK} ≤ 66MHz	FLWT[3:0]=4'b0001 Insert 1 waiting period	not support
2MHz < F _{HCLK} ≤ 33MHz	FLWT[3:0]=4'b0000 No wait read	not support
1.876MHz < F _{HCLK} ≤ 2MHz	Ditto	FLWT[3:0]=4'b1111 Insert 15 waiting period
1.752MHz < F _{HCLK} ≤ 1.876MHz	Ditto	FLWT[3:0]=4'b1110 Insert 14 waiting period
1.628MHz < F _{HCLK} ≤ 1.752MHz	Ditto	FLWT[3:0]=4'b1101 Insert 13 waiting period
1.504MHz < F _{HCLK} ≤ 1.628MHz	Ditto	FLWT[3:0]=4'b1100 Insert 12 waiting period
1.38MHz < F _{HCLK} ≤ 1.504MHz	Ditto	FLWT[3:0]=4'b1011 Insert 11 waiting period
1.256MHz < F _{HCLK} ≤ 1.38MHz	Ditto	FLWT[3:0]=4'b1010 Insert 10 waiting period
1.132MHz < F _{HCLK} ≤ 1.256MHz	Ditto	FLWT[3:0]=4'b1001 Insert 9 waiting period
1.008MHz < F _{HCLK} ≤ 1.256MHz	Ditto	FLWT[3:0]=4'b1000 Insert 8 waiting period
884KHz < F _{HCLK} ≤ 1.008MHz	Ditto	FLWT[3:0]=4'b0111 Insert 7 waiting period
760KHz < F _{HCLK} ≤ 884KHz	Ditto	FLWT[3:0]=4'b0110 Insert 6 waiting period
636KHz < F _{HCLK} ≤ 760KHz	Ditto	FLWT[3:0]=4'b0101 Insert 5 waiting period
512KHz < F _{HCLK} ≤ 636KHz	Ditto	FLWT[3:0]=4'b0100 Insert 4 waiting period
388KHz < F _{HCLK} ≤ 512KHz	Ditto	FLWT[3:0]=4'b0011 Insert 3 waiting period
264KHz < F _{HCLK} ≤ 388KHz	Ditto	FLWT[3:0]=4'b0010 Insert 2 waiting period
140KHz < F _{HCLK} ≤ 264KHz	Ditto	FLWT[3:0]=4'b0001 Insert 1 waiting period
F _{HCLK} ≤ 140KHz	Ditto	FLWT[3:0]=4'b0000 No wait read

7.5 FLASH Read Acceleration Cache

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

In order to read FLASH data quickly, FLASH controller configures a read acceleration cache, which optimizes the read wait period. In order to give full play to the performance of the processor, the accelerator saves the ICODE and DCODE bus access data of FLASH into the cache register, thereby improving the program execution speed.

The system provides 1KBytes of space as cache memory to effectively reduce the time loss caused by instruction jump. Cache enable (CACHE) position 1 in EFM _ FRMCregister makes cache function effective. Whenever an instruction or data miss occurs (that is, the requested instruction or data is not present in the currently used instruction line or cache memory), the system copies the newly read data line (128 bits) into the cache memory. If the CPU requests an instruction or data that already exists in the cache, it can be acquired immediately without any delay. After the cache is full, the LRU (least recently used) policy is used to determine the data to be replaced in the cache.

When the CPU reads instructions or data, the FLASH address is buffered. When the cache hits, the number of cycles to read the FLASH will change. For details, please refer toTable 7-2.

Table 7-2 FLASH Actual Read Cycles

EFM_FRM.C. FLWT[3:0] setting	Cache disabled (EFM_FRM.C.CACHE=0)		Cache enable (EFM_FRM.C.CACHE=1)	
	Buffering, cache hit	Buffering, cache miss	Buffering, cache hit	Buffering, cache miss
0	1	1	1	1
1	1	2	1	2
2	1	3	1	3
3	1	4	1	5
4	1	5	1	6
N(N>4)	1	N+1	1	N+2

7.6 FLASH Programming and Erasing Operations

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

The FLASH programming unit is 4Bytes, and the last bit of the programming address must be aligned with 4 (the last bit address is: 0x0, 0x4, 0x8, 0xC). Repeated programming cannot ensure the correctness of programming. The erase unit of FLASH sector is 8KBytes. Before erasing and programming the FLASH, please disable the cache. The following describes the setup steps for programming and erasing operations.

7.6.1 Single Programming Read-free Mode

The steps for setting single programming read-free mode are as follows:

- 1) Release the write protection of FLASH registers. (EFM_FAPRT first writes 0x0123, then 0x3210)
- 2) Set programming, erase mode permissions. (EFM_FWMC.PEMODE=1)
- 3) Set a single programming mode. (EFM_FWMC.PEMODE[2:0]=001)
- 4) Write 32-bit data to the programming address.
- 5) Waiting for FLASH to be idle. (EFM_FSR.RDY=1)
- 6) Read out the programming address value to judge whether it is consistent with the written value;
Consistent, indicating successful programming, inconsistent, indicating that the FLASH address has been destroyed and permanently discarded.
- 7) Clear the end of programming flag. (EFM_FSR.OPTEND)

7.6.2 Single Programming Read-back Mode

Single programming readback mode refers to self-reading the programming address after programming and comparing it with the written data, and outputting the consistent flag bit EFM_FSR.PGMISMTCH.

The steps for setting single-programming read-back mode are as follows:

- 1) Release the write protection of FLASH registers. (EFM_FAPRT first writes 0x0123, then 0x3210)
- 2) Set programming, erase mode permissions. (EFM_FWMC.PEMODE=1)
- 3) Set a single programming read-back mode. (EFM_FWMC.PEMODE[2:0]=010)
- 4) Write 32-bit data to the programming address.
- 5) Waiting for FLASH to be idle. (EFM_FSR.RDY=1)
- 6) Judge the programming self-read result flag bit. (EFM_FSR.PGMISMTCH) If it is 0, it means that the programming is successful; if it is 1, it means that the FLASH address has been destroyed and is permanently discarded.

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

7.6.3 Continuous Programming Operation

Continuous programming mode is recommended when continuously programming FLASH addresses. Continuous programming mode can save more than 50% time than single programming mode. In continuous programming mode, the continuous programming and writing interval must be less than 16us. Continuous programming operations are set as follows:

- 1) Release the write protection of FLASH registers. (EFM_FAPRT first writes 0x0123, then 0x3210)
- 2) Set programming, erase mode permissions. (EFM_FWMC.PEMODE=1)
- 3) Set continuous programming mode. (EFM_FWMC.PEMOD[2:0]=011)
- 4) Pass step 8) 9) 10) 11) 12) 13) 14) to an area outside of FLASH for execution.
- 5) Go to Step 4) Destination Address.
- 6) Read the programming address and judge whether it is consistent with the written value.
- 7) Consistent, indicating successful programming, inconsistent, indicating that the FLASH address has been destroyed and permanently discarded.
- 8) Write 32-bit data to the programming address.
- 9) Wait for the end of operation flag bit (EFM_FSR.OPTEND) to be set.
- 10) Clear the operation end flag until the flag EFM_FSR.OPTEND is read as 0.
- 11) Repeat 8), 9), 10) until all data is written.
- 12) Modify the wipe mode control register to discontinuous programming mode.
- 13) Waiting for FLASH to be idle. (EFM_FSR.RDY=1)
- 14) Jumping back to the main program.

Note:

- During FLASH continuous programming, an indeinterrupt value will be read if a read operation to FLASH occurs. The read operation sets the read conflict EFM_FSR.COLERR to be cleared by setting the EFM_FSCLRRegister.

7.6.4 Erase Operation

EFM provides two erase methods: Sector erase and whole erase. After the sector erase operation is performed on the FLASH, the data of the address (8KBytes space) in the sector is refreshed to all 1s; after the full erase operation is performed on the FLASH, the data of all addresses (except OTP) in The entire FLASH area are refreshed to all 1s. The steps for setting sector and whole erase operations are as follows:

- 1) Release the write protection of FLASH registers. (EFM_FAPRT first writes 0x0123, then 0x3210)
- 2) Set programming, erase mode permissions. (EFM_FWMC.PEMODE=1)

- 3) Set the erase mode. (Sector Erase EFM_FWMC.PEMOD[2:0]=100, Full Erase EFM_FWMC.PEMOD[2:0]=101)
- 4) Write any 32-bit value to any address in the sector that needs to be erased (address to be aligned with 4).
- 5) Write 32-bit arbitrary value to any FLASH address (address must be aligned with 4) during whole erase.
- 6) Waiting for FLASH to be idle. (EFM_FSR.RDY=1)
- 7) Clear the erase end flag bit. (EFM_FSR.OPTEND)

7.6.5 Data Security Protection

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

Protection level 0: no protection

Debug interface, ISP interface and test interface can access (read and rewrite) MCU resources, including FLASH data.

Protection level 1:

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

After protection level 1 is enabled

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

After activation of protection level 1, it can be restored to protection level 0 through password authentication and full erasure. Please consult the sales office for the password authentication method and the full erasing method.

Protection level 2:

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

- The debug interface is closed, and the ISP interface and test interface cannot access FLASH data.

- User program cannot program and erase sector 0, sector 1 and sector 63.
- The data at address 0x0317_FFE0~0x0317_FFF7 cannot be read.

After activation of protection level 2, it can be reverted to protection level 0 by full erasing.

Protection level 1 and protection level 2 can be enabled individually or at the same time. When enabled at the same time, the protection measures take effect superimposed.

7.6.6 Bus hold function

By setting registerEFM_FWMC.BUSHLDCTL bits, FLASH programming can be set. During erasing, the bus is on hold or released. FLASH programming, when the erase instruction is executed on FLASH, the control bit must be set to 0; This control bit can be freely set as needed when an erase instruction is executed in a space other than FLASH, such as speed RAM.

When BUSHLDCTL is set to 1 (FLASH programming, flashtime, bus release state), read-write access to FLASH before the end of the flashtime (EFM_FSR.RDY = 1) is ignored, flags are EFM_FSR.COLERR position bits.

7.6.7 FLASH Wipe, Programming Window Protection

Provides window protection for FLASH, which can only be erased and programmed by sectors in the allowed area FLASH, otherwise, an erasing error interrupts. When programming and sector erasing operations, the hardware circuit will pre-judge whether it is in the allowed area, whole erase mode is not limited by window protection. The start and end positions of the window are set by registerEFM_FPMTEW and EFM_FPMTSW.

Specific protection functions are as follows:

- RegisterEFM_FPMTEW = registerEFM_FPMTSW The entire FLASH area is erasable and programmed.
- RegisterEFM_FPMTEW > registerEFM_FPMTSW is erasable and the programming area is between them.
- Register EFM_FPMTEW < register EFM_FPMTSW The entire FLASH area cannot be erased and programmed.

7.6.8 Interrupt

The EFM module has 3 interrupts in total, which are PE (programming/erasing) error interrupt, read-write conflict interrupt and operation end interrupt.

FLASH cannot be programmed/erased/erased completely after PE error interrupt is set in FLASH. You must reset the flag bit before you can program/erase/erase FLASH again.

1. PE error interrupt EFM _ PEERR:

Position:

- The program address is not aligned by 4 or the data size is not 32 bits (PGSZERR=1).
- Program FLASH window protection inland address, sector wipe operation (PEPRERR = 1).
- Perform a write operation on FLASH (PEWERR = 1) when the write mode is not set.
- In single programming read-back mode, the programming address self-read value does not match the write value (PGMISMTCH = 1).

Clearance:

Register EFM _ FSCLR corresponds to flag clear bit write 1 and status bit reset.

2. FLASH read and write conflict interrupt EFM _ COLERR:

Position:

- FLASH read operation occurs in FLASH continuous programming mode.
- FLASH read/write operation occurs when FLASH stops mode.
- FLASH programming, FLASH read and write operations occur before the erase ends.
- Write operations to FLASH occur in low-power read mode.

Clearance:

Register EFM _ FSCLR corresponds to purge position 1, status bit zeroed.

3. End of operation interrupt EFM _ OPTEND:

Position:

- Programming mode: End of programming for a single address.
- Wipe mode: Sector wipe, full wipe end.

Clearance:

Register EFM _ FSCLR corresponds to purge position 1, status bit zeroed.

7.7 One Time Programmable Byte

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

Table 7-3 OTP address composition

block name	OTP data block address	OTP lock address
Block 0	0x0300_0C00~0x0300_0C3F	0x0300_0FC0~0x0300_0FC3
Block 1	0x0300_0C40~0x0300_0C7F	0x0300_0FC4~0x0300_0FC7
Block 2	0x0300_0C80~0x0300_0CBF	0x0300_0FC8~0x0300_0FCB
Block 3	0x0300_0CC0~0x0300_0CFF	0x0300_0FCC~0x0300_0FCF
Block 4	0x0300_0D00~0x0300_0D3F	0x0300_0FD0~0x0300_0FD3
Block 5	0x0300_0D40~0x0300_0D7F	0x0300_0FD4~0x0300_0FD7
Block 6	0x0300_0D80~0x0300_0DBF	0x0300_0FD8~0x0300_0FDB
Block 7	0x0300_0DC0~0x0300_0DFF	0x0300_0FDC~0x0300_0FDF
Block 8	0x0300_0E00~0x0300_0E3F	0x0300_0FE0~0x0300_0FE3
Block 9	0x0300_0E40~0x0300_0E7F	0x0300_0FE4~0x0300_0FE7
Block 10	0x0300_0E80~0x0300_0EBF	0x0300_0FE8~0x0300_0FEB
Block 11	0x0300_0EC0~0x0300_0EFF	0x0300_0FEC~0x0300_0FEF
Block 12	0x0300_0F00~0x0300_0F3F	0x0300_0FF0~0x0300_0FF3
Block13	0x0300_0F40~0x0300_0F7F	0x0300_0FF4~0x0300_0FF7
Block 14	0x0300_0F80~0x0300_0FBF	0x0300_0FF8~0x0300_0FFB

The OTP area is divided into 15 data blocks of 64 bytes, and each data block corresponds to a 4Bytes latch address. The latch address is used to latch the corresponding data block. When the latch address data is all 1, the corresponding OTP area data block can be programmed; when the latch address data is all 0, the corresponding OTP area data cannot be programmed. All OTP data blocks and latch addresses cannot be erased.

7.8 Boot Swap

If users want to upgrade the bootloader, they need to erase and write sector 0 (0x0000_0000~0x0000_1FFF). If unexpected accidents (such as reset, power failure) occur during erasing, the entire chip may not start normally. For 512KB products, EFM provides a boot sector swap function (256KB products do not have this function). Before erasing sector 0, write the new boot program into sector 1 (0x0000_2000~0x0000_3FFF) in advance, and then program data 0xFFFF_4321 to EFM address 0x0007_FFDC. After the MCU is reset, the CPU starts a new boot program from sector 1, and then erases sector 0 and reprograms a new user program.

Please refer to the start exchange operation Figure 7-3.

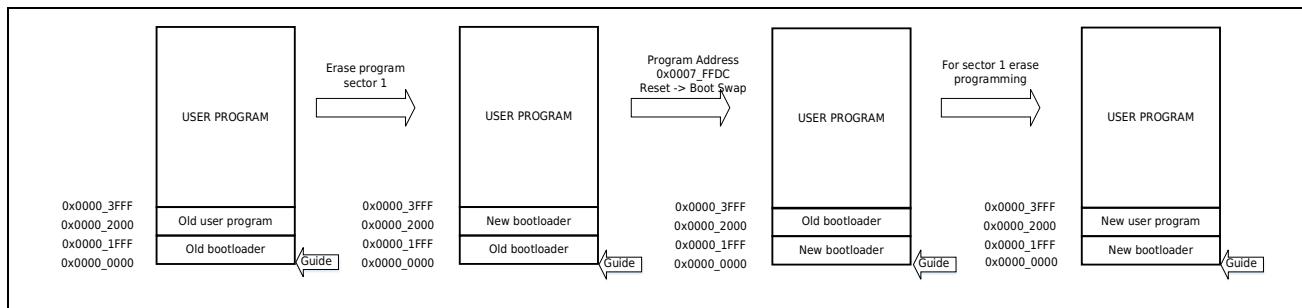


Figure 7-3 Start sector swap function 1

When the user needs to upgrade the boot program again, since the address 0x0007_FFDC for saving the boot sector exchange information has been programmed (the user can judge whether the boot exchange function has been used by reading the FLASH address or the EFM_FSWP register), sector 63 needs to be sectorized. Erase and do the bootloader upgrade again. Before sector 0 is erased, a new boot program is written into sector 1 in advance, and then sector 63 is erased. After the MCU is reset, the CPU starts a new boot program from sector 1, and then erases sector 0 and reprograms a new user program. The operation process is shown in Figure 7-4.

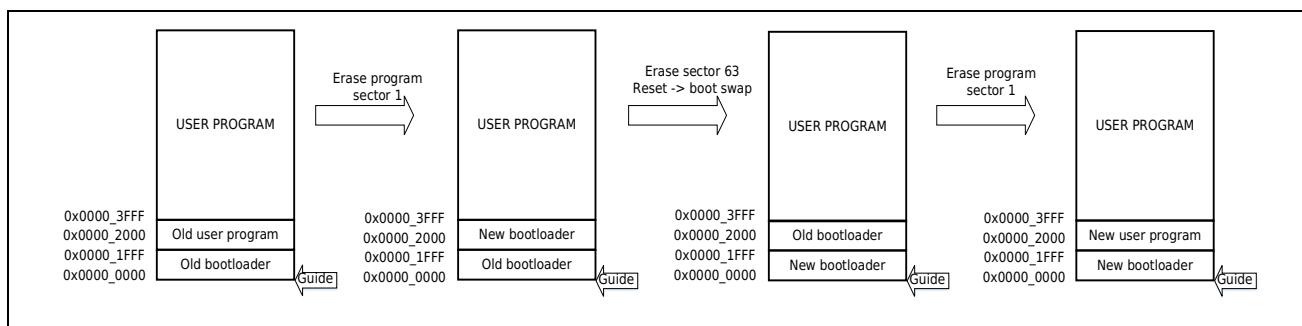


Figure 7-4 Start swap function 2

7.9 Register Description

EFM_BASE_ADDR: 0x4001_0400

Table 7-4 Register Summary

Register description	Register name	Offset	Bit width	Reset value
FLASH access protection register	EFM_FAPRT	0x0000	32	0x0000_0000
FLASH stop register	EFM_FSTP	0x0004	32	0x0000_0000
FLASH read mode register	EFM_FRMC	0x0008	32	0x0000_0000
FLASH Write Mode register	EFM_FWMC	0x000C	32	0x0000_0000
FLASH status register	EFM_FSR	0x0010	32	0x0000_0100
FLASH state clear register	EFM_FSCLR	0x0014	32	0x0000_0000
FLASH interrupt enable register	EFM_FITE	0x0018	32	0x0000_0000
FLASH Boot Swap Status Register	EFM_FSWP	0x001C	32	Indefinite
FLASH write allow area start address register	EFM_FPMTSW	0x0020	32	0x0000_0000
FLASH write allow area end address register	EFM_FPMTEW	0x0024	32	0x0000_0000
FLASH unique ID register	EFM_UQID1	0x0050	32	Indefinite
FLASH unique ID register	EFM_UQID2	0x0054	32	Indefinite
FLASH unique ID register	EFM_UQID3	0x0058	32	Indefinite

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	Marking	Place name	Function	Read and write
b31-b16	Reserved	-	Read as "0", write as "0"	R
b15-b0	FAPRT[15:0]	EFMregister write protection	Access EFMregister protected register. Method: Write "16-bit data 0x0123" before "16-bit data 0x3210" on FAPRT. In the unprotected state, any data is written, and EFMregister enters the protected state again. When the EFMregister access protection is valid, the register read value is 0x0000_0000. When EFMregister access protection is invalid, the register read value is 0x0000_0001.	R/W

7.9.2 FLASH Stop Register (EFM_FSTP)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31-b1	Reserved	-	Read as "0", write as "0"	R
b0	FSTP	FLASH stop mode control	0: FLASH active state 1: FLASH in stop mode When the register bit is set from 1 to 0, please confirm that the FSR.RDY bit is 1, and then perform FLASH access.	R/W

7.9.3 Read Mode Register (EFM _ FRMC)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b25	Reserved	-	Read as "0", write as "0"	R
b24	CRST	Cache reset	0: Cached data not reset 1: Reset Cache Data	R/W
b23~b17	Reserved	-	Read as "0", write as "0"	R
b16	CACHE	Cache permission bit	0: Disable caching 1: Cache function enabled	R/W
b15~b9	Reserved	-	Read as "0", write as "0"	R
b8	LVM	Ultra low speed operation mode	0: Close the ultra-low speed operation mode 1: Turn on the ultra-low speed operation mode	R/W
b7-b4	FLWT[3:0]	Waiting period of FLASH read insertion	0000b: Do not insert wait cycles 0001b: Insert 1 wait cycle 0010b: Insert 2 wait cycles 1110b: insert 14 wait cycles 1111b: insert 15 wait cycles	R/W
b3~b1	Reserved	-	Read as "0", write as "0"	R
b0	SLPMD	Ultra Low Power Read	0: Common CPU read mode 1: Ultra-low power read mode	R/W

7.9.4 Erase Mode Register (EFM _ FWMC)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	BUS HLDC TL	-	PEMOD[2:0]	-	-	-	-	PEM ODE	

Bit	Marking	Place name	Function	Read and write
b31~b9	Reserved	-	Read as "0", write as "0"	R
b8	BUSHLDCTL	FLASH wipe, programming period bus control	0: The bus was occupied during FLASH programming erase. 1: The bus is released during FLASH programming erase.	R/W
b7	Reserved	-	Read as "0", write as "0"	R
b6~b4	PEMOD[2:0]	FLASH Erase, Programming Mode	000: Read-only mode 001: Single programming mode 010: Single-programming read-back mode 011: Continuous programming mode 100: Sector Wipe Mode 101: Full wipe mode 110: Read-only mode 111: Read-only mode PEMOD [2: 0] can only be written in the case of PEMODE = 1.	R/W
b3~b1	Reserved	-	Read as "0", write as "0"	R
b0	PEMODE	FLASH Erase, Programming Licensing Mode	0: PEMOD [2: 0] overridden 1: PEMOD[2:0] rewrite permission	R/W

7.9.5 Status Register (EFM_FSR)

Reset value: 0x0000_0100

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

-	-	-	-	-	-	-	RDY	-	-	COLE RR	OPTE ND	PGMISMTCH	PGSZERR	PEPR TERR	PEW ERR
---	---	---	---	---	---	---	-----	---	---	------------	------------	-----------	---------	--------------	------------

Bit	Marking	Place name	Function	Read and write
b31~b9	Reserved	-	Read as "0", write as "0"	R
b8	RDY	FLASH Busy/Idle State	0: Flash Busy State 1: FLASH idle state	R
b7~b6	Reserved	-	Read as "0", write as "0"	R
b5	COLERR	FLASH read/write access error flag bit	0: FLASH read/write access normal 1: FLASH read/write access error The FLASH read-write access operation for this bit will be ignored.	R
b4	OPTEND	End of operation flag bit	0: Not in FLASH or FLASH 1: end of flash erase, Setting conditions: End of programming/erase/whole erase operation	R
b3	PGMISMTCH	Single programming read-back value mismatch flag bit	0: Single programming read-back values are consistent 1: Inconsistent read-back values for single programming	R
b2	PGSZERR	Program address and size misalignment flag	0: Program address and size alignment 1: Program address and size are not aligned	R
b1	PEPR TERR	Programming/erasing error flag bits for protected addresses	0: Programming/ erasing address to allow overwriting 1: Programming/erasing the address of the protection window	R
b0	PEW ERR	Erase mode error flag bit	0: Write FLASH in Write permission mode 1: Erase FLASH in Write-Not-Allowed Mode	R

7.9.6 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
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	COLE RRCLR	OPTENDCLR	PGMISM TCHCLR	PGSZERRCLR	PEPTERRCLR	PEWERRCLR

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R
b5	COLERRCLR	Clear read and write conflict error flag bit	0: No purge action 1: Clear read and write conflict errors When read, this bit is always 0.	R/W
b4	OPTENDCLR	End-of-purge flag	0: No purge action 1: End-of-purge flag When read, this bit is always 0.	R/W
b3	PGMISMTCCHCLR	Clear programming read-back mismatch flag bit	0: No purge action 1: Clear programming read-back inconsistency flag bit When read, this bit is always 0.	R/W
b2	PGSZERRCLR	Clear programming address and size misalignment flag	0: No purge action 1: Clear misalignment error flag bit When read, this bit is always 0.	R/W
b1	PEPRTERRCLR	Clear programming/erasing error flag bits for protected addresses	0: No purge action 1: Erase programming/erase error When read, this bit is always 0.	R/W
b0	PEWERRCLR	Erase mode error flag bit	0: No purge action 1: Erase mode write error When read, this bit is always 0.	R/W

7.9.7 Interrupt 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	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R
b2	COLERRITE	read-write conflict error interrupt permission	0: Read and write conflict error interrupt is not allowed 1: Read and write conflict error interrupt permission	R/W
b1	OPTENDITE	End of operation interrupt enable	0: End of operation interrupt disallowed 1: End of operation interrupt enable	R/W
b0	PEERRITE	Programming/Erase Error Interrupt License	0: Programming/Erase Error Interrupt Not Permitted 1: Programming/Erase Error Interrupt License	R/W

7.9.8 Boot Swap Status Register (EFM_FSWP)

Reset value: Indefinite

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

Bit	Marking	Place name	Function	Read and write
b31-b1	Reserved	-	Read as "0", write as "0"	R
b0	FSWP	Sector 0 and Sector 1 Address Swap Bits	0: Sector 0 and sector 1 address exchange After reset, the CPU starts from sector 1. 1: The addresses of sector 0 and sector 1 are not exchanged After reset, the CPU starts from sector 0. The initial value of the register is determined by the FLASH address 0x0007_FFDC~0x0007_FFDF. When the address data is 0xFFFF_4321, the initial value of the FSWP register bit FSWP is 0 after reset, otherwise it is 1.	R

7.9.9 FLASH Window Protection Start Address register (EFM_FPMTSW)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31-b19	Reserved	-	Read as "0", write as "0"	R
b18-b0	FPMTSW[18: 0]	Protection window start address	FLASH protection window start address	R/W

7.9.10 FLASH Window Protection End Address register (EFM_FPMTEW)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31-b19	Reserved	-	Read as "0", write as "0"	R
b18-b0	FPMTEW[18: 0]	Protection window end address	End address of FLASH protection window	R/W

7.9.11 UNIQUE ID Register (EFM_UQID1)

Reset value: Indefinite

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	Marking	Place name	Function	Read and write
b31-b0	UQID1[31:0]	Unique code	Chip unique code	R

7.9.12 UNIQUE ID Register (EFM_UQID2)

Reset value: Indefinite

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]															

Bit	Marking	Place name	Function	Read and write
b31-b0	UQID2[31:0]	Unique code	Chip unique code	R

7.9.13 UNIQUE ID Register (EFM_UQID3)

Reset value: Indefinite

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
UQID3[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UQID3[15:0]															

Bit	Marking	Place name	Function	Read and write
b31-b0	UQID3[31:0]	Unique code	Chip unique code	R

7.10 Precautions

1. When the FLASH is erased, reset, erasing operation will be forced to stop, FLASH data will not be guaranteed. The user needs to erase the address after resetting.
2. If the erasing and writing operation will involve the data in the cache, please reset the cache circuit (FRMC.CRST0=1) after the erasing and writing operation is completed.
3. Please set the cache as invalid (FRMC.CACHE=0) before programming and erasing.
4. The one-time programmable area programming operation is not controlled by the window protection registers (FPMTSW, FPMTEW).
5. In continuous programming mode, FLASH analog circuit will have high voltage state, long-term high voltage state will affect FLASH characteristic. In continuous programming mode, MCU is not allowed to enter low-power mode (sleep mode, stop mode, power-off mode).
6. When setting the bus release (FWMC.BUSHLDCTL=1) during programming and erasing, if you need to respond to interrupts during programming and erasing, please set the interrupt vector and interrupt subroutine to RAM.

8 Built-in SRAM (SRAM)

8.1 Introduction

This product has 4KB power-down mode retention SRAM (Ret_SRAM) and 188KB system SRAM (SRAMH/ SRAM1/ SRAM2/ SRAM3).

SRAM can be accessed by byte, half word (16 bits) or full word (32 bits). Read and write operations are performed at CPU speed, with insertable wait cycles. The relationship between the wait cycle setting for read and write access and the CPU clock frequency is shown in Table 8-1the figure. The waiting cycle of each SRAM's read and write access is set by the SRAM wait control register (SRAM_WTCR).

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

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

Ret_SRAM can provide 4KB data holding space in Power down mode.

SRAM3 has ECC check (Error Checking and Correcting). ECC check is one-check-two code, which can correct one bit error and check two bit errors; SRAMH/ SRAM1/ SRAM2/ Ret_SRAM has parity check (Even- parity check), each byte of data has a parity bit. See SRAM for a detailed definitionTable 8-2.

Table 8-2 SRAM Space Allocation

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

Note:

- When using SRAM3 as a stack space, the waiting time of SRAM3 must be set to 1wait, that is, more than 2 CPU cycles for access.
- In the case where RAM parity errors are allowed to generate NMI interrupts and resets, when accessing data, the RAM space used must be initialized in word units; when executing instructions from the SRAMH space, the RAM space used must be +3 The word area is initialized in units of words.
- In the case of allowing RAM ECC verification errors to generate NMI interrupts and resets, when accessing data, the used RAM space must be initialized in word units.
- The boundaries of SRAMH and SRAM1 do not support non-aligned access. In use, it is necessary to avoid initiating 32bit access to 0x1FFF_FFFD/ 0x1FFF_FFFE/ 0x1FFF_FFFF address, and initiating 16bit access to 0x1FFF_FFFF address.

8.2 Register Description

Register name	initial address	Reset value
SRAM Wait Control Register (SRAM_WTCR)	0x4005_0800	0x0000_0000
SRAM Wait Protection Register (SRAM_WTPR)	0x4005_0804	0x0000_0000
SRAM check control register (SRAM_CKCR)	0x4005_0808	0x0000_0000
SRAM check protection register (SRAM_CKPR)	0x4005_080C	0x0000_0000
SRAM Check Status Register (SRAM_CKSR)	0x4005_0810	0x0000_0000

8.2.1 SRAM Wait Control Register (SRAM_WTCR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Rev	SRAMR_WWT[2: 0]	Rev	SRAMR_RWT[2: 0]	Rev	SRAMH_WWT[2: 0]	Rev	SRAMH_RWT[2: 0]	Rev	SRAM3_WWT[2: 0]	Rev	SRAM3_RWT[2: 0]	Rev	SRAM12_WWT[2: 0]	Rev	SRAM12_RWT[2: 0]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Rev	SRAM3_WWT[2: 0]	Rev	SRAM3_RWT[2: 0]	Rev	SRAM12_WWT[2: 0]	Rev	SRAM12_RWT[2: 0]	Rev	SRAM3_WWT[2: 0]	Rev	SRAM3_RWT[2: 0]	Rev	SRAM12_WWT[2: 0]	Rev	SRAM12_RWT[2: 0]

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30~b28	SRAMR_WWT[2: 0]	Ret_SRAM write cycle selection	000b: 1 cycle write 001b: 2-cycle write 010b: 3-cycle write 011b: 4-cycle write 100b: 5-cycle write 101b: 6-cycle write 110b: 7-cycle write 111b: 8-cycle write	R/W
b27	Reserved	-	Read as "0", write as "0"	R/W
b26~b24	SRAMR_RWT[2: 0]	Ret_SRAM read cycle selection	000b: 1 cycle read 001b: 2-cycle read 010b: 3-cycle read 011b: 4-cycle read 100b: 5-cycle read 101b: 6-cycle read 110b: 7-cycle read 111b: 8-cycle read	R/W
b23	Reserved	-	Read as "0", write as "0"	R/W
b22~b20	SRAMH_WWT[2: 0]	SRAMH write cycle selection	000b: 1 cycle write 001b: 2-cycle write 010b: 3-cycle write 011b: 4-cycle write 100b: 5-cycle write 101b: 6-cycle write 110b: 7-cycle write 111b: 8-cycle write	R/W
b19	Reserved	-	Read as "0", write as "0"	R/W
b18~b16	SRAMH_RWT[2: 0]	SRAMH read cycle selection	000b: 1 cycle read 001b: 2-cycle read 010b: 3-cycle read 011b: 4-cycle read 100b: 5-cycle read 101b: 6-cycle read 110b: 7-cycle read 111b: 8-cycle read	R/W
b15	Reserved	-	Read as "0", write as "0"	R/W

			000b: 1 cycle write 001b: 2-cycle write 010b: 3-cycle write 011b: 4-cycle write 100b: 5-cycle write 101b: 6-cycle write 110b: 7-cycle write 111b: 8-cycle write	
b14~b12	SRAM3_WWT[2: 0]	SRAM3 write cycle selection		R/W
b11	Reserved	-	Read as "0", write as "0"	R/W
b10~b8	SRAM3_RWT[2: 0]	SRAM3 read cycle selection	000b: 1 cycle read 001b: 2-cycle read 010b: 3-cycle read 011b: 4-cycle read 100b: 5-cycle read 101b: 6-cycle read 110b: 7-cycle read 111b: 8-cycle read	R/W
b7	Reserved	-	Read as "0", write as "0"	R/W
b6~b4	SRAM12_WWT[2: 0]	SRAM1 and SRAM2 write cycle selection	000b: 1 cycle write 001b: 2-cycle write 010b: 3-cycle write 011b: 4-cycle write 100b: 5-cycle write 101b: 6-cycle write 110b: 7-cycle write 111b: 8-cycle write	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2~b0	SRAM12_RWT[2: 0]	SRAM1 and SRAM2 read cycle selection	000b: 1 cycle read 001b: 2-cycle read 010b: 3-cycle read 011b: 4-cycle read 100b: 5-cycle read 101b: 6-cycle read 110b: 7-cycle read 111b: 8-cycle read	R/W

8.2.2 SRAM Wait Protection Register (SRAM_WTPR)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b1	WTPRKW[6:0]	Write key	When writing to the current register, you need to write "3b" in 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 waits for control register write enable	R/W

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

8.2.3 SRAM Check Control Register (SRAM_CKCR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Rev				ECCMOD [1: 0]		Rev				ECC OAD					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Rev															PYOAD
Bit	Marking		Place name		Function								Read and write		
b31~b26	Reserved		-		Read as "0", write as "0"								R/W		
b25~b24	ECC MOD[1: 0]	ECC parity enable bit of SRAM3		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, A 1-bit error flag is generated, no interrupt/reset is generated; 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			
b23~b17	Reserved	-		Read as "0", write as "0"								R/W			
b16	ECCOAD	ECC check Action after error		0: Non-maskable interrupt 1: Reset								R/W			
b15~b1	Reserved	-		Read as "0", write as "0"								R/W			
b0	PYOAD	Parity Action after error		0: Non-maskable interrupt 1: Reset								R/W			

Note:

- In the case where RAM parity errors are allowed to generate NMI interrupts and resets, when accessing data, the RAM space used must be initialized in word units; when executing instructions from the SRAMH space, the RAM space used must be +3 The word area is initialized in units of words.
- In the case of allowing RAM ECC verification errors to generate NMI interrupts and resets, when accessing data, the used RAM space must be initialized in word units.

8.2.4 SRAM Check Protection Register (SRAM_CKPR)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b1	CKPRKW[6:0]	Write key	When writing to the current register, you need to write "3b" in these bits to enable the current register	R/W
b0	CKPRC	SRAM parity control register write enable	0: SRAM verification control register writing disabled 1: SRAM verification control register write enable	R/W

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

8.2.5 SRAM Check Status Register (SRAM_CKSR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Rev															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Rev										SRAMR_PYERR	SRAMH_PYERR	SRAM12_PYERR	SRAM3_2ERR	SRAM3_1ERR	

Bit	Marking	Place name	Function	Read and write
b31~b5	Reserved	-	Read as "0", write as "0"	R/W
b4	SRAMR_PYERR	Ret_SRAM parity Parity error flag	0: No parity error occurs 1: A parity error occurs	R/W (Note 1)
b3	SRAMH_PYERR	SRAMH Parity Parity error flag	0: No parity error occurs 1: A parity error occurs	R/W (Note 1)
b2	SRAM12_PYERR	SRAM1 and SRAM2 parity Parity error flag	0: No parity error occurs 1: A parity error occurs	R/W (Note 1)
b1	SRAM3_2ERR	SRAM3 occurred ECC 2-bit error flag	0: No 2-bit ECC error occurs 1: 2-bit ECC error occurs	R/W (Note 1)
b0	SRAM3_1ERR	SRAM3 occurred ECC 1-bit error flag	0: No 1-bit ECC error occurs 1: 1-bit ECC error occurs	R/W (Note 1)

Note:

- Write 1 to clear 0.

9 General IO (GPIO)

Some of the abbreviations used in this chapter are:

- Px (x=A~E,H) indicates a group of ports, such as PA indicates the 16 I/O ports of PA0~PA15.
- Pxy (x= A~E,H, y=0~15, the same below) means a single I/O port, for example, PB10 port means the 10th I/O in 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
- Supports high, medium and low drive modes
- Support for external interrupt input
- Support I/O pin peripheral function multiplexing, one I/O pin can have up to 64 optional multiplexing functions
- Individual I/O pins can be programmed independently
- Each I/O pin can select 2 functions to be valid at the same time (does not support 2 output functions to be valid at the same time)

9.2 Port Function Summary

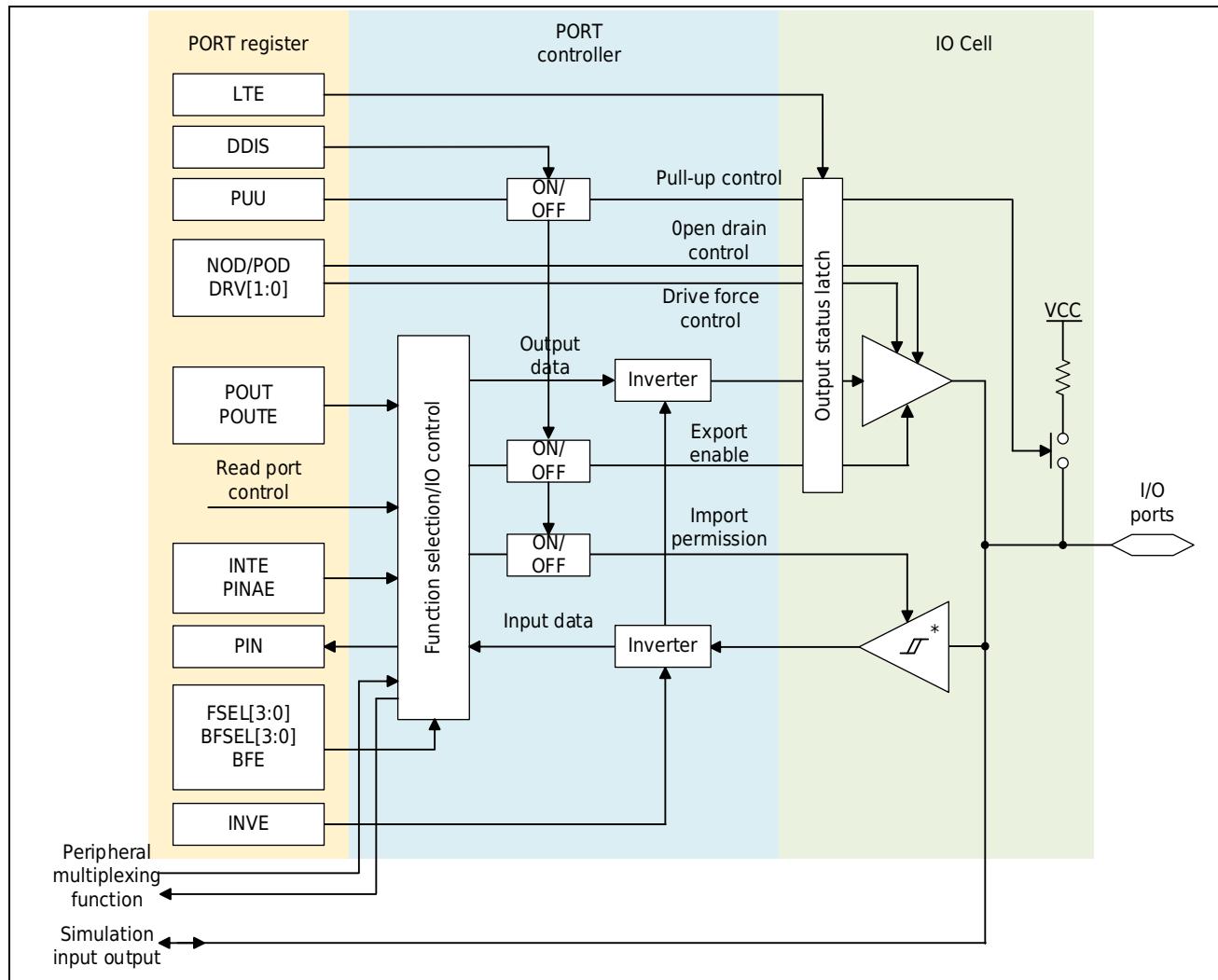


Figure 9-1 Port Basic Structure Diagram

For details on the number of GPIO ports, 5V withstand voltage, and drive capability configuration, please refer to the pin configuration and function chapters in the Data Manual.

9.3 Action 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 setting value of FSEL[5:0] in the function selection register PFSRxy. The status of the current port can be obtained by accessing port input data registerPIDRx. The corresponding single I/O port status can also be queried through the PIN bit of the port control register PCRxy, and the PIDRx.PIN[y] register bit is equivalent to the PCRxy.PIN bit.

In order to reduce power consumption, when the I/O does not select peripheral functions, the input MOS of the I/O is turned off. Only when PIDRx is read, PCRxyregister is opened. This mode of operation of I/O will make its input hysteresis characteristics unable to function normally, because the input MOS is from the high level of the off state to the high or low level (according to the input value of I/O) for Each read operation. Switching, so the threshold VIL from high level to low level is normal, but the threshold VIH from low level to high level cannot be used. In order for the I/O to play the input hysteresis characteristic normally, it is necessary to keep the MOS in the open state. This can be achieved by setting the register PINAER.PINAE[x] to 1, or setting PFSRxy to select a peripheral function (except GPO).

Due to the delay of I/O input, when the system is running on a high-speed clock, the input status value may not be read correctly in a single cycle. At this time, it is necessary to set the register PCCR.RDWT[1:0] and insert several wait cycles. For details, refer to the description of the register PCCR.

General Output Function GPO:

Except for input-specific PB11 ports, other I/O ports have general output GPO capabilities. 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 registerPOERx to allow or disable the output of I/O and the generic output data registerPODRx. The output values of I/O can also be controlled using the following three registers: Output data zeroed registerPORRx, output data set registerPOSRx, and output data reversed registerPOTRx. Writing 1 to the corresponding bits in the register can make the corresponding I/O output 0, 1, invert. I/O output status does not change at write 0.

The registers above are operated by 16 PORT groups. For easy control of a single I/O, I/O output can also be allowed or disabled by setting PCRxy.POUTE, the PCRxy.POUREregister bit is equivalent to POERx.POUTE [y]. The output value of I/O can be controlled by setting PCRxy.POOUT, where the

PCRxy.POUTregister bit is equivalent to PODRx.POUT [y]. PCRxy is suitable for controlling a single PORT and POERx/PODRx is suitable for controlling an 16-bit set of PORTs.

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

Note:

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

9.3.2 Peripheral Functions

Through the FSEL[5:0] of the function selection register PFSRxy, each port can configure up to 64 functions. These include the generic output GPO function corresponding to FSEL [5: 0] = 0x0. For the specific configuration functions of each port, please refer to the pin function table in the Data Manual.

JTAG/SWD debug function, use register PSPCR to select. PSPCR.SPFE[z], when z=0~4 is 1, the PFSRxy.FSEL[5:0] register bit of the corresponding port is invalid, that is, the priority of SPFE is higher than that of FSEL. The initial value of the PSPCR register is 0x1F, and the JTAG/SWD function is valid. If you want to set these ports as functions 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, it is necessary to set a port 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 PFSRxy.BFE to 1 and setting the public control register PCCR.BFSEL[3:0]. For example: set PFSRxy.FSEL[5:0]=0x2, PCCR.BFSEL[3:0]=0x5, PFSRxy.BFE=0x1, then function 2 and function 5 on Pxy will be valid at the same time. It is forbidden to enable 2 output functions on the same port at the same time.

9.3.4 Event Port Input and Output Functions

Support 4 groups of Event Ports, each with 16 ports. Event Port1 includes EVNTP100~EVNTP115, Event Port2 includes EVNTP200~EVNTP215, and so on. EVNTPmn (m=1~4, n=0~15) The port can be used as a trigger source to trigger other peripheral devices (such as TIMER, ADC, DMA, etc.) to start specific actions according to the event generated by the port input. It can also be used as a triggered object to 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 filter function, and set the function selection register PFSRxy to select EVNTPmn function. When the selected edge is entered 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 PEVNTTRGRm 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 flips according to the setting values of PEVNTODRm, PEVNTORRm, and PEVNTOSRm when the selected event occurs. When the input function is selected, the EVNTPmn input state is saved into the register PEVNTIDRx when the selected event occurs.

When using the Event Port function, it is necessary to first set the automatic operation system AOS function enable bit of the function clock control 0 register (PWC_FCG0) to valid.

Note:

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

9.3.5 External InterruptEIRQ Input Function

Each I/O port has an external interrupt input function. When the PCRxy.INTE bit is set to 1, this I/O is allowed to be entered as an external interrupt source EIRQy. Each EIRQy can be configured with more than one I/O. Please do not allow multiple I/O inputs simultaneously for each EIRQy. The EIRQy input function and the peripheral functions selected by PFSRxy.FSEL (including GPIO) can be valid at the same time.

In addition, external non-shielded interruptNMI is multiplexed with PB11/MD ports.

When I/O port is used as external interruptEIRQ, 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)] for details.

9.3.6 Analogfunction

Some I/O ports have analog input and output functions (including primary and secondary 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 resistor

Each I/O port has an internal pull-up resistor. You can set the register PFSRxy.PUU bit to allow 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 PUU will be ignored, and the internal pull-up function will be forced to be invalid.

PA11, PA12 are multiplexed with USBFS_DM, USBFS_DP pins, with a built-in pull-down resistor of about 400KΩ, and it is always valid.

2. Drive capability control

Each I/O port has three levels of high, medium and low drive capability adjustable, and the register PFRSxy.DRV[1:0] can be set as required.

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

3. Open-drain output mode

Setting the PFRSxy.NOD bit can 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 I2Cx_SCL/I2Cx_SDA function, the setting of the register NOD will be ignored, and the open-drain output mode is forced to be valid.

The general control features described above, if not specified, are independent of the port-specific selected features, FSEL [5: 0] settings.

9.4 Register Description

BASE_ADDR: 0x4005_3800

Table 9-1 PORT Register Overview 1

Register name	Symbol	Offset address	Bit width	Reset value
General input data register	PIDRx	0x00+0x10*n *1	16/32	0XXXXX
General output data register	PODRx	0x04+0x10*n	16/32	0x0000
General output 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 flip register	POTRx	0x0C+0x10*n	16/32	0x0000
Special control	PSPCR	0x3F4	16/32	0x001F
Public control	PCCR	0x3F8	16/32	0x4000
Input control register	PINAER	0x3FA	16/32	0x0000
Write protection register	PWPR	0x3FC	16/32	0x0000
General control register	PCRxy	0x400+0x40*n+0x4*y	16/32	0x0X00 *2
Function Select Register	PFSRxy	0x402+0x40*n+0x4*y	16/32	0x0000

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

*2: 32K secondary oscillator multiplex port PCRC14, the reset value of PCRC15 is 0x8100.

BASE_ADDR: 0x4001_0800

Table 9-2 PORT Register Overview 2

Register name	Symbol	Offset address	Bit width	Reset value
Event Port Direction Selection Register	PEVNTDIRm	0x100+0x1C*(m-1)	32	0x0000_0000
Event Port input data register	PEVNTIDRm	0x104+0x1C*(m-1)	32	0x0000_0000
Event Port output data register	PEVNTODRm	0x108+0x1C*(m-1)	32	0x0000_0000
Event Port output data reset register	PEVNTORRm	0x10C+0x1C*(m-1)	32	0x0000_0000
Event Port output data setting register	PEVNTOSRm	0x110+0x1C*(m-1)	32	0x0000_0000
Event Port rising edge input permission register	PEVNTRISRm	0x114+0x1C*(m-1)	32	0x0000_0000
Event Port falling edge input permission register	PEVNTFALRm	0x118+0x1C*(m-1)	32	0x0000_0000
Event Port input filter control register	PEVNTNFCR	0x170	32	0x0000_0000

Note: m=1~4

9.4.1 General Input Register (PIDRx)

Reset value: 0XXXXX

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

Bit	Marking	Place 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 and DDIS=0, the input status 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. DDIS=1 When the digital function of the port is disabled, the corresponding PIN bit readout value is fixed value 0x1 because the I/O input MOS is off.

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

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

9.4.3 General Output License Register (POERx)

Reset value: 0x0000

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

Bit	Marking	Place name	Function	Read and write
b15~b0	POUTE[15:0]	Export enable	0: Output disable 1: Export 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 without port bits.

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

The read value for 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	Place name	Function	Read and write
b15~b0	POR[15:0]	Output low	0: No change to PODRx.POUT 1: corresponding to clear PODRx.POUT	R/W

The read value for this register is always 0x0000.

9.4.6 General Output Flip 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	Place 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 for this register is always 0x0000.

9.4.7 Special Control Register (PSPCR)

Reset value: 0x001F

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

Bit	Marking	Place name	Function	Read and write
b15~b6	Reserved	-	Read at 0, write at 0	R
b4	SPFE[4]	Special function selection	0: Invalid NJTRST feature 1: The NJTRST function is valid	R/W
b3	SPFE[3]	Special function selection	0: Invalid JTDI feature 1: The 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:

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

9.4.8 Public Control Register (PCCR)

Reset value: 0x4000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0											
RDWT[1:0]	-	-	-	-	-	-	-	-	-	-	-	BFSEL[3:0]														

Bit	Marking	Place name	Function	Read and write
Set the number of wait cycles inserted when reading registers PIDRx, PCRxy				
b15-b14	RDWT[1:0]	Read Port Waiting	Set value	Waiting period Recommended frequency of work
			00	No wait ~42MHz
			01 (default)	1 cycle 42-84MHz
			10	2 cycles 84-126MHz
			11	3 cycle 126-200MHz
b13~b4	Reserved	-	Read at 0, write at 0	R
b3~b0	BFSEL[3:0]	Secondary function selection	For the function configuration of each port, please refer to the pin function table in the Data Manual	R/W

9.4.9 Input Control Register (PINAER)

Reset value: 0x0000

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

Bit	Marking	Place name	Function	Read and write
b15~b6	Reserved	-	Read at 0, write at 0	R
b5~b0	PINAЕ[5:0]	Input normally open	0: Input MOS normally open invalid 1: Input MOS normally open 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] Control PH0 ~ PH2	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	Place name	Function	Read and write
b15~b8	WP[7:0]	Write protection code	Read at 0x00 When b15 ~ b8 write value is 0xA5, b0 write WE When writing a value other than 0xA5, WE automatically clears	W
b7~b1	Reserved	-	Read at 0, write at 0	R
b0	WE	Write permission	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	-	-	INVE	PIN	-	PUU	DRV[1:0]	-	NOD	POUTE	POUT	

Bit	Marking	Place name	Function	Read and write
b15	DDIS	Digital function disabled	0: digital function is valid 1: digital function disabled	R/W
b14	LTE	Output state latch	0: Invalid output latch 1: Output latch effective	R/W
b13	Reserved	-	Read at 0, write at 0	R
b12	INTE	External interrupt enable	0: External interrupt input disable 1: External interrupt input enable	R/W
b11~b10	Reserved	-	Read at 0, write at 0	R
b9	INVE	Inverse enable	0: Input/output data not inverted 1: Inverted input/output data	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 registerPIDRx	R
b7	Reserved	-	Read at 0, write at 0	R
b6	PUU	Pull-up enable	0: Invalid internal pull-up resistor 1: Pullup resistance is effective	R/W
b5~b4	DRV[1:0]	Drive mode selection	b00: Low driving force mode b01: Medium driving force mode b1*: High driving force mode	R/W
b3	Reserved	-	Read at 0, write at 0	R
b2	NOD	NMOS Opening Leakage	0: Normal CMOS Output Mode 1: NMOS open-drain output	R/W
b1	POUTE	Export enable	0: Output disable 1: Export 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 invalid, including general-purpose input and output, peripheral digital input and output, pull-up/pull-down functions, and external interrupt input functions. When the port is used as an analog input, please set the DDIS bit to 1.

When LTE is set to 1 Output Lock active, the current output status of the port remains 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, before the function switching, the output status of the port is latched in LTE write 1, and then the register switch function is selected. Finally, the LTE write 0 is unlatched 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.

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

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	Place name	Function	Read and write
b15~b9	Reserved	-	Read at 0, write at 0	R
b8	BFE	Secondary feature license	Control whether the secondary function selected by PCCR.BFSEL[3:0] is valid 0: Minor function disabled 1: The auxiliary function is valid	R/W
b7~b6	Reserved	-	Read at 0, write at 0	R
b5~b0	FSEL[5:0]	Functional selection	For the function configuration of each port, please refer to the pin function table in the Data Manual	R/W

Each I/O port can select one of several features configured on the port via FSEL [5: 0]. Refer to the pin function table in the Data Manual, set FSEL[5:0] to b000000 to select Func0, set to b000001 to select Func1, and so on, set to b001111 to select Func15. The general output function GPO corresponding to Func0 Table 2-1 Pin Function Table.

Note:

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

9.4.13 Event Port Direction Selection Register (PEVNTDIRRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PDIR[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read at 0, write at 0	R											
b15~b0	PDIR[15:0]	Direction selection	0: Event Port is an input function 1: Event Port is an output function	R/W											

Note:

- The EVNTP211 function has no output function (configured on the PB11 port).

9.4.14 Event Port Input Data Register (PEVNTIDRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PIN[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read at 0, write at 0	R											
b15~b0	PIN[15:0]	Port input state	0: Event Port input state is low level when the event is triggered 1: The Event Port input state is high 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.

9.4.15 Event Port Output Data Register (PEVNTODRm)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read at 0, write at 0	R
b15~b0	POUT[15:0]	Port output value	0: Event Port output low level 1: Event Port output high level	R/W

When the direction of the Event Port is set to the output state, write this register, before the set event is triggered, the initial output value of the Event Port. When the selected event is triggered, according to the setting values of PEVNTORRm and PEVNTOSRm, the corresponding bits of PEVNTODRm.POUT are cleared to 0, set to 1, or inverted, and output to the EVNTPmn port at the same time.

9.4.16 Event Port Output Data Reset Register (PEVNTORRm)

Reset value: 0x0000_0000

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

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

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

9.4.17 Event Port Output Data Set Register (PEVNTOSRm)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read at 0, write at 0	R
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	W

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

9.4.18 Event Port Rising Edge Input Permission Register (PEVNTRISRm)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read at 0, write at 0	R
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

Event Port is used as an event source. When the RIS bit is set to 1, when the corresponding EVNTPmn input rising edge, 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 one of the ports will output the event EVENT_PORTm after detecting the edge.

9.4.19 Event Port Falling Edge Input Enable Register (PEVNTFALRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FAL[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	Read at 0, write at 0										R		
b15~b0	FAL[15:0]	Falling edge detection permission	0: EVNTPmn falling edge event detection is invalid 1: EVNTPmn falling edge event detection is valid PEVNTRISRm.FAL[n] corresponds to EVNTPmn										R/W		

Event Port is used as an event source. When the FAL bit is set to 1, when the corresponding EVNTP input falls, 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 one of the ports will output the event EVENT_PORTm after detecting the edge.

9.4.20 Event Port Input Filter Control Register (PEVNTNFCR)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b27	Reserved	-	Read at 0, write at 0	R
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 Filtering License	0: Event Port4 digital filter is invalid 1: Event Port4 digital filter is valid	R/W
b23~b19	Reserved	-	Read at 0, write at 0	R
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 Filtering License	0: Event Port3 digital filter is invalid 1: Event Port3 digital filter is valid	R/W
b15~b11	Reserved	-	Read at 0, write at 0	R
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 Filtering License	0: Event Port2 digital filter is invalid 1: Event Port2 digital filter is valid	R/W
b7~b3	Reserved	-	Read at 0, write at 0	R
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 Filtering License	0: Event Port1 digital filter is invalid 1: Event Port1 digital filter is valid	R/W

9.4.21 32bit Access

Among the registers mentioned above, except the Event Port related registers which only support 32bit access, other registers support 16bit and 32bit access. The combination of these registers during 32bit access is as follows:

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

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

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

9.5 Precautions

Do not set the same feature to multiple ports.

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

Please perform port feature switching when the output latch is active (LTE = 1) to avoid out-of-sight burrs on the port during the switchover.

10 Interrupt Controller (INTC)

10.1 Introduction

The function of the interrupt controller (INTC) is to select the interrupt event request as the interrupt input to NVIC, wake up WFI; select the interrupt event request as the event input, wake up WFE; select the interrupt event request as the low power mode (sleep mode and stop mode) Wake-up condition; interrupt control function of external pins NMI and EIRQ; interrupt/event selection function of software interrupt.

Main specifications:

- 1) NVIC interrupt vector: Please refer to the actual number of interrupt vectors used [10.3.1 Interrupt Digraph] (excluding the 16 interrupt lines of Cortex™ -M4F), each interrupt vector can select the corresponding peripheral interrupt event request according to the interrupt selection register. For more instructions on exceptions and NVIC programming, please refer to Chapter 5: Exceptions and Chapter 8: Nested Vectored Interrupt Controller in the "ARM Cortex™ -M4F Technical Reference Manual".
- 2) Programmable priority: 16 programmable priorities (4-bit interrupt priority register is used).
- 3) Nomaskable interrupt: In addition to the NMI pin as a nomaskable interrupt source, multiple system interrupt event requests can be selected independently as nomaskable interrupt requests.
- 4) Equipped with 16 external pinterrupts.
- 5) Configure various peripheral interrupt event requests, please refer to for details [10.3.2 Interrupt Event Request Sequence Number].
- 6) Equipped with 32 software interrupt event requests.
- 7) Interrupt wakes up system sleep mode and stop mode.

Input pins:

Pin name	I/O	Note
NMI	Input	Nomaskable interrupt request pin
EIRQ0	Input	External pinterrupt event request 0
EIRQ1	Input	External pinterrupt event request 1
EIRQ2	Input	External pinterrupt event request 2
EIRQ3	Input	External pinterrupt event request 3
EIRQ4	Input	External pinterrupt event request 4
EIRQ5	Input	External pinterrupt event request 5
EIRQ6	Input	External pinterrupt event request 6
EIRQ7	Input	External pinterrupt event request 7
EIRQ8	Input	External pinterrupt event request 8
EIRQ9	Input	External pinterrupt event request 9
EIRQ10	Input	External pinterrupt event request 10
EIRQ11	Input	External pinterrupt event request 11
EIRQ12	Input	External pinterrupt event request 12
EIRQ13	Input	External pinterrupt event request 13
EIRQ14	Input	External pinterrupt event request 14
EIRQ15	Input	External pinterrupt event request 15

10.2 INTC System Block Diagram

10.2.1 System Block Diagram

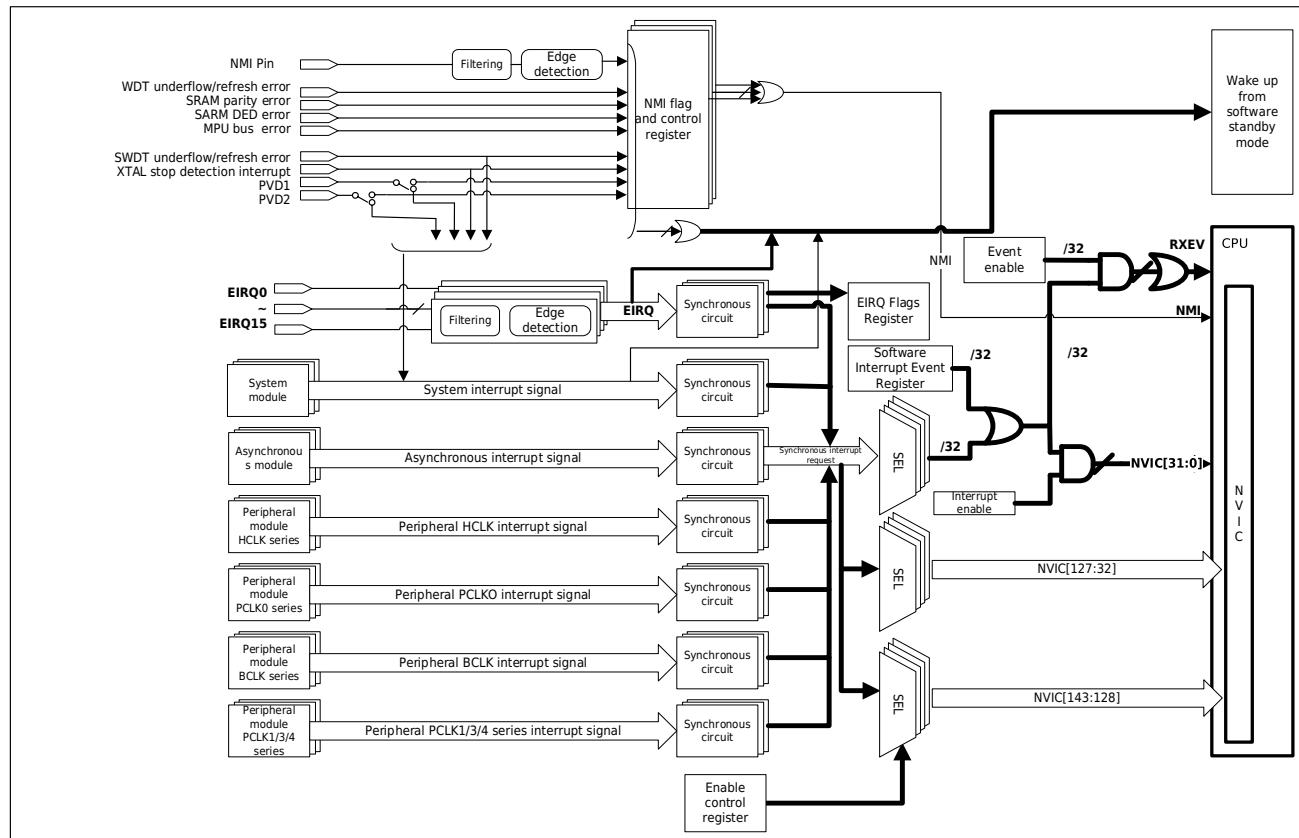


Figure 10-1 Interrupt System Block Diagram

10.3 Directional Scale

10.3.1 Interrupt Digraph

Table 10-1 Interrupt Vector Table

Address	Vect or Serial number	IRQ Serial number	Interrupt source	Note
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	INT_SEL0	Interrupt event request/software interrupt selected by register INT_SEL0.
0x0000_0044	17	1	INT_SEL1	Interrupt event request/software interrupt selected by register INT_SEL1.
0x0000_0048	18	2	INT_SEL2	Interrupt event request/software interrupt selected by register INT_SEL2.
0x0000_004C	19	3	INT_SEL3	Interrupt event request/software interrupt selected by register INT_SEL3.
0x0000_0050	20	4	INT_SEL4	Interrupt event request/software interrupt selected by register INT_SEL4.
0x0000_0054	21	5	INT_SEL5	Interrupt event request/software interrupt selected by register INT_SEL5.
0x0000_0058	22	6	INT_SEL6	Interrupt event request/software interrupt selected by register INT_SEL6.
0x0000_005C	23	7	INT_SEL7	Interrupt event request/software interrupt selected by register INT_SEL7.
0x0000_0060	24	8	INT_SEL8	Interrupt event request/software interrupt selected by register INT_SEL8.
0x0000_0064	25	9	INT_SEL9	Interrupt event request/software interrupt selected by register INT_SEL9.
0x0000_0068	26	10	INT_SEL10	Interrupt event request/software interrupt selected by register INT_SEL10.

Address	Vect or Serial number	IRQ Serial number	Interrupt source	Note
0x0000_006C	27	11	INT_SEL11	Interrupt event request/software interrupt selected by register INT_SEL11.
0x0000_0070	28	12	INT_SEL12	Interrupt event request/software interrupt selected by register INT_SEL12.
0x0000_0074	29	13	INT_SEL13	Interrupt event request/software interrupt selected by register INT_SEL13.
0x0000_0078	30	14	INT_SEL14	Interrupt event request/software interrupt selected by register INT_SEL14.
0x0000_007C	31	15	INT_SEL15	Interrupt event request/software interrupt selected by register INT_SEL15.
0x0000_0080	32	16	INT_SEL16	Interrupt event request/software interrupt selected by register INT_SEL16.
0x0000_0084	33	17	INT_SEL17	Interrupt event request/software interrupt selected by register INT_SEL17.
0x0000_0088	34	18	INT_SEL18	Interrupt event request/software interrupt selected by register INT_SEL18.
0x0000_008C	35	19	INT_SEL19	Interrupt event request/software interrupt selected by register INT_SEL19.
0x0000_0090	36	20	INT_SEL20	Interrupt event request/software interrupt selected by register INT_SEL20.
0x0000_0094	37	21	INT_SEL21	Interrupt event request/software interrupt selected by register INT_SEL21.
0x0000_0098	38	22	INT_SEL22	Interrupt event request/software interrupt selected by register INT_SEL22.
0x0000_009C	39	23	INT_SEL23	Interrupt event request/software interrupt selected by register INT_SEL23.
0x0000_00A0	40	24	INT_SEL24	Interrupt event request/software interrupt selected by register INT_SEL24.
0x0000_00A4	41	25	INT_SEL25	Interrupt event request/software interrupt selected by register INT_SEL25.
0x0000_00A8	42	26	INT_SEL26	Interrupt event request/software interrupt selected by register INT_SEL26.
0x0000_00AC	43	27	INT_SEL27	Interrupt event request/software interrupt selected by register INT_SEL27.
0x0000_00B0	44	28	INT_SEL28	Interrupt event request/software interrupt selected by register INT_SEL28.
0x0000_00B4	45	29	INT_SEL29	Interrupt event request/software interrupt selected by register INT_SEL29.
0x0000_00B8	46	30	INT_SEL30	Interrupt event request/software interrupt selected by register INT_SEL30.
0x0000_00BC	47	31	INT_SEL31	Interrupt event request/software interrupt selected by register INT_SEL31.
0x0000_00C0	48	32	INT_SEL32	Interrupt event request selected by register INT_SEL32.
0x0000_00C4	49	33	INT_SEL33	Interrupt event request selected by register INT_SEL33.
0x0000_00C8	50	34	INT_SEL34	Interrupt event request selected by register INT_SEL34.
0x0000_00CC	51	35	INT_SEL35	Interrupt event request selected by register INT_SEL35.
0x0000_00D0	52	36	INT_SEL36	Interrupt event request selected by register INT_SEL36.
0x0000_00D4	53	37	INT_SEL37	Interrupt event request selected by register INT_SEL37.
0x0000_00D8	54	38	INT_SEL38	Interrupt event request selected by register INT_SEL38.
0x0000_00DC	55	39	INT_SEL39	Interrupt event request selected by register INT_SEL39.
0x0000_00E0	56	40	INT_SEL40	Interrupt event request selected by register INT_SEL40.
0x0000_00E4	57	41	INT_SEL41	Interrupt event request selected by register INT_SEL41.

Address	Vect or Serial number	IRQ Serial number	Interrupt source	Note
0x0000_00E8	58	42	INT_SEL42	Interrupt event request selected by register INT_SEL42.
0x0000_00EC	59	43	INT_SEL43	Interrupt event request selected by register INT_SEL43.
0x0000_00F0	60	44	INT_SEL44	Interrupt event request selected by register INT_SEL44.
0x0000_00F4	61	45	INT_SEL45	Interrupt event request selected by register INT_SEL45.
0x0000_00F8	62	46	INT_SEL46	Interrupt event request selected by register INT_SEL46.
0x0000_00FC	63	47	INT_SEL47	Interrupt event request selected by register INT_SEL47.
0x0000_0100	64	48	INT_SEL48	Interrupt event request selected by register INT_SEL48.
0x0000_0104	65	49	INT_SEL49	Interrupt event request selected by register INT_SEL49.
0x0000_0108	66	50	INT_SEL50	Interrupt event request selected by register INT_SEL50.
0x0000_010C	67	51	INT_SEL51	Interrupt event request selected by register INT_SEL51.
0x0000_0110	68	52	INT_SEL52	Interrupt event request selected by register INT_SEL52.
0x0000_0114	69	53	INT_SEL53	Interrupt event request selected by register INT_SEL53.
0x0000_0118	70	54	INT_SEL54	Interrupt event request selected by register INT_SEL54.
0x0000_011C	71	55	INT_SEL55	Interrupt event request selected by register INT_SEL55.
0x0000_0120	72	56	INT_SEL56	Interrupt event request selected by register INT_SEL56.
0x0000_0124	73	57	INT_SEL57	Interrupt event request selected by register INT_SEL57.
0x0000_0128	74	58	INT_SEL58	Interrupt event request selected by register INT_SEL58.
0x0000_012C	75	59	INT_SEL59	Interrupt event request selected by register INT_SEL59.
0x0000_0130	76	60	INT_SEL60	Interrupt event request selected by register INT_SEL60.
0x0000_0134	77	61	INT_SEL61	Interrupt event request selected by register INT_SEL61.
0x0000_0138	78	62	INT_SEL62	Interrupt event request selected by register INT_SEL62.
0x0000_013C	79	63	INT_SEL63	Interrupt event request selected by register INT_SEL63.
0x0000_0140	80	64	INT_SEL64	Interrupt event request selected by register INT_SEL64.
0x0000_0144	81	65	INT_SEL65	Interrupt event request selected by register INT_SEL65.
0x0000_0148	82	66	INT_SEL66	Interrupt event request selected by register INT_SEL66.
0x0000_014C	83	67	INT_SEL67	Interrupt event request selected by register INT_SEL67.
0x0000_0150	84	68	INT_SEL68	Interrupt event request selected by register INT_SEL68.
0x0000_0154	85	69	INT_SEL69	Interrupt event request selected by register INT_SEL69.
0x0000_0158	86	70	INT_SEL70	Interrupt event request selected by register INT_SEL70.
0x0000_015C	87	71	INT_SEL71	Interrupt event request selected by register INT_SEL71.
0x0000_0160	88	72	INT_SEL72	Interrupt event request selected by register INT_SEL72.
0x0000_0164	89	73	INT_SEL73	Interrupt event request selected by register INT_SEL73.
0x0000_0168	90	74	INT_SEL74	Interrupt event request selected by register INT_SEL74.
0x0000_016C	91	75	INT_SEL75	Interrupt event request selected by register INT_SEL75.
0x0000_0170	92	76	INT_SEL76	Interrupt event request selected by register INT_SEL76.
0x0000_0174	93	77	INT_SEL77	Interrupt event request selected by register INT_SEL77.

Address	Vect or Serial number	IRQ Serial number	Interrupt source	Note
0x0000_0178	94	78	INT_SEL78	Interrupt event request selected by register INT_SEL78.
0x0000_017C	95	79	INT_SEL79	Interrupt event request selected by register INT_SEL79.
0x0000_0180	96	80	INT_SEL80	Interrupt event request selected by register INT_SEL80.
0x0000_0184	97	81	INT_SEL81	Interrupt event request selected by register INT_SEL81.
0x0000_0188	98	82	INT_SEL82	Interrupt event request selected by register INT_SEL82.
0x0000_018C	99	83	INT_SEL83	Interrupt event request selected by register INT_SEL83.
0x0000_0190	100	84	INT_SEL84	Interrupt event request selected by register INT_SEL84.
0x0000_0194	101	85	INT_SEL85	Interrupt event request selected by register INT_SEL85.
0x0000_0198	102	86	INT_SEL86	Interrupt event request selected by register INT_SEL86.
0x0000_019C	103	87	INT_SEL87	Interrupt event request selected by register INT_SEL87.
0x0000_01A0	104	88	INT_SEL88	Interrupt event request selected by register INT_SEL88.
0x0000_01A4	105	89	INT_SEL89	Interrupt event request selected by register INT_SEL89.
0x0000_01A8	106	90	INT_SEL90	Interrupt event request selected by register INT_SEL90.
0x0000_01AC	107	91	INT_SEL91	The interrupt event request selected by the register INT_SEL91.
0x0000_01B0	108	92	INT_SEL92	Interrupt event request selected by register INT_SEL92.
0x0000_01B4	109	93	INT_SEL93	Interrupt event request selected by register INT_SEL93.
0x0000_01B8	110	94	INT_SEL94	Interrupt event request selected by register INT_SEL94.
0x0000_01BC	111	95	INT_SEL95	Interrupt event request selected by register INT_SEL95.
0x0000_01C0	112	96	INT_SEL96	Interrupt event request selected by register INT_SEL96.
0x0000_01C4	113	97	INT_SEL97	Interrupt event request selected by register INT_SEL97.
0x0000_01C8	114	98	INT_SEL98	Interrupt event request selected by register INT_SEL98.
0x0000_01CC	115	99	INT_SEL99	Interrupt event request selected by register INT_SEL99.
0x0000_01D0	116	100	INT_SEL100	Interrupt event request selected by register INT_SEL100.
0x0000_01D4	117	101	INT_SEL101	Interrupt event request selected by register INT_SEL101.
0x0000_01D8	118	102	INT_SEL102	Interrupt event request selected by register INT_SEL102.
0x0000_01DC	119	103	INT_SEL103	Interrupt event request selected by register INT_SEL103.
0x0000_01E0	120	104	INT_SEL104	Interrupt event request selected by register INT_SEL104.
0x0000_01E4	121	105	INT_SEL105	Interrupt event request selected by register INT_SEL105.
0x0000_01E8	122	106	INT_SEL106	Interrupt event request selected by register INT_SEL106.
0x0000_01EC	123	107	INT_SEL107	Interrupt event request selected by register INT_SEL107.
0x0000_01F0	124	108	INT_SEL108	Interrupt event request selected by register INT_SEL108.
0x0000_01F4	125	109	INT_SEL109	Interrupt event request selected by register INT_SEL109.
0x0000_01F8	126	110	INT_SEL110	Interrupt event request selected by register INT_SEL110.
0x0000_01FC	127	111	INT_SEL111	Interrupt event request selected by register INT_SEL111.
0x0000_0200	128	112	INT_SEL112	Interrupt event request selected by register INT_SEL112.

Address	Vect or Serial number	IRQ Serial number	Interrupt source	Note
0x0000_0204	129	113	INT_SEL113	Interrupt event request selected by register INT_SEL113.
0x0000_0208	130	114	INT_SEL114	Interrupt event request selected by register INT_SEL114.
0x0000_020C	131	115	INT_SEL115	Interrupt event request selected by register INT_SEL115.
0x0000_0210	132	116	INT_SEL116	Interrupt event request selected by register INT_SEL116.
0x0000_0214	133	117	INT_SEL117	Interrupt event request selected by register INT_SEL117.
0x0000_0218	134	118	INT_SEL118	Interrupt event request selected by register INT_SEL118.
0x0000_021C	135	119	INT_SEL119	Interrupt event request selected by register INT_SEL119.
0x0000_0220	136	120	INT_SEL120	Interrupt event request selected by register INT_SEL120.
0x0000_0224	137	121	INT_SEL121	Interrupt event request selected by register INT_SEL121.
0x0000_0228	138	122	INT_SEL122	Interrupt event request selected by register INT_SEL122.
0x0000_022C	139	123	INT_SEL123	Interrupt event request selected by register INT_SEL123.
0x0000_0230	140	124	INT_SEL124	Interrupt event request selected by register INT_SEL124.
0x0000_0234	141	125	INT_SEL125	Interrupt event request selected by register INT_SEL125.
0x0000_0238	142	126	INT_SEL126	Interrupt event request selected by register INT_SEL126.
0x0000_023C	143	127	INT_SEL127	Interrupt event request selected by register INT_SEL127.
0x0000_0240	144	128	INT_VSSEL128	Register INT_VSSEL128 shares this vector with interrupt event requests selected by enable bits.
0x0000_0244	145	129	INT_VSSEL129	Register INT_VSSEL129 shares this vector with interrupt event requests selected by enable bits.
0x0000_0248	146	130	INT_VSSEL130	Register INT_VSSEL130 shares this vector with interrupt event requests selected by enable bits.
0x0000_024C	147	131	INT_VSSEL131	Register INT_VSSEL131 shares this vector with interrupt event requests selected by enable bits.
0x0000_0250	148	132	INT_VSSEL132	Register INT_VSSEL132 shares this vector with interrupt event requests selected by enable bits.
0x0000_0254	149	133	INT_VSSEL133	Register INT_VSSEL133 shares this vector with interrupt event requests selected by enable bits.
0x0000_0258	150	134	INT_VSSEL134	Register INT_VSSEL134 shares this vector with interrupt event requests selected by enable bits.
0x0000_025C	151	135	INT_VSSEL135	Register INT_VSSEL135 shares this vector with interrupt event requests selected by enable bits.
0x0000_0260	152	136	INT_VSSEL136	Register INT_VSSEL136 shares this vector with interrupt event requests selected by enable bits.
0x0000_0264	153	137	INT_VSSEL137	Register INT_VSSEL137 shares this vector with interrupt event requests selected by enable bits.
0x0000_0268	154	138	INT_VSSEL138	Register INT_VSSEL138 shares this vector with interrupt event requests selected by enable bits.
0x0000_026C	155	139	INT_VSSEL139	Register INT_VSSEL139 shares this vector with interrupt event requests selected by enable bits.
0x0000_0270	156	140	INT_VSSEL140	Register INT_VSSEL140 shares this vector with interrupt event requests selected by enable bits.
0x0000_0274	157	141	INT_VSSEL141	Register INT_VSSEL141 shares this vector with interrupt event requests selected by enable bits.
0x0000_0278	158	142	INT_VSSEL142	Register INT_VSSEL142 shares this vector with interrupt event requests selected by enable bits.
0x0000_027C	159	143	INT_VSSEL143	Register INT_VSSEL143 shares this vector with interrupt event requests selected by enable bits.

Note:

- For the selected interrupt event request number, please refer to the register description chapter.

10.3.2 Interrupt Event Request Sequence Number

An interrupt event request is generated by a peripheral. When an interrupt event request is selected by the interrupt controller as the input of the NVIC, the interrupt event request is called an interrupt source; when it is selected as an event input, it is called an event source. Peripheral interrupt event requests can also be used as a condition for MCU low-power mode returns.

Table 10-2 Interrupt Event Request Sequence Number and Selection

Number	Interrupt event sequence number	Function	Functional name	Is it optional to interrupt	Can be selected as internal trigger source	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
0	000h	PORT	PORT_EIRQ0	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[0]
1	001h		PORT_EIRQ1	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[1]
2	002h		PORT_EIRQ2	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[2]
3	003h		PORT_EIRQ3	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[3]
4	004h		PORT_EIRQ4	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[4]
5	005h		PORT_EIRQ5	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[5]
6	006h		PORT_EIRQ6	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[6]
7	007h		PORT_EIRQ7	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[7]
8	008h		PORT_EIRQ8	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[8]
9	009h		PORT_EIRQ9	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[9]
10	00Ah		PORT_EIRQ10	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[10]
11	00Bh		PORT_EIRQ11	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[11]
12	00Ch		PORT_EIRQ12	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[12]
13	00Dh		PORT_EIRQ13	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[13]
14	00Eh		PORT_EIRQ14	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[14]
15	00Fh		PORT_EIRQ15	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[15]
16	010h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[16]
17	011h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[17]
18	012h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[18]
19	013h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[19]
20	014h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[20]

Nu mb er	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
21	015h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[21]
22	016h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[22]
23	017h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[23]
24	018h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[24]
25	019h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[25]
26	01Ah	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[26]
27	01Bh	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[27]
28	01Ch	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[28]
29	01Dh	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[29]
30	01Eh	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[30]
31	01Fh	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[31]
32	020h	DMA	DMA1_TC0	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[0]
33	021h		DMA1_TC1	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[1]
34	022h		DMA1_TC2	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[2]
35	023h		DMA1_TC3	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[3]
36	024h		DMA2_TC0	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[4]
37	025h		DMA2_TC1	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[5]
38	026h		DMA2_TC2	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[6]
39	027h		DMA2_TC3	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[7]
40	028h		DMA1_BTC0	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[8]
41	029h		DMA1_BTC1	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[9]
42	02Ah		DMA1_BTC2	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[10]
43	02Bh		DMA1_BTC3	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[11]
44	02Ch		DMA2_BTC0	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[12]
45	02Dh		DMA2_BTC1	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[13]
46	02Eh		DMA2_BTC2	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[14]
47	02Fh		DMA2_BTC3	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[15]
48	030h	DMA	DMA1_ERR	✓	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[16]
49	031h		DMA2_ERR	✓	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[17]
50	032h	EFM	EFM_PEERR	✓	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[18]
51	033h		EFM_COLERR	✓	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[19]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
52	034h		EFM_OPTEND	√	√	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[20]
53	035h	USBFS	USBFS_SOF	-	√	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[21]
54	036h	QSPI	QSPI_INTR	√	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[22]
55	037h	DCU	DCU1	√	√	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[23]
56	038h		DCU2	√	√	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[24]
57	039h		DCU3	√	√	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[25]
58	03Ah		DCU4	√	√	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[26]
59	03Bh	-	-	-	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[27]
60	03Ch	-	-	-	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[28]
61	03Dh	-	-	-	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[29]
62	03Eh	-	-	-	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[30]
63	03Fh	-	-	-	-	INT_SEL0~31	INT_SEL38~43	INT_VSSEL129[31]
64	040h	Timer0 _1	TMR01_GCMA	√	√	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[0]
65	041h		TMR01_GCMB	√	√	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[1]
66	042h	Timer0 _2	TMR02_GCMA	√	√	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[2]
67	043h		TMR02_GCMB	√	√	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[3]
68	044h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[4]
69	045h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[5]
70	046h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[6]
71	047h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[7]
72	048h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[8]
73	049h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[9]
74	04Ah	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[10]
75	04Bh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[11]
76	04Ch	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[12]
77	04Dh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[13]
78	04Eh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[14]
79	04Fh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[15]
80	050h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[16]
81	051h	RTC	RTC_ALM	√	√	INT_SEL0~31	INT_SEL44~49	-
82	052h		RTC_PRD	√	√	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[18]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
83	053h		-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[19]
84	054h	XTAL32	XTAL32_STOP	✓	-	INT_SEL0~31	INT_SEL44~49	-
85	055h	XTAL	XTAL_STOP	✓	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[21]
86	056h	WKTM	WKTM_PRD	✓	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[22]
87	057h	SWDT	SWDT_REFUD_F	✓	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[23]
88	058h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[24]
89	059h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[25]
90	05Ah	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[26]
91	05Bh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[27]
92	05Ch	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[28]
93	05Dh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[29]
94	05Eh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[30]
95	05Fh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[31]
96	060h	Timer6_1	TMR61_GCMA	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[0]
97	061h		TMR61_GCMB	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[1]
98	062h		TMR61_GCMC	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[2]
99	063h		TMR61_GCMD	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[3]
100	064h		TMR61_GCME	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[4]
101	065h		TMR61_GCMF	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[5]
102	066h		TMR61_GOVF	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[6]
103	067h		TMR61_GUDF	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[7]
104	068h		TMR61_GDTE	✓	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[8]
105	069h		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[9]
106	06Ah		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[10]
107	06Bh		TMR61_SCMA	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[11]
108	06Ch		TMR61_SCMB	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[12]
109	06Dh	-	-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[13]
110	06Eh	-	-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[14]
111	06Fh	-	-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[15]
112	070h	Timer6_2	TMR62_GCMA	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[16]
113	071h		TMR62_GCMB	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[17]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
114	072h	Timer6 _3	TMR62_GCMC	√	√	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[18]
115	073h		TMR62_GCMD	√	√	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[19]
116	074h		TMR62_GCME	√	√	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[20]
117	075h		TMR62_GCMF	√	√	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[21]
118	076h		TMR62_GOVF	√	√	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[22]
119	077h		TMR62_GUDF	√	√	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[23]
120	078h		TMR62_GDTE	√	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[24]
121	079h		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[25]
122	07Ah		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[26]
123	07Bh		TMR62_SCMA	√	√	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[27]
124	07Ch		TMR62_SCMB	√	√	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[28]
125	07Dh		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[29]
126	07Eh		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[30]
127	07Fh		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[31]
128	080h		TMR63_GCMA	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[0]
129	081h		TMR63_GCMB	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[1]
130	082h		TMR63_GCMC	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[2]
131	083h		TMR63_GCMD	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[3]
132	084h		TMR63_GCME	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[4]
133	085h		TMR63_GCMF	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[5]
134	086h		TMR63_GOVF	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[6]
135	087h		TMR63_GUDF	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[7]
136	088h		TMR63_GDTE	√	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[8]
137	089h		-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[9]
138	08Ah		-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[10]
139	08Bh		TMR63_SCMA	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[11]
140	08Ch		TMR63_SCMB	√	√	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[12]
141	08Dh		-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[13]
142	08Eh		-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[14]
143	08Fh		-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[15]
144	090h		-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[16]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
145	091h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[17]
146	092h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[18]
147	093h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[19]
148	094h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[20]
149	095h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[21]
150	096h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[22]
151	097h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[23]
152	098h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[24]
153	099h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[25]
154	09Ah	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[26]
155	09Bh	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[27]
156	09Ch	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[28]
157	09Dh	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[29]
158	09Eh	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[30]
159	09Fh	-	-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[31]
160	0A0h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[0]
161	0A1h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[1]
162	0A2h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[2]
163	0A3h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[3]
164	0A4h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[4]
165	0A5h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[5]
166	0A6h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[6]
167	0A7h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[7]
168	0A8h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[8]
169	0A9h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[9]
170	0AAh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[10]
171	0ABh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[11]
172	0ACH	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[12]
173	0ADh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[13]
174	0AEh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[14]
175	0AFh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[15]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
176	0B0h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[16]
177	0B1h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[17]
178	0B2h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[18]
179	0B3h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[19]
180	0B4h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[20]
181	0B5h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[21]
182	0B6h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[22]
183	0B7h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[23]
184	0B8h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[24]
185	0B9h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[25]
186	0BAh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[26]
187	0BBh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[27]
188	0BCh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[28]
189	0BDh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[29]
190	0BEh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[30]
191	0BFh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[31]
192	0C0h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[0]
193	0C1h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[1]
194	0C2h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[2]
195	0C3h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[3]
196	0C4h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[4]
197	0C5h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[5]
198	0C6h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[6]
199	0C7h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[7]
200	0C8h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[8]
201	0C9h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[9]
202	0CAh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[10]
203	0CBh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[11]
204	0CCh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[12]
205	0CDh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[13]
206	0CEh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[14]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
207	0CFh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[15]
208	0D0h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[16]
209	0D1h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[17]
210	0D2h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[18]
211	0D3h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[19]
212	0D4h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[20]
213	0D5h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[21]
214	0D6h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[22]
215	0D7h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[23]
216	0D8h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[24]
217	0D9h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[25]
218	0DAh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[26]
219	0DBh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[27]
220	0DCh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[28]
221	0DDh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[29]
222	0DEh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[30]
223	0DFh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[31]
224	0E0h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[0]
225	0E1h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[1]
226	0E2h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[2]
227	0E3h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[3]
228	0E4h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[4]
229	0E5h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[5]
230	0E6h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[6]
231	0E7h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[7]
232	0E8h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[8]
233	0E9h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[9]
234	0EAh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[10]
235	0EBh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[11]
236	0Ec h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[12]
237	0Ed h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[13]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
238	0EEh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[14]
239	0EFh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[15]
240	0F0h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[16]
241	0F1h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[17]
242	0F2h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[18]
243	0F3h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[19]
244	0F4h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[20]
245	0F5h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[21]
246	0F6h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[22]
247	0F7h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[23]
248	0F8h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[24]
249	0F9h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[25]
250	0FAh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[26]
251	0FBh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[27]
252	0FCh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[28]
253	0FDh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[29]
254	0FEh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[30]
255	0FFh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[31]
256	100h	TimerA _1	TMRA1_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[0]
257	101h		TMRA1_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[1]
258	102h		TMRA1_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[2]
259	103h	TimerA _2	TMRA2_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[3]
260	104h		TMRA2_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[4]
261	105h		TMRA2_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[5]
262	106h	TimerA _3	TMRA3_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[6]
263	107h		TMRA3_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[7]
264	108h		TMRA3_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[8]
265	109h	TimerA _4	TMRA4_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[9]
266	10Ah		TMRA4_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[10]
267	10Bh		TMRA4_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[11]
268	10Ch	TimerA	TMRA5_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[12]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
		-5						
269	10Dh		TMRA5_UDF	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[13]
270	10Eh		TMRA5_CMP	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[14]
271	10Fh		-	-	-	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[15]
272	110h		TMRA6_OVF	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[16]
273	111h	TimerA -6	TMRA6_UDF	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[17]
274	112h		TMRA6_CMP	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[18]
275	113h	USBFS	USBFS_GLB	√	-	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[19]
276	114h	-	-	-	-	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[20]
277	115h	-	-	-	-	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[21]
278	116h	USART 1	USART1_REI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[22]
279	117h		USART1_RI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[23]
280	118h		USART1_TI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[24]
281	119h		USART1_TCI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[25]
282	11Ah		USART1_RTOI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[26]
283	11Bh	USART 2	USART2_REI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[27]
284	11Ch		USART2_RI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[28]
285	11Dh		USART2_TI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[29]
286	11Eh		USART2_TCI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[30]
287	11Fh		USART2_RTOI	√	√	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[31]
288	120h	USART 3	USART3_REI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[0]
289	121h		USART3_RI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[1]
290	122h		USART3_TI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[2]
291	123h		USART3_TCI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[3]
292	124h		USART3_RTOI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[4]
293	125h	USART 4	USART4_REI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[5]
294	126h		USART4_RI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[6]
295	127h		USART4_TI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[7]
296	128h		USART4_TCI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[8]
297	129h		USART4_RTOI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[9]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
298	12Ah	-	-	-	-	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[10]
299	12Bh	SPI1	SPI1_SPRI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[11]
300	12Ch		SPI1_SPTI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[12]
301	12Dh		SPI1_SPII	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[13]
302	12Eh		SPI1_SPEI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[14]
303	12Fh		SPI1_SPTEND	-	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[15]
304	130h	SPI2	SPI2_SPRI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[16]
305	131h		SPI2_SPTI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[17]
306	132h		SPI2_SPII	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[18]
307	133h		SPI2_SPEI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[19]
308	134h		SPI2_SPTEND	-	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[20]
309	135h	SPI3	SPI3_SPRI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[21]
310	136h		SPI3_SPTI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[22]
311	137h		SPI3_SPII	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[23]
312	138h		SPI3_SPEI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[24]
313	139h		SPI3_SPTEND	-	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[25]
314	13Ah	SPI4	SPI4_SPRI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[26]
315	13Bh		SPI4_SPTI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[27]
316	13Ch		SPI4_SPII	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[28]
317	13Dh		SPI4_SPEI	√	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[29]
318	13Eh		SPI4_SPTEND	-	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[30]
319	13Fh	AOS_ST RG	AOS_STRG*2	-	√	INT_SEL0~31	INT_SEL86~91	INT_VSSEL137[31]
320	140h	Timer4 _1	TMR41_GCMU H	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSSEL138[0]
321	141h		TMR41_GCMU L	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSSEL138[1]
322	142h		TMR41_GCMV H	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSSEL138[2]
323	143h		TMR41_GCMV L	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSSEL138[3]
324	144h		TMR41_GCMW H	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSSEL138[4]
325	145h		TMR41_GCMW L	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSSEL138[5]
326	146h		TMR41_GOVF	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSSEL138[6]
327	147h		TMR41_GUDF	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSSEL138[7]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
328	148h		TMR41_RLOU	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[8]
329	149h		TMR41_RLOV	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[9]
330	14Ah		TMR41_RLOW	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[10]
331	14Bh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[11]
332	14Ch		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[12]
333	14Dh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[13]
334	14Eh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[14]
335	14Fh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[15]
336	150h	Timer4_2	TMR42_GCMU_H	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[16]
337	151h		TMR42_GCMU_L	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[17]
338	152h		TMR42_GCMV_H	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[18]
339	153h		TMR42_GCMV_L	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[19]
340	154h		TMR42_GCMW_H	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[20]
341	155h		TMR42_GCMW_L	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[21]
342	156h		TMR42_GOVF	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[22]
343	157h		TMR42_GUDF	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[23]
344	158h		TMR42_RLOU	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[24]
345	159h		TMR42_RLOV	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[25]
346	15Ah		TMR42_RLOW	√	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[26]
347	15Bh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[27]
348	15Ch		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[28]
349	15Dh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[29]
350	15Eh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[30]
351	15Fh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[31]
352	160h	Timer4_3	TMR43_GCMU_H	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSEL139[0]
353	161h		TMR43_GCMU_L	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSEL139[1]
354	162h		TMR43_GCMV_H	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSEL139[2]
355	163h		TMR43_GCMV_L	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSEL139[3]
356	164h		TMR43_GCMW_H	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSEL139[4]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
357	165h		TMR43_GCMWL	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[5]
358	166h		TMR43_GOVF	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[6]
359	167h		TMR43_GUDF	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[7]
360	168h		TMR43_RLOU	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[8]
361	169h		TMR43_RLOV	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[9]
362	16Ah		TMR43_RLOW	√	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[10]
363	16Bh		-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[11]
364	16Ch	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[12]
365	16Dh	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[13]
366	16Eh	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[14]
367	16Fh	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[15]
368	170h	Timer4 _1 EVT	TMR41_SCMUH	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[16]
369	171h		TMR41_SCMUL	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[17]
370	172h		TMR41_SCMVH	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[18]
371	173h		TMR41_SCMVL	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[19]
372	174h		TMR41_SCMW	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[20]
373	175h		TMR41_SCMWL	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[21]
374	176h	Timer4 _2 EVT	TMR42_SCMUH	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[22]
375	177h		TMR42_SCMUL	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[23]
376	178h		TMR42_SCMVH	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[24]
377	179h		TMR42_SCMVL	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[25]
378	17Ah		TMR42_SCMW	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[26]
379	17Bh		TMR42_SCMWL	-	√	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[27]
380	17Ch	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[28]
381	17Dh	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[29]
382	17Eh	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[30]
383	17Fh	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[31]
384	180h	Timer4 _3 EVT	TMR43_SCMUH	-	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[0]
385	181h		TMR43_SCMU	-	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[1]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
		L	L					
386	182h		TMR43_SCMVH	-	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[2]
387	183h		TMR43_SCMVL	-	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[3]
388	184h		TMR43_SCMW _H	-	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[4]
389	185h		TMR43_SCMW _L	-	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[5]
390	186h	EMB	EMB_GR0	√	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[6]
391	187h		EMB_GR1	√	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[7]
392	188h		EMB_GR2	√	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[8]
393	189h		EMB_GR3	√	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[9]
394	18Ah	EVENT port	EVENT_PORT1	√	√	INT_SEL0~31	INT_SEL104~109	-
395	18Bh		EVENT_PORT2	√	√	INT_SEL0~31	INT_SEL104~109	-
396	18Ch		EVENT_PORT3	√	√	INT_SEL0~31	INT_SEL104~109	-
397	18Dh		EVENT_PORT4	√	√	INT_SEL0~31	INT_SEL104~109	-
398	18Eh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[14]
399	18Fh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[15]
400	190h	I2S1	I2S1_TXIRQOUT	√	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[16]
401	191h		I2S1_RXIRQOUT	√	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[17]
402	192h		I2S1_ERRIRQOUT	√	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[18]
403	193h	I2S2	I2S2_TXIRQOUT	√	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[19]
404	194h		I2S2_RXIRQOUT	√	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[20]
405	195h		I2S2_ERRIRQOUT	√	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[21]
406	196h	I2S3	I2S3_TXIRQOUT	√	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[22]
407	197h		I2S3_RXIRQOUT	√	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[23]
408	198h		I2S3_ERRIRQOUT	√	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[24]
409	199h	I2S4	I2S4_TXIRQOUT	√	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[25]
410	19Ah		I2S4_RXIRQOUT	√	√	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[26]
411	19Bh		I2S4_ERRIRQOUT	√	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[27]
412	19Ch	-	-	-	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[28]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
413	19Dh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[29]
414	19Eh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[30]
415	19Fh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[31]
416	1A0h	ACMP	ACMP1	✓	✓	INT_SEL0~31	INT_SEL110~115	-
417	1A1h		ACMP2	✓	✓	INT_SEL0~31	INT_SEL110~115	-
418	1A2h		ACMP3	✓	✓	INT_SEL0~31	INT_SEL110~115	-
419	1A3h	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[3]
420	1A4h	I2C1	I2C1_RXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[4]
421	1A5h		I2C1_TXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[5]
422	1A6h		I2C1_TEI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[6]
423	1A7h		I2C1_EE1	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[7]
424	1A8h	I2C2	I2C2_RXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[8]
425	1A9h		I2C2_TXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[9]
426	1AAh		I2C2_TEI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[10]
427	1ABh		I2C2_EE1	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[11]
428	1ACh	I2C3	I2C3_RXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[12]
429	1ADh		I2C3_TXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[13]
430	1AEh		I2C3_TEI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[14]
431	1AFh		I2C3_EE1	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[15]
432	1B0h	USART 1	USART1_WUPI	✓	-	INT_SEL0~31	INT_SEL110~115	-
433	1B1h	PVD	PVD_PVD1	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[17]
434	1B2h		PVD_PVD2	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[18]
435	1B3h	OTS	OTS	✓	✓	INT_SEL0~31	INT_SEL110~115	-
436	1B4h	FCM	FCMFERRI	✓	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[20]
437	1B5h		FCMMENDI	✓	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[21]
438	1B6h		FCMCOVFI	✓	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[22]
439	1B7h	WDT	WDT_REFUDF	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[23]
440	1B8h	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[24]
441	1B9h	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[25]
442	1BAh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[26]
443	1BBh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[27]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
444	1BCh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[28]
445	1BDh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[29]
446	1BEh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[30]
447	1BFh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[31]
448	1C0h	ADC1	ADC1_EOCA	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[0]
449	1C1h		ADC1_EOCB	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[1]
450	1C2h		ADC1_CHCMP	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[2]
451	1C3h		ADC1_SEQCM_P	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[3]
452	1C4h	ADC2	ADC2_EOCA	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[4]
453	1C5h		ADC2_EOCB	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[5]
454	1C6h		ADC2_CHCMP	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[6]
455	1C7h		ADC2_SEQCM_P	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[7]
456	1C8h	TRNG	TRNG_END	✓	✓	INT_SEL0~31	INT_SEL116~121	-
457	1C9h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[9]
458	1CAh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[10]
459	1CBh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[11]
460	1CCh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[12]
461	1CDh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[13]
462	1CEh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[14]
463	1CFh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[15]
464	1D0h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[16]
465	1D1h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[17]
466	1D2h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[18]
467	1D3h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[19]
468	1D4h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[20]
469	1D5h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[21]
470	1D6h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[22]
471	1D7h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[23]
472	1D8h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[24]
473	1D9h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[25]
474	1DAh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[26]

Nu m ber	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
475	1DBh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[27]
476	1DCh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[28]
477	1DDh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[29]
478	1DEh		-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[30]
479	1DFh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[31]
480	1E0h	SDIOC1	SDIOC1_DMAR	-	√	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[0]
481	1E1h		SDIOC1_DMAW	-	√	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[1]
482	1E2h		SDIOC1_SD	√	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[2]
483	1E3h	SDIOC2	SDIOC2_DMAR	-	√	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[3]
484	1E4h		SDIOC2_DMAW	-	√	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[4]
485	1E5h		SDIOC2_SD	√	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[5]
486	1E6h	CAN	CAN_INT	√	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[6]
487	1E7h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[7]
488	1E8h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[8]
489	1E9h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[9]
490	1EAh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[10]
491	1EBh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[11]
492	1Ec h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[12]
493	1EDh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[13]
494	1EEh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[14]
495	1EFh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[15]
496	1F0h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[16]
497	1F1h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[17]
498	1F2h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[18]
499	1F3h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[19]
500	1F4h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[20]
501	1F5h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[21]
502	1F6h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[22]
503	1F7h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[23]
504	1F8h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[24]
505	1F9h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[25]

Nu mb er	Inter rupt even t requ est sequ ence num ber	Functi on	Functional name	Is it opt ion al to int err upt	Can be sel ect ed as int ern al trig ger sou rce	Interrupt select register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
506	1FAh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[26]
507	1FBh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[27]
508	1FCh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[28]
509	1FDh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[29]
510	1FEh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[30]
511	1FFh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSSEL143[31]

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

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

10.4 Functional Description

10.4.1 Nonmaskable Interrupt

The nonmaskable interrupt source is as follows:

- NMI Pinterrupt
- Detecting the main generator stops terminating
- WDT underflow/refresh interrupt
- SWDT underflow/refresh interrupt
- Low voltage detection PVD1 interrupt
- Low voltage detection PVD2 interrupt
- SRAM parity error interrupt
- SRAM ECC parity error interrupt
- MPU bus error interrupt

Nonmaskable interrupt has the highest priority. Since a nonmaskable interrupt can select multiple interrupt event requests, the status of each interrupt event request can be determined by the query flag register (INT_NMIFR). Please make sure all the flag bits are "0" before the non-maskable interrupt processing exits. Nonmaskable interrupt is disabled by default and can be set by ICG or control register.

Use the register to set timing as follows:

1. When the NMI pin needs to be used, first clear the INT_NMICR.NFEN bit to "0" to disable the digital filter; set the NMITRG bit of the INT_NMICR register to select the NMI trigger edge; set the SMPCLK bit to select the sampling clock of the digital filter; set NFEN bit, enables the digital filter.
2. When using other unmasked interrupt event requests, configure the appropriate features.
3. Write "1" to each register bit of INT_NMICFR to clear the INT_NMIFR flag register bit to prevent malfunction.
4. Enable non-maskable interrupt events by setting the INT_NMIENR selection register.

Note:

- Once the corresponding bit of INT_NMIENR is set to "1", it cannot be changed unless it is reset with RESET.

The ICG setting is only for the external NMI pin, and the ICG setting is enabled by configuring the ICG1.NMIIICGENA bit of the ICG register. Set ICG1.NMITRG to select NMI trigger edge; set ICG1.SMPCLK bit to select digital filter sampling clock; set ICG1.NFEN bit to enable digital filter; set ICG1.NMIENR bit to enable NMI pinterrrupt. After the ICG is set, the register setting is invalid. For the register description of ICG1, please refer to the Initial Configuration (ICG) chapter.

10.4.2 External Pinterrupt Event Request

When you need to use an external pin to interrupt an event request, please set it according to the following procedure:

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

10.4.3 Interrupt Source Selection

This interrupt controller uses a total of 144 interrupt vectors, provides 3 kinds of interrupt event request selection methods, and meets various interrupt configuration requirements through flexible combination.

The first way

A total of 32 interrupt vectors, all interrupt event requests select 1 as the interrupt source, select through the interrupt/event selection register INT_SEL0~31, enable through the INT_IER register, and the corresponding NVIC interrupt vector is 0~31;

The second approach

A total of 96 interrupt vectors, 32 select 1 as the interrupt source, select through the interrupt selection register INT_SEL32~127, the corresponding interrupt vector is 32~127;

Third mode

There are 16 interrupt vectors in total, and 32 peripheral interrupt event requests share one interrupt vector. All peripherals can apply for interrupts, which are distinguished by peripheral flag bits, and interrupt event requests are enabled through the interrupt enable register INT_VSEL128~143, corresponding to The interrupt vector of NVIC is 128~143. Select an interrupt event request with a peripheral flag bit, for details refer to 10.3.2 Interrupt Event Request Sequence Number.

For specific interrupt vector assignment, please refer to the 10.3.2 Interrupt Event Request Sequence Number chapter.

10.4.4 Software Interrupt

The software interrupt function can generate an interrupt event request by directly writing the software interrupt control register INT_SWIER, which is controlled by the INT_IER interrupt enable bit. A total of 32 software interrupt event requests are configured, and the corresponding interrupt vectors are 0~31. For details, please refer to the 10.3.1 Interrupt Digraph chapter.

10.4.5 Interrupt/Event Selection

The interrupt selected by the interrupt selection register INT_SEL0~31 and the software interrupt share the interrupt vector No. 0~31 of the NVIC, which is controlled by the INT_IER interrupt enable register; at the same time, these interrupt event requests can also be selected as event inputs to wake up the core (WFE), Select through the event enable register INT_EVTER, the specific block diagram is as follows.

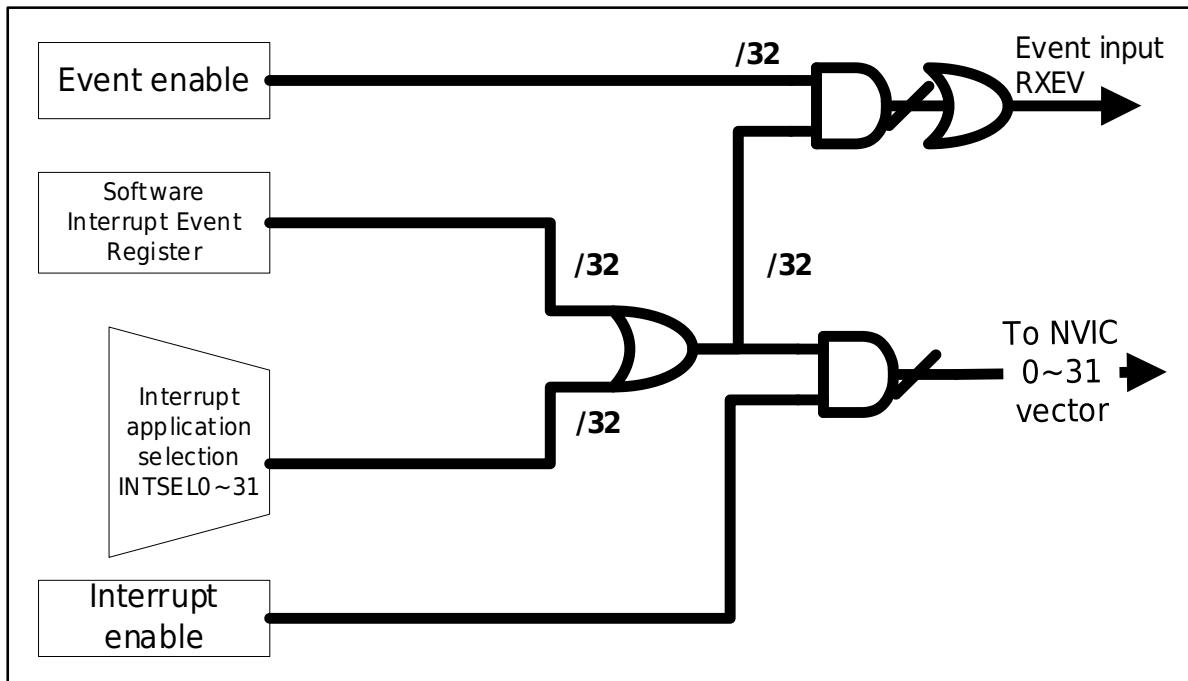


Figure 10-2 Interrupt event selection

10.4.6 WFE Wake-up Event Management

The MCU can handle events to wake up the core (WFE). The wake-up event can be generated from NVIC interrupt input or event input in two ways. The setting process is as follows.

- Wake up WFE from interrupt input from NVIC. Enable an interrupt in the control register of the peripheral, set the INT_SEL n and INT_IER registers according to the selected interrupt vector, but not enable it in the NVIC, INT_EVTER is also set to be disabled, and enable the Cortex™-M4F system control register SCR at the same time the SEVONPEND bit. When the MCU recovers from the WFE, it is necessary to clear the interrupt flag bit and the NVIC interrupt flag register of the corresponding peripheral.

The WFE stop mode entry and wake-up process is as follows:

- 1) Set the stop mode register;
- 2) Set the stop mode wake-up register INT_WUPEN;
- 3) Set the pin EIRQ input and the EIRQ control register INT_EIRQCR n ;
- 4) Select INT_SEL n to select the corresponding EIRQ interrupt event request sequence number;

- 5) The INT_IER register is to enable the corresponding interrupt event request, and INT_EVTER is not enabled;
 - 6) Set SEVONPEND bit in SCR to "1";
 - 7) Do the following to ensure that the system enters stop mode:
 - _SEV(); Set up internal event register
 - _WFE(); Clear event register
 - _WFE(); System enters stop mode
 - 8) Waiting for the selected interrupt event request to occur, the system wakes up from stop mode but does not enter the interrupt processing subroutine.
- Wake up WFE from event input from NVIC. Configure an interrupt event request as an event input and enable it through the event enable register INT_EVTER. When the CPU resumes from WFE, the interrupt flag bit of the corresponding peripheral needs to be cleared.
- The WFE stop mode wake-up process is as follows:
- 1) Set the stop mode register;
 - 2) Set the stop mode wake-up register INT_WUPEN;
 - 3) Set the pin EIRQ input and the EIRQ control register INT_EIRQCRn;
 - 4) Select INT_SELn to select the corresponding EIRQ interrupt event request sequence number;
 - 5) The INT_IER register is in a non-enabled state, and INT_EVTER enables the corresponding interrupt event request;
 - 6) Do the following to ensure that the system enters stop mode:
 - _SEV(); Set up internal event register
 - _WFE(); Clear event register
 - _WFE(); System enters stop mode
 - 7) Waiting for the selected interrupt event request to occur, the system wakes up from stop mode but does not enter the interrupt processing subroutine.

10.4.7 Digital Filter

For NMI and EIRQx ($x=0\sim15$) pin event inputs, digital filters can be selected for noise filtering. The sampling clock of the filter is PCLK3, and the input signal less than 3 filter cycles will be filtered. The sequence of work is as follows:

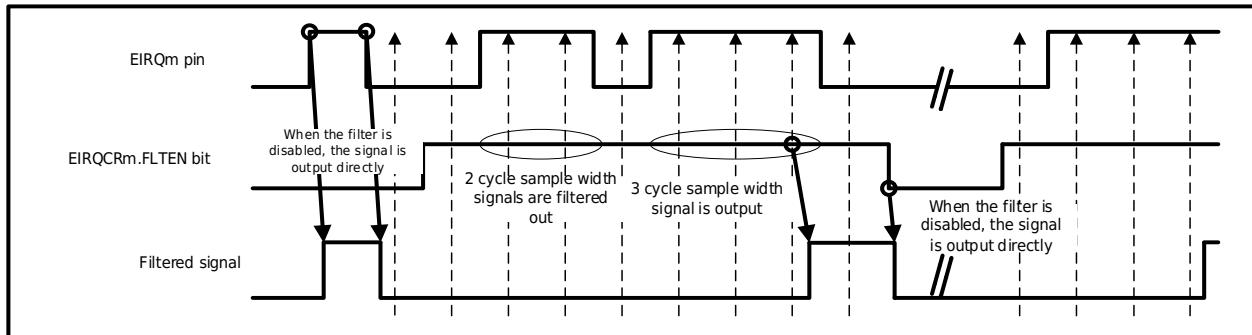


Figure 10-3 Schematic diagram of digital filter operation

Before entering stop mode, please set INT_NMICR.NFEN and INT_EIRQCRm.EFEN to disable the digital filter. The digital filter is enabled after returning from the stop mode. The setting process is as follows:

- 1) Set system stop mode register
- 2) Configuration stop mode wakeup interrupt
- 3) Stop digital filter
- 4) Execute WFI. The system enters stop mode.

10.4.8 Low Power Mode Return

10.4.8.1 Dormant Mode Return

When you select the event interrupt source as the sleep mode return condition, you need to set the following:

- Select event as CPU interrupt source
- Enable control register in NVIC
- INT_NMIENR enable register needs to be set if nonmaskable interrupt needs to be used

10.4.8.2 Stop Mode Return

You can select either a nonmaskable event interrupt source or a shielded event source selected inT_WUPENNRegister as the return condition for the stop mode.

The following settings are required to return from the stop mode:

- Select the event interrupt source as the return condition for stopping mode
 - A. For non-maskable interrupts, set by INT_NMIENR enable register NMI pinterrupt, SWDT, PVD1, PVD2 can wake up stop mode.
 - B. For maskable interrupts, it is set by the INT_WUPEN enable register.
- Select event as CPU interrupt source
- Enable control register in NVIC

The unselected EIRQ pin will not be detected because the clock is turned off.

10.4.8.3 Power-down Mode Return

The return of power-down mode can be returned by the conditions shown in Chapter 5 Power Control (PWC), RES# pin reset, power-on reset and low voltage detection 0 conditions. After returning, the CPU enters the reset interrupt processing. For details, please refer to the [Power Control (PWC)]section.

10.4.8.4 Non-maskable Interrupt and WFI Instruction

Before the WFI instruction is executed, verify that the nonmaskable interrupt flag register INT_NMIFR has all the state bits' 0'.

10.4.9 Internal Trigger Event

ADC, Timer, DMA, PORT, DCU and other peripheral peripherals can not only start working by configuring the registers of the module itself, but also trigger other modules to start working by writing the peripheral trigger event register. To use the internal trigger event, you need to clear the PWC_FCG0.AOS bit to enable the peripheral circuit trigger function. For the specific setting process, please refer to the chapters of each module.

10.5 Register Description

The following table is an INTRegister list.

INTC Base Address: 0x4005_1000

Register name	Symbol	Offset address	Bit width	Reset value
NMI pin non-maskable interrupt control register	INT_NMICR	0x0000	32	0x0000_0000
Nomaskable interrupt enable register	INT_NMIENR	0x0004	32	0x0000_0000
Nomaskable interrupt flag register	INT_NMIFR	0x0008	32	0x0000_0000
Nomaskable interrupt flag clear register	INT_NMICFR	0x000C	32	0x0000_0000
External pinterrupt control register 0	INT_EIRQCR0	0x0010	32	0x0000_0000
External pinterrupt control register 1	INT_EIRQCR1	0x0014	32	0x0000_0000
External pinterrupt control register 2	INT_EIRQCR2	0x0018	32	0x0000_0000
External pinterrupt control register 3	INT_EIRQCR3	0x001C	32	0x0000_0000
External pinterrupt control register 4	INT_EIRQCR4	0x0020	32	0x0000_0000
External pinterrupt control register 5	INT_EIRQCR5	0x0024	32	0x0000_0000
External pinterrupt control register 6	INT_EIRQCR6	0x0028	32	0x0000_0000
External pinterrupt control register 7	INT_EIRQCR7	0x002C	32	0x0000_0000
External pinterrupt control register 8	INT_EIRQCR8	0x0030	32	0x0000_0000
External pinterrupt control register 9	INT_EIRQCR9	0x0034	32	0x0000_0000
External Pinterrupt Control Register 10	INT_EIRQCR10	0x0038	32	0x0000_0000
External pinterrupt control register 11	INT_EIRQCR11	0x003C	32	0x0000_0000
External pinterrupt control register 12	INT_EIRQCR12	0x0040	32	0x0000_0000
External pinterrupt control register 13	INT_EIRQCR13	0x0044	32	0x0000_0000
External pinterrupt control register 14	INT_EIRQCR14	0x0048	32	0x0000_0000
External pinterrupt control register 15	INT_EIRQCR15	0x004C	32	0x0000_0000
Stop Mode Wakeup Event Enable Register	INT_WUPEN	0x0050	32	0x0000_0000
External Pinterrupt Flag Register	INT_EIFR	0x0054	32	0x0000_0000
External pinterrupt flag clear register	INT_EICFR	0x0058	32	0x0000_0000
Interrupt/Event Select Register 0	INT_SEL0	0x005C	32	0x0000_01FF
Interrupt/Event Select Register 1	INT_SEL1	0x0060	32	0x0000_01FF
Interrupt/Event Select Register 2	INT_SEL2	0x0064	32	0x0000_01FF
Interrupt/Event Select Register 3	INT_SEL3	0x0068	32	0x0000_01FF
Interrupt/Event Select Register 4	INT_SEL4	0x006C	32	0x0000_01FF
Interrupt/Event Select Register 5	INT_SEL5	0x0070	32	0x0000_01FF
Interrupt/Event Select Register 6	INT_SEL6	0x0074	32	0x0000_01FF
Interrupt/Event Select Register 7	INT_SEL7	0x0078	32	0x0000_01FF
Interrupt/Event Select Register 8	INT_SEL8	0x007C	32	0x0000_01FF
Interrupt/Event Select Register 9	INT_SEL9	0x0080	32	0x0000_01FF

Register name	Symbol	Offset address	Bit width	Reset value
Interrupt/Event Select Register 10	INT_SEL10	0x0084	32	0x0000_01FF
Interrupt/Event Select Register 11	INT_SEL11	0x0088	32	0x0000_01FF
Interrupt/Event Select Register 12	INT_SEL12	0x008C	32	0x0000_01FF
Interrupt/Event Select Register 13	INT_SEL13	0x0090	32	0x0000_01FF
Interrupt/Event Select Register 14	INT_SEL14	0x0094	32	0x0000_01FF
Interrupt/Event Select Register 15	INT_SEL15	0x0098	32	0x0000_01FF
Interrupt/Event Select Register 16	INT_SEL16	0x009C	32	0x0000_01FF
Interrupt/Event Select Register 17	INT_SEL17	0x00A0	32	0x0000_01FF
Interrupt/Event Select Register 18	INT_SEL18	0x00A4	32	0x0000_01FF
Interrupt/Event Select Register 19	INT_SEL19	0x00A8	32	0x0000_01FF
Interrupt/Event Select Register 20	INT_SEL20	0x00AC	32	0x0000_01FF
Interrupt/Event Select Register 21	INT_SEL21	0x00B0	32	0x0000_01FF
Interrupt/Event Select Register 22	INT_SEL22	0x00B4	32	0x0000_01FF
Interrupt/Event Select Register 23	INT_SEL23	0x00B8	32	0x0000_01FF
Interrupt/Event Select Register 24	INT_SEL24	0x00BC	32	0x0000_01FF
Interrupt/Event Select Register 25	INT_SEL25	0x00C0	32	0x0000_01FF
Interrupt/Event Select Register 26	INT_SEL26	0x00C4	32	0x0000_01FF
Interrupt/Event Select Register 27	INT_SEL27	0x00C8	32	0x0000_01FF
Interrupt/Event Select Register 28	INT_SEL28	0x00CC	32	0x0000_01FF
Interrupt/Event Select Register 29	INT_SEL29	0x00D0	32	0x0000_01FF
Interrupt/Event Select Register 30	INT_SEL30	0x00D4	32	0x0000_01FF
Interrupt/Event Select Register 31	INT_SEL31	0x00D8	32	0x0000_01FF
Interrupt Select Register 32	INT_SEL32	0x00DC	32	0x0000_01FF
Interrupt Select Register 33	INT_SEL33	0x00E0	32	0x0000_01FF
Interrupt Select Register 34	INT_SEL34	0x00E4	32	0x0000_01FF
Interrupt Select Register 35	INT_SEL35	0x00E8	32	0x0000_01FF
Interrupt Select Register 36	INT_SEL36	0x00EC	32	0x0000_01FF
Interrupt Select Register 37	INT_SEL37	0x00F0	32	0x0000_01FF
Interrupt Select Register 38	INT_SEL38	0x00F4	32	0x0000_01FF
Interrupt Select Register 39	INT_SEL39	0x00F8	32	0x0000_01FF
Interrupt Select Register 40	INT_SEL40	0x00FC	32	0x0000_01FF
Interrupt Select Register 41	INT_SEL41	0x0100	32	0x0000_01FF
Interrupt Select Register 42	INT_SEL42	0x0104	32	0x0000_01FF
Interrupt Select Register 43	INT_SEL43	0x0108	32	0x0000_01FF
Interrupt Select Register 44	INT_SEL44	0x010C	32	0x0000_01FF
Interrupt Select Register 45	INT_SEL45	0x0110	32	0x0000_01FF
Interrupt Select Register 46	INT_SEL46	0x0114	32	0x0000_01FF
Interrupt Select Register 47	INT_SEL47	0x0118	32	0x0000_01FF

Register name	Symbol	Offset address	Bit width	Reset value
Interrupt Select Register 48	INT_SEL48	0x011C	32	0x0000_01FF
Interrupt Select Register 49	INT_SEL49	0x0120	32	0x0000_01FF
Interrupt Select Register 50	INT_SEL50	0x0124	32	0x0000_01FF
Interrupt Select Register 51	INT_SEL51	0x0128	32	0x0000_01FF
Interrupt Select Register 52	INT_SEL52	0x012C	32	0x0000_01FF
Interrupt Select Register 53	INT_SEL53	0x0130	32	0x0000_01FF
Interrupt Select Register 54	INT_SEL54	0x0134	32	0x0000_01FF
Interrupt Select Register 55	INT_SEL55	0x0138	32	0x0000_01FF
Interrupt Select Register 56	INT_SEL56	0x013C	32	0x0000_01FF
Interrupt Select Register 57	INT_SEL57	0x0140	32	0x0000_01FF
Interrupt Select Register 58	INT_SEL58	0x0144	32	0x0000_01FF
Interrupt Select Register 59	INT_SEL59	0x0148	32	0x0000_01FF
Interrupt Select Register 60	INT_SEL60	0x014C	32	0x0000_01FF
Interrupt Select Register 61	INT_SEL61	0x0150	32	0x0000_01FF
Interrupt Select Register 62	INT_SEL62	0x0154	32	0x0000_01FF
Interrupt Select Register 63	INT_SEL63	0x0158	32	0x0000_01FF
Interrupt Select Register 64	INT_SEL64	0x015C	32	0x0000_01FF
Interrupt Select Register 65	INT_SEL65	0x0160	32	0x0000_01FF
Interrupt Select Register 66	INT_SEL66	0x0164	32	0x0000_01FF
Interrupt Select Register 67	INT_SEL67	0x0168	32	0x0000_01FF
Interrupt Select Register 68	INT_SEL68	0x016C	32	0x0000_01FF
Interrupt Select Register 69	INT_SEL69	0x0170	32	0x0000_01FF
Interrupt Select Register 70	INT_SEL70	0x0174	32	0x0000_01FF
Interrupt Select Register 71	INT_SEL71	0x0178	32	0x0000_01FF
Interrupt Select Register 72	INT_SEL72	0x017C	32	0x0000_01FF
Interrupt Select Register 73	INT_SEL73	0x0180	32	0x0000_01FF
Interrupt Select Register 74	INT_SEL74	0x0184	32	0x0000_01FF
Interrupt Select Register 75	INT_SEL75	0x0188	32	0x0000_01FF
Interrupt Select Register 76	INT_SEL76	0x018C	32	0x0000_01FF
Interrupt Select Register 77	INT_SEL77	0x0190	32	0x0000_01FF
Interrupt Select Register 78	INT_SEL78	0x0194	32	0x0000_01FF
Interrupt Select Register 79	INT_SEL79	0x0198	32	0x0000_01FF
Interrupt Select Register 80	INT_SEL80	0x019C	32	0x0000_01FF
Interrupt Select Register 81	INT_SEL81	0x01A0	32	0x0000_01FF
Interrupt Select Register 82	INT_SEL82	0x01A4	32	0x0000_01FF
Interrupt Select Register 83	INT_SEL83	0x01A8	32	0x0000_01FF
Interrupt Select Register 84	INT_SEL84	0x01AC	32	0x0000_01FF
Interrupt Select Register 85	INT_SEL85	0x01B0	32	0x0000_01FF

Register name	Symbol	Offset address	Bit width	Reset value
Interrupt Select Register 86	INT_SEL86	0x01B4	32	0x0000_01FF
Interrupt Select Register 87	INT_SEL87	0x01B8	32	0x0000_01FF
Interrupt Select Register 88	INT_SEL88	0x01BC	32	0x0000_01FF
Interrupt Select Register 89	INT_SEL89	0x01C0	32	0x0000_01FF
Interrupt Select Register 90	INT_SEL90	0x01C4	32	0x0000_01FF
Interrupt Select Register 91	INT_SEL91	0x01C8	32	0x0000_01FF
Interrupt Select Register 92	INT_SEL92	0x01CC	32	0x0000_01FF
Interrupt Select Register 93	INT_SEL93	0x01D0	32	0x0000_01FF
Interrupt Select Register 94	INT_SEL94	0x01D4	32	0x0000_01FF
Interrupt Select Register 95	INT_SEL95	0x01D8	32	0x0000_01FF
Interrupt Select Register 96	INT_SEL96	0x01DC	32	0x0000_01FF
Interrupt Select Register 97	INT_SEL97	0x01E0	32	0x0000_01FF
Interrupt Select Register 98	INT_SEL98	0x01E4	32	0x0000_01FF
Interrupt Select Register 99	INT_SEL99	0x01E8	32	0x0000_01FF
Interrupt Select Register 100	INT_SEL100	0x01EC	32	0x0000_01FF
Interrupt Select Register 101	INT_SEL101	0x01F0	32	0x0000_01FF
Interrupt Select Register 102	INT_SEL102	0x01F4	32	0x0000_01FF
Interrupt Select Register 103	INT_SEL103	0x01F8	32	0x0000_01FF
Interrupt Select Register 104	INT_SEL104	0x01FC	32	0x0000_01FF
Interrupt Select Register 105	INT_SEL105	0x0200	32	0x0000_01FF
Interrupt Select Register 106	INT_SEL106	0x0204	32	0x0000_01FF
Interrupt Select Register 107	INT_SEL107	0x0208	32	0x0000_01FF
Interrupt Select Register 108	INT_SEL108	0x020C	32	0x0000_01FF
Interrupt Select Register 109	INT_SEL109	0x0210	32	0x0000_01FF
Interrupt Select Register 110	INT_SEL110	0x0214	32	0x0000_01FF
Interrupt Select Register 111	INT_SEL111	0x0218	32	0x0000_01FF
Interrupt Select Register 112	INT_SEL112	0x021C	32	0x0000_01FF
Interrupt Select Register 113	INT_SEL113	0x0220	32	0x0000_01FF
Interrupt Select Register 114	INT_SEL114	0x0224	32	0x0000_01FF
Interrupt Select Register 115	INT_SEL115	0x0228	32	0x0000_01FF
Interrupt Select Register 116	INT_SEL116	0x022C	32	0x0000_01FF
Interrupt Select Register 117	INT_SEL117	0x0230	32	0x0000_01FF
Interrupt Select Register 118	INT_SEL118	0x0234	32	0x0000_01FF
Interrupt Select Register 119	INT_SEL119	0x0238	32	0x0000_01FF
Interrupt Select Register 120	INT_SEL120	0x023C	32	0x0000_01FF
Interrupt Select Register 121	INT_SEL121	0x0240	32	0x0000_01FF
Interrupt Select Register 122	INT_SEL122	0x0244	32	0x0000_01FF
Interrupt Select Register 123	INT_SEL123	0x0248	32	0x0000_01FF

Register name	Symbol	Offset address	Bit width	Reset value
Interrupt Select Register 124	INT_SEL124	0x024C	32	0x0000_01FF
Interrupt Select Register 125	INT_SEL125	0x0250	32	0x0000_01FF
Interrupt Select Register 126	INT_SEL126	0x0254	32	0x0000_01FF
Interrupt Select Register 127	INT_SEL127	0x0258	32	0x0000_01FF
Vector Shared Interrupt Select Register 128	INT_VSSEL128	0x025C	32	0x0000_0000
Vector Shared Interrupt Select Register 129	INT_VSSEL129	0x0260	32	0x0000_0000
Vector Shared Interrupt Select Register 130	INT_VSSEL130	0x0264	32	0x0000_0000
Vector Shared Interrupt Select Register 131	INT_VSSEL131	0x0268	32	0x0000_0000
Vector Shared Interrupt Select Register 132	INT_VSSEL132	0x026C	32	0x0000_0000
Vector Shared Interrupt Select Register 133	INT_VSSEL133	0x0270	32	0x0000_0000
Vector Shared Interrupt Select Register 134	INT_VSSEL134	0x0274	32	0x0000_0000
Vector Shared Interrupt Select Register 135	INT_VSSEL135	0x0278	32	0x0000_0000
Vector Shared Interrupt Select Register 136	INT_VSSEL136	0x027C	32	0x0000_0000
Vector Shared Interrupt Select Register 137	INT_VSSEL137	0x0280	32	0x0000_0000
Vector Shared Interrupt Select Register 138	INT_VSSEL138	0x0284	32	0x0000_0000
Vector Shared Interrupt Select Register 139	INT_VSSEL139	0x0288	32	0x0000_0000
Vector Shared Interrupt Select Register 140	INT_VSSEL140	0x028C	32	0x0000_0000
Vector Shared Interrupt Select Register 141	INT_VSSEL141	0x0290	32	0x0000_0000
Vector Shared Interrupt Select Register 142	INT_VSSEL142	0x0294	32	0x0000_0000
Vector Shared Interrupt Select Register 143	INT_VSSEL143	0x0298	32	0x0000_0000
Software Interrupt Event Register	INT_SWIER	0x029C	32	0x0000_0000
Event Enable Register	INT_EVTER	0x02A0	32	0x0000_0000
Interrupt enable register	INT_IER	0x02A4	32	0xFFFF_FFFF

10.5.1 NMI Interrupt Control Register (INT_NMICR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Bit	Marking	Place name	Function										Read and write		
b31~b8	Reserved	-	Read as "0", write as "0"										R/W		
b7	NFEN	NMI digital filter enable	0: Disable digital filter function 1: Licensed digital filter function										R/W		
b7~b6	Reserved	-	Read as "0", write as "0"										R/W		
b5~b4	SMPCLK[1:0]	Filter sampling clock selection	0 0: PCLK3 0 1: PCLK3/8 1 0: PCLK3/32 1 1:: PCLK3/64										R/W		
b3~b1	Reserved	-	Read as "0", write as "0"										R/W		
b0	NMITRG	Trigger edge selection	0: Falling edge 1: Rising edge										R/W		

10.5.2 NMI Interrupt Enable Register (INT_NMIENR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	WDT ENR	BUS MEN R	REC CENR	REPE NR	-	-	XTAL STPE NR	-	PVD 2ENR	PVD 1ENR	SWD TENR	NMIE NR
Bit	Marking		Place name				Function						Read and write		
b31~b13	Reserved		-				Read as "0", write as "0"						R/W		
b12	Reserved		-				Read as "0", write as "0"						R/W		
b11	WDTENR		WDT underflow/refresh interrupt selection				0: Disable interrupt as a nomaskable interrupt source 1: Choose interrupt as nomaskable interrupt source						R/W		
b10	BUSMENR		MPU main bus error interrupt selection				0: Disable interrupt as a nomaskable interrupt source 1: Choose interrupt as nomaskable interrupt source						R/W		
b9	RECCENR		SRAM ECC check error interrupt selection				0: Disable interrupt as a nomaskable interrupt source 1: Choose interrupt as nomaskable interrupt source						R/W		
b8	REPENR		SRAM parity error interrupt select				0: Disable interrupt as a nomaskable interrupt source 1: Choose interrupt as nomaskable interrupt source						R/W		
b7	Reserved		-				Read as "0", write as "0"						R/W		
b6	Reserved		-				Read as "0", write as "0"						R/W		
b5	XATLSTPENR		Detect Master Oscillator Stop Interrupt Selection				0: Disable interrupt as a nomaskable interrupt source 1: Choose interrupt as nomaskable interrupt source						R/W		
b4	Reserved		-				Read as "0", write as "0"						R		
b3	PVD2ENR		Low voltage detection PVD2 interrupt selection				0: Disable interrupt as a nomaskable interrupt source 1: Choose interrupt as nomaskable interrupt source						R/W		
b2	PVD1ENR		Low voltage detection PVD1 interrupt selection				0: Disable interrupt as a nomaskable interrupt source 1: Choose interrupt as nomaskable interrupt source						R/W		
b1	SWDTENR		SWDT underflow/refresh interrupt selection				0: Disable interrupt as a nomaskable interrupt source 1: Choose interrupt as nomaskable interrupt source						R/W		
b0	NMIENR		NMI Pininterrupt Selection				0: Disable interrupt as a nomaskable interrupt source 1: Choose interrupt as nomaskable interrupt source						R/W		

10.5.3 NMI Flag Register (INT_NMIFR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	WDT FR	BUS MFR	REC CFR	REP F R	-	-	XTAL STPF R		PVD 2FR	PVD 1FR	SWD TFR	NMIF R
Bit	Marking		Place name				Function						Read and write		
b31~b13	Reserved		-				Read as "0", write as "0"						R		
b12	Reserved		-				Read as "0", write as "0"						R		
b11	WDTFR		WDT underflow/refresh interrupt flag				0: No WDT underflow/refresh application occurs 1: WDT underflow/refresh request occurs						R		
b10	BUSMFR		MPU main bus error interrupt flag				0: No MPU master bus error application occurs 1: MPU main bus error application occurs						R		
b9	RECCFR		SRAM DED parity error interrupt flag				0: SRAM DED verification error application does not occur 1: SRAM DED verification error application occurs						R		
b8	REPFR		SRAM parity error interrupt flag				0: No SRAM parity error application occurs 1: SRAM parity error application occurs						R		
b7	Reserved		-				Read as "0", write as "0"						R		
b6	Reserved		-				Read as "0", write as "0"						R		
b5	XTALSTPFR		Detect main oscillator stop interrupt flag				0: No detection occurs Main oscillator stop application 1: Occurrence detection main oscillator stop application						R		
b4	Reserved		-				Read as "0", write as "0"						R		
b3	PVD2FR		Low voltage detection PVD2 interrupt flag				0: No low voltage detection occurs PVD2 application 1: PVD2 application for low voltage detection						R		
b2	PVD1FR		Low voltage detection PVD1 interrupt flag				0: No low voltage detection occurs PVD1 application 1: PVD1 application for low voltage detection						R		
b1	SWDTFR		SWDT underflow/refresh interrupt flag				0: No SWDT underflow/refresh request occurs 1: SWDT underflow/refresh request occurs						R		
b0	NMIFR		NMI Pin interrupt Mark				0: No NMI pin application occurs 1: NMI pin application occurs						R		

10.5.4 NMI Clear Flag Register (INT_NMICFR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	WDT CFR	BUS MCFR	REC CCFR	REPC FR	-	-	XTAL STPC FR	-	PVD 2CFR	PVD 1CFR	SWD TCFR	NMIC FR
Bit	Marking		Place name				Function				Read and write				
b31~b13	Reserved		-				Read as "0", write as "0"				R/W[注1]				
b12	Reserved		-				Read as "0", write as "0"				R/W[注1]				
b11	WDTCFR		WDT underflow/refresh interrupt flag clear				0: Invalid 1: Clear WDT underflow/refresh flag				R/W[注1]				
b10	BUSMCFR		MPU main bus error interrupt flag cleared				0: Invalid 1: Clear the MPU main bus error flag				R/W[注1]				
b9	RECCCFR		SRAM DED parity error interrupt flag clear				0: Invalid 1: Clear the SRAM DED verification error flag				R/W[注1]				
b8	REPCFR		SRAM parity error interrupt flag cleared				0: Invalid 1: Clear the SRAM parity error flag				R/W[注1]				
b7	Reserved		-				Read as "0", write as "0"				R/W[注1]				
b6	Reserved		-				Read as "0", write as "0"				R/W[注1]				
b5	XTALSTPCFR		Detect main oscillator stop interrupt flag clear				0: Invalid 1: Clear the detection of the stop flag of the main oscillator				R/W[注1]				
b4	Reserved		-				Read as "0", write as "0"				R				
b3	PVD2CFR		Low voltage detection PVD2 interrupt flag clear				0: Invalid 1: Clear low voltage detection PVD2 flag				R/W[注1]				
b2	PVD1CFR		Low voltage detection PVD1 interrupt flag clear				0: Invalid 1: Clear low voltage detection PVD1 flag				R/W[注1]				
b1	SWDTCFR		SWDT underflow/refresh interrupt flag cleared				0: Invalid 1: Clear SWDT underflow/refresh flag				R/W[注1]				
b0	NMICFR		NMI Pin interrupt Mark Removal				0: Invalid 1: Clearing NMI Pin Marks				R/W[注1]				

[Note 1] Only "1" can be written, and it will be "0" when read.

10.5.5 EIRQ Control Register (INT_EIRQCRx) (x=0~15)

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
-	-	-	-	-	-	-	-	-	EFEN	-	EISMPCLK [1:0]	-	-	EIRQTRG[1:0]	
Bit	Marking		Place name				Function						Read and write		
b31~b10	Reserved		-				Read as "0", write as "0"						R/W		
b9	Reserved		-				Read as "0", write as "0"						R/W		
b8	Reserved		-				Read as "0", write as "0"						R/W		
b7	EFEN		EIRQ digital filter enable				0: Disable digital filter function 1: Allow digital filter function						R/W		
b6	Reserved		-				Read as "0", write as "0"						R/W		
b5~b4	EISMPCLK[1:0]		Filter sampling clock selection				0 0: PCLK3 0 1: PCLK3/8 1 0: PCLK3/32 1 1:: PCLK3/64						R/W		
b3~b2	Reserved		-				Read as "0", write as "0"						R/W		
b1~b0	EIRQTRG[1:0]		Trigger selection				0 0: Falling edge 01: Rising edge 1 0: Double edge 1 1: Low level						R/W		

10.5.6 EIRQ Flag Register (INT_EIFR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
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	Marking	Place name	Function	Read and write											
b31~b10	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	EIFR	EIFR flag	0: EIRQ event did not occur, or write EIFCR bit to clear the bit 1: The selected EIRQ event occurs	R											

10.5.7 EIRQ Flag Clear Register (INT_EICFR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EICFR[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b10	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	EICFR	EIFR clear bit	0: Writing "0" is invalid 1: Write "1" to clear the INT_EIFR register	R/W											

10.5.8 Interrupt Source Select Register (INT_SEL0~31)

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
INTSEL[8:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b9	Reserved	-	Read as "0", write as "0"	R/W											
b8~b0	INTSEL[8:0]	Interrupt Event Request Selection	9'h000~9'h1FE: 10.3.2 The event corresponding to the sequence number of the interrupt event request	R/W											

10.5.9 Interrupt Source Select Register (INT_SEL32~127)

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16												
Reserved																											
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0												
-	-	-	-	-	-	-	-	INTSEL[8:0]																			
Bit	Marking	Place name	Function												Read and write												
b31~b9	Reserved	-	Read as "0", write as "0"												R/W												
Select the event corresponding to the sequence number of the interrupt event request, and the specific correspondence is as follows:																											
INT_SEL32~INT_SEL37: Select the interrupt event request corresponding to 9'h000~9'h01F, other selections are invalid.																											
INT_SEL38~INT_SEL43: Select the interrupt event request corresponding to 9'h020~9'h03F, other selections are invalid.																											
INT_SEL44~INT_SEL49: Select the interrupt event request corresponding to 9'h040~9'h05F, other selections are invalid.																											
INT_SEL50~INT_SEL55: Select the interrupt event request corresponding to 9'h060~9'h07F, other selections are invalid.																											
INT_SEL56~INT_SEL61: Select the interrupt event request corresponding to 9'h080~9'h09F, other selections are invalid.																											
INT_SEL62~INT_SEL67: Select the interrupt event request corresponding to 9'h0A0~9'h0BF, other selections are invalid.																											
INT_SEL68~INT_SEL73: Select the interrupt event request corresponding to 9'h0C0~9'h0DF, other selections are invalid.																											
INT_SEL74~INT_SEL79: Select the interrupt event request corresponding to 9'h0E0~9'h0FF, other selections are invalid.																											
INT_SEL80~INT_SEL85: Select the interrupt event request corresponding to 9'h100~9'h11F, other selections are invalid.																											
INT_SEL86~INT_SEL91: Select the interrupt event request corresponding to 9'h120~9'h13F, other selections are invalid.																											
INT_SEL92~INT_SEL97: Select the interrupt event request corresponding to 9'h140~9'h15F, other selections are invalid.																											
INT_SEL98~INT_SEL103: Select the interrupt event request corresponding to 9'h160~9'h17F, other selections are invalid.																											
INT_SEL104~INT_SEL109: Select the interrupt event request corresponding to 9'h180~9'h19F, other selections are invalid.																											
INT_SEL110~INT_SEL115: Select the interrupt event request corresponding to 9'h1A0~9'h1BF, other selections are invalid.																											
INT_SEL116~INT_SEL121: Select the interrupt event request corresponding to 9'h1C0~9'h1DF, other selections are invalid.																											
INT_SEL122~INT_SEL127: Select the interrupt event request corresponding to 9'h1E0~9'h1FF, other selections are invalid.																											
b8~b0 INTSEL[8:0] Interrupt Event Request Selection R/W																											

10.5.10 Vector Sharing Interrupt Source Select Register (INT_VSEL128~143)

Reset value: 0x0000_0000

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	Marking	Place name	Function	Read and write
b31~b0	VSEL[31:0]	Interrupt Enable	INT_VSEL128: Each bit enables the interrupt event request corresponding to 9'h000~9'h01F. INT_VSEL129: Each bit enables the interrupt event request corresponding to 9'h020~9'h03F. INT_VSEL130: Each bit enables the interrupt event request corresponding to 9'h040~9'h05F. INT_VSEL131: Each bit enables the interrupt event request corresponding to 9'h060~9'h07F. INT_VSEL132: Each bit enables the interrupt event request corresponding to 9'h080~9'h09F. INT_VSEL133: Each bit enables the interrupt event request corresponding to 9'h0A0~9'h0BF. INT_VSEL134: Each bit enables the interrupt event request corresponding to 9'h0C0~9'h0DF. INT_VSEL135: Each bit enables the interrupt event request corresponding to 9'h0E0~9'h0FF. INT_VSEL136: Each bit enables the interrupt event request corresponding to 9'h100~9'h11F. INT_VSEL137: Each bit enables the interrupt event request corresponding to 9'h120~9'h13F. INT_VSEL138: Each bit enables the interrupt event request corresponding to 9'h140~9'h15F. INT_VSEL139: Each bit enables the interrupt event request corresponding to 9'h160~9'h17F. INT_VSEL140: Each bit enables the interrupt event request corresponding to 9'h180~9'h19F. INT_VSEL141: Each bit enables the interrupt event request corresponding to 9'h1A0~9'h1BF. INT_VSEL142: Each bit enables the interrupt event request corresponding to 9'h1C0~9'h1DF. INT_VSEL143: Each bit enables the interrupt event request corresponding to 9'h1E0~9'h1FF. Please refer to the sequence number of the interrupt event request Table 10-2 Interrupt Event Request Sequence Number and Selection.	R/W

10.5.11 Soft-standby Wake Up Enable Register (INT_WUPEN)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	RXW UEN	-	TMR OWU EN	RTCP RDW UEN	RTCA LMW UEN	WKT MWU EN	CMPI OWU EN	PVD 2WU EN	PVD 1WU EN	SWD TWU EN
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0

EIRQWUEN[15:0]

Bit	Marking	Place name	Function	Read and write
b31~27	Reserved	-	Read as "0", write as "0"	R/W
b26	Reserved	-	Read as "0", write as "0"	R/W
b25	RXWUEN	USART1_WUPI Stop mode wake-up enable	0: Arousal prohibition 1: Arousal permit	R/W
b24	Reserved	-	Read as "0", write as "0"	R/W
b23	TMROWUEN	TMR01_GCMA stop mode wake-up enable	0: Arousal prohibition 1: Arousal permit	R/W
b22	RTCP RDWUEN	RTC_PRD stop mode wakeup enable	0: Arousal prohibition 1: Arousal permit	R/W
b21	RTCALMWUEN	RTC_ALM stop mode wake-up enable	0: Arousal prohibition 1: Arousal permit	R/W
b20	WKTM_PRD	WKTM_PRD cycle stop mode wakeup enable	0: Arousal prohibition 1: Arousal permit	R/W
B19	CMPI1WUEN	ACMP1 stop mode wake-up enable	0: Arousal prohibition 1: Arousal permit	R/W
B18	PVD2WUEN	PVD_PVD2 stop mode wake-up enable	0: Arousal prohibition 1: Arousal permit	R/W
B17	PVD1WUEN	PVD_PVD1 stop mode wake-up enable	0: Arousal prohibition 1: Arousal permit	R/W
B16	SWDTWUEN	SWDT_REFUDF stop mode wakeup enable	0: Arousal prohibition 1: Arousal permit	R/W
B15~b0	EIRQWUEN[15:0]	EIRQ stop mode wakeup enable	0: Arousal prohibition 1: Arousal permit	R/W

10.5.12 Software Interrupt & Event Register (INT_SWIER)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
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	Marking	Place name	Function	Read and write
b31~b0	SWIE	Software Interrupt/Event Register Bits	0: Invalid 1: A software interrupt event occurs Note: A software interrupt/event occurs after writing a '1'. Cleared after writing "0".	R/W

10.5.13 Event Enable Register (INT_EVTER)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
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	Marking	Place name	Function	Read and write
b31~b0	EVTE	Event Enable Register Bits	0: Event selection disabled 1: Event selection permission	R/W

10.5.14 Interrupt Enable Register (INT_IER)

Reset value: 0xFFFF_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
IER 31	IER 30	IER 29	IER 28	IER 27	IER 26	IER 25	IER 24	IER 23	IER 22	IER 21	IER 20	IER 19	IER 18	IER 17	IER 16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IER 15	IER 14	IER 13	IER 12	IER 11	IER 10	IER 9	IER 8	IER 7	IER 6	IER 5	IER 4	IER 3	IER 2	IER 1	IER 0

Bit	Marking	Place name	Function	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 request selected by the selection register INTSEL[8:0] corresponding to each interrupt vector will not be received by the NVIC. 0: Interrupt event requests selected by INTSEL[8:0] and software interrupt event requests are disabled 1: The interrupt event request selected by INTSEL[8:0] and the software interrupt event request are allowed	R/W

10.6 Precautions for use

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

11 Automatic Operation System (AOS)

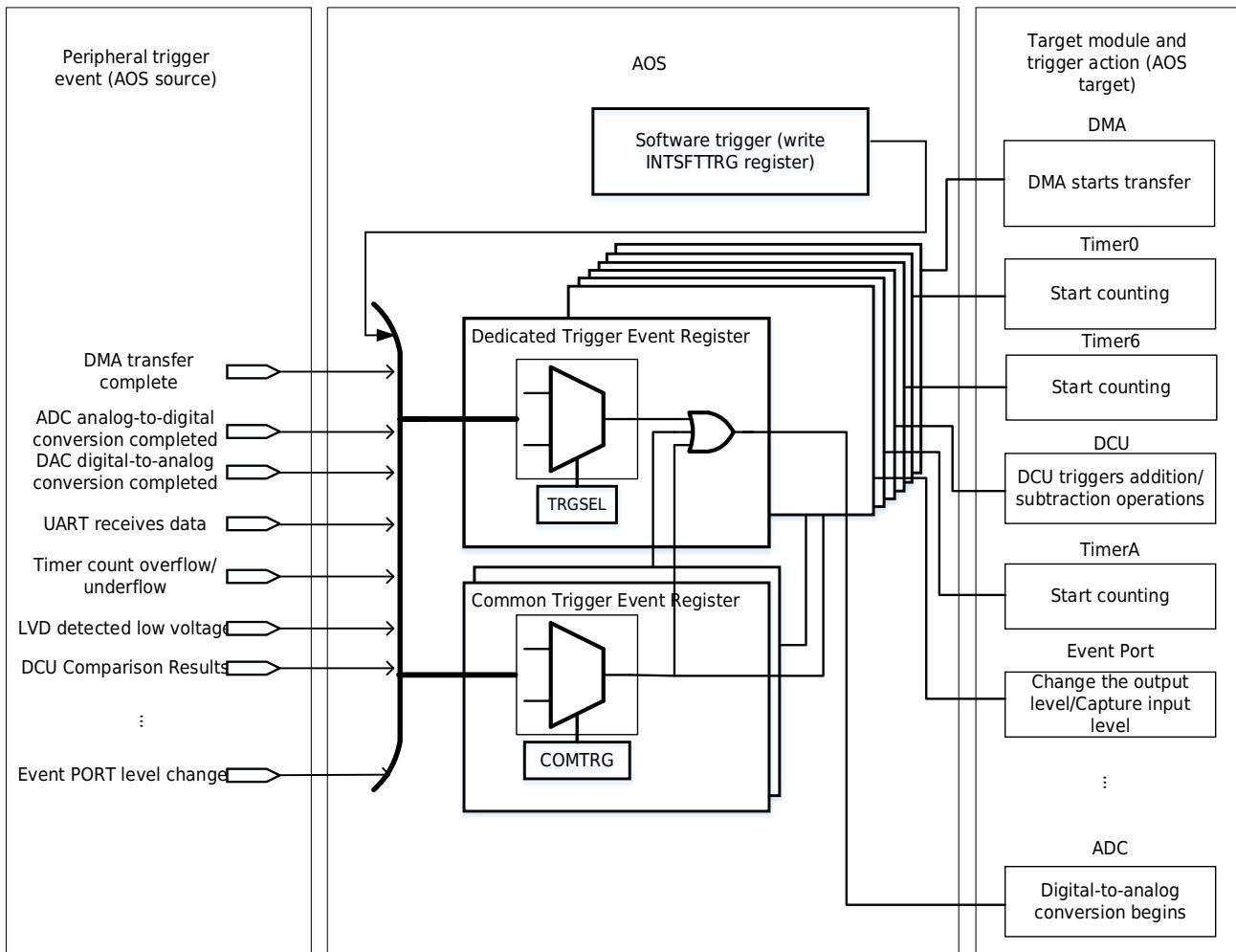
11.1 Introduction

The automatic operation system (Automatic Operation System) is used to realize the linkage between peripheral hardware circuits without the help of the CPU. Use the events generated by the peripheral circuit as the AOS source (AOS Source), such as the comparison and matching of the timer, timing overflow, the periodic signal of the RTC, and the various states of the communication module's sending and receiving data (idle, receiving data full, sending data end, Send data empty), ADC conversion end, etc., to trigger other peripheral circuit actions. The triggered peripheral circuit action is called AOS target (AOS Target).

11.1.1 Function Overview

- A Total of 195 AOS Sources, Except for Special RESTRICTIONS, EACH AOS TARGET Can Choose One of them as a Trigger Source, and Two Additional Sources Can Be selected Through the Public Trigger Source Selection Register 1 and Public Trigger Source Selection Register 2, 3, 4 triggers. The AOS target can be triggered when a trigger event occurs in any one of the sources. All AOS targets share these two common trigger sources.
- It can be triggered by the hardware of the peripheral circuit, or it can be triggered by the software by writing the register.
- The peripheral circuits that can be used as AOS targets operate as follows:
 - 4 DCU trigger targets, used to trigger DCU1~DCU4
 - Nine DMA trigger targets for two 4-channel DMAs to start data transfers and one DMA event to trigger channel resets
 - 2 advanced control timers (Timer6) trigger targets
 - 1 general-purpose timer 0 (Timer0) trigger target
 - 2 Event Port trigger targets, where Event Port Group1 and Event Port Group2 share an AOS target, and Event Port Group3 and Event Port Group4 share an AOS target
 - 2 general-purpose timer A (TimerA) trigger targets
 - 1 temperature sensor (OTS) trigger target
 - 2 groups of 2 AD trigger targets in each group, used for AD1~AD2 sequence trigger

11.1.2 Module diagram



11.2 Functional Description

11.2.1 List of AOS Source Events

For the number of AOS source events, see Table Table 10-2 in Section Interrupt Event Request Sequence Number in Chapter [Interrupt Controller (INTC)]. Events marked with "✓" in the "Can be selected as an event" column in the table can be used as AOS sources.

11.2.2 AOS Target List

Modules	action
DCU1	Trigger addition/subtraction operation
DCU2	Trigger addition/subtraction operation
DCU3	Trigger addition/subtraction operation
DCU4	Trigger addition/subtraction operation
DMA1	Channel 0 starts transmission Channel 1 starts transmission Channel 2 starts transmission Channel 3 starts transmission
DMA2	Channel 0 starts transmission Channel 1 starts transmission Channel 2 starts transmission Channel 3 starts transmission
DMA1&2	Event triggered channel reset
Timer6	Start counting
Timer0	Start counting
Event Port	Event Port1&2 trigger action Event Port3&4 trigger action
TimerA	Start counting/capturing
OTS	Start measuring temperature
ADC1	Start analog to digital conversion
ADC2	Start analog to digital conversion

11.3 Action Description

11.3.1 Dedicated Trigger Source

The peripheral circuit module with AOS target is equipped with a dedicated peripheral trigger source selection register for each AOS target. When this register is written into the event number corresponding to the AOS source, the AOS target will select this AOS source as the trigger source. When an event from the AOS source occurs, the event will be delivered 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 Public 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). Used to implement the function of multiple AOS sources triggering the same AOS target. When using, first enable the public trigger source enable position in the AOS target-specific peripheral trigger source selection register, and then write the event number corresponding to the AOS source in the public trigger source selection register. When an event of the AOS source occurs, the event will be passed 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 setting. When the dedicated trigger source and public 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 any trigger event occurs in any 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 public trigger source selection register, it is necessary to disable the public trigger source enable position in its dedicated peripheral trigger source selection register to prevent wrong trigger actions.

11.4 Register Description

Register overview

Register base address: 0x4001_0800

Abbreviation	Name	Offset address
INTSFTTRG	Peripheral Trigger Event Register	0x00
DCU_TRGSELx(x=1~4)	DCU trigger source select register	0x04,0x08,0x0C,0x10
DMA1_TRGSELx(x=0~3)	DMA1 Transmit start trigger source select register	0x14,0x18,0x1C,0x20
DMA2_TRGSELx(x=0~3)	DMA2 Transmit start trigger source select register	0x24,0x28,0x2C,0x30
DMA_TRGSELRC	DMA channel reset trigger source select register	0x34
TMR6_HTSSRx(x=0~1)	Timer6 hardware trigger event selection register	0x38,0x3C
TMR0_HTSSR	Timer0 trigger selection register	0x40
PEVNTTRGSR12	Event Port1,2 trigger source selection register	0x44
PEVNTTRGSR34	Event Port3,4 trigger source selection register	0x48
TMRA_HTSSR0	TimerA internal trigger event selection register 0	0x4C,
TMRA_HTSSR1	TimerA internal trigger event selection register 1	0x50
OTS_TRG	OTS trigger source selection register	0x54
ADC1_ITRGSELRx(x=0,1)	A/D1 starts on-chip trigger source selection register	0x58, 0x5C
ADC2_ITRGSELRx(x=0,1)	A/D2 start on-chip trigger source selection register	0x60, 0x64
AOS_COMTRG1	Common trigger source selection register 1	0x68
AOS_COMTRG2	Common trigger source selection register 2	0x6C

11.4.1 Peripheral Trigger Event Register (INTSFTTRG)

Register Description: Writing this register will generate a trigger event.

Offset address: 0x00

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Bit	Marking	Place name	Function										Read and write		
b31~b1	Reserved	-	Read at "0", write at "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, writing 0 by software is invalid											W	

11.4.2 DCU Trigger Source Selection Register (DCU_TRGSELx) (x=1~4)

Register description: After the DCU selects the hardware trigger start mode, 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 the event and perform operations. When DCU_TRGSEL1 writes the event number, DCU1 will be triggered when the numbered event occurs; when DCU_TRGSEL2 writes the event number, when the numbered event occurs, DCU2 will be triggered; when DCU_TRGSEL3 writes the event number, when the numbered event occurs, DCU3 will be triggered ; When DCU_TRGSEL4 writes the event number, when the numbered event occurs, DCU4 will be triggered.

Offset address: 0x04, 0x08, 0x0C, 0x10

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG2 from triggering the DCU 1: Allow public trigger events of AOS_COMTRG2 to trigger DCU										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG1 from triggering the DCU 1: Allow public trigger events of AOS_COMTRG1 to trigger DCU										R/W		
b29~b9	Reserved	-	Read at "0", write at "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected Please refer to the chapter Interrupt Controller (INTC) for the specific number.										R/W		

11.4.3 DMA1 Transfer Start Trigger Source Selection Register (DMA1_TRGSELx) (x=0~3)

Register description: After DMA1 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, DMA1 will be triggered by the event to start and transmit.

Offset address: 0x14, 0x18, 0x1C, 0x20

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							
<hr/>															
Bit	Marking	Place name	Function												Read and write
b31	COMEN[1]	Common trigger enable	0: Disable AOS_COMTRG2 public trigger event to trigger DMA1 transfer 1: Allow public trigger event of AOS_COMTRG2 to trigger DMA1 transfer												R/W
b30	COMEN[0]	Common trigger enable	0: Disable AOS_COMTRG1 public trigger event to trigger DMA1 transfer 1: Allow public trigger event of AOS_COMTRG1 to trigger DMA1 transfer												R/W
b29~b9	Reserved	-	Read at "0", write at "0"												R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Select the event number to start the corresponding channel for transmission. Please refer to the chapter Interrupt Controller (INTC) for the specific number.												R/W

11.4.4 DMA2 Transfer Start Trigger Source Selection Register (DMA2_TRGSELx) (x=0~3)

Register description: After DMA2 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, DMA2 will be triggered by the event and start transmission.

Offset address: 0x24, 0x28, 0x2C, 0x30

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
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	COMEN[1]	Common trigger enable	0: Disable AOS_COMTRG2 public trigger event to trigger DMA2 transfer 1: Allow public trigger events of AOS_COMTRG2 to trigger DMA2 transfers										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable AOS_COMTRG1 public trigger event to trigger DMA2 transfer 1: Allow public trigger events of AOS_COMTRG1 to trigger DMA2 transfers										R/W		
b29~b9	Reserved	-	Read at "0", write at "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Select the event number to start the corresponding channel for transmission. Please refer to the chapter Interrupt Controller (INTC) for the specific number.										R/W		

11.4.5 DMA Channel Reset Trigger Source Selection Register (DMA_TRGSELRC)

Register description: After the DMA 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, the DMA will be triggered by the event to reset the channel. DMA1 and DMA2 share this register

Offset address: 0x34

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable AOS_COMTRG2 public trigger event to trigger DMA channel reset 1: Allow public trigger event of AOS_COMTRG2 to trigger DMA channel reset										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable AOS_COMTRG1 public trigger event to trigger DMA channel reset 1: Allow public trigger event of AOS_COMTRG1 to trigger DMA channel reset										R/W		
b29~b9	Reserved	-	Read at "0", write at "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Select the event number that triggers the channel to reset. Please refer to the chapter Interrupt Controller (INTC) for the specific number. DMA_1, DMA_2 share a reset trigger source.										R/W		

11.4.6 Timer6 Hardware Trigger Event Selection Register (TMR6_HTSSRx) ($x=0\sim1$)

Register description: After Timer6 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, Timer6 will be triggered by the event.

Offset address: 0x38, 0x3C

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

Bit	Marking	Place name	Function	Read and write
b31	COMEN[1]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG2 from triggering TMR6 1: Allow public trigger events of AOS_COMTRG2 to trigger TMR6	R/W
b30	COMEN[0]	Common trigger enable	0: disable AOS_COMTRG1 public trigger event to trigger TMR6 1: Allow public trigger events of AOS_COMTRG1 to trigger TMR6	R/W
b29~b9	Reserved	-	Read at "0", write at "0"	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write trigger source number Please refer to the chapter Interrupt Controller (INTC) for the specific number.	R/W

Note:

- The trigger selection register (TMR6_HTSSR0~1) is two registers independent of the unit, which are shared by the three units Timer6.

11.4.7 Timer0 Hardware Trigger Event Selection Register (TMR0-HTSSR)

Register description: After Timer0 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, Timer0 will be triggered by the event.

Offset address: 0x40

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	COMEN[1]	Common trigger enable	0: disable AOS_COMTRG2 common trigger event to trigger TMR0 1: Allow the public trigger event of AOS_COMTRG2 to trigger TMR0										R/W		
b30	COMEN[0]	Common trigger enable	0: disable AOS_COMTRG1 public trigger event to trigger TMR0 1: Allow the public trigger event of AOS_COMTRG1 to trigger TMR0										R/W		
b29~b9	Reserved	-	Read at "0", write at "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write trigger source number Please refer to the chapter Interrupt Controller (INTC) for the specific number.										R/W		

Note:

- The trigger select register (TMR0-HTSSR) is an independent register shared by Timer0 of 2 units.

11.4.8 Event Port Trigger Source Select Register (PEVNTTRGSR12, PEVNTTRGSR34)

Register Description: Set the corresponding event number to trigger the Event Port to output a specified level, or to latch the input status of the I/O port. PEVNTTRGSR12 sets the trigger source of Event Port1 and 2, and PEVNTTRGSR34 sets the trigger source of Event Port3 and 4.

Offset address: 0x44, 0x48

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

Bit	Marking	Place name	Function	Read and write
b31	COMEN[1]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG2 from triggering the Event Port 1: Allow the public trigger event of AOS_COMTRG2 to trigger Event Port	R/W
b30	COMEN[0]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG1 from triggering the Event Port 1: Allow the public trigger event of AOS_COMTRG1 to trigger Event Port	R/W
b29~b9	Reserved	-	Read at "0", write at "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 to latch the input status of the I/O port. PEVNTTRGSR12 sets the trigger source of Event Port1 and 2, and PEVNTTRGSR34 sets the trigger source of Event Port3 and 4. Please refer to the chapter Interrupt Controller (INTC) for the specific number.	R/W

11.4.9 TimerA Internal Trigger Event Selection Register 0 (TMRA_HTSSR0)

Register description: After TimerA 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, TimerA will be triggered by the event.

Offset address: 0x4C

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								CNTTRGSEL[8:0]							
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable AOS_COMTRG2 public trigger events from triggering TMRA counters 1: Allow public trigger events of AOS_COMTRG2 to trigger TMRA counters										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable AOS_COMTRG1 public trigger event from triggering TMRA counter 1: Allow public trigger events of AOS_COMTRG1 to trigger TMRA counters										R/W		
b29~b9	Reserved	-	Read at "0", write at "0"										R/W		
b8~b0	CNTTRGSEL[8:0]	Counter trigger event trigger source selection	Counter trigger event trigger source number write Please refer to the chapter Interrupt Controller (INTC) for the specific number.										R/W		

Note:

- The internal trigger event selection register (TMRA_HTSSR0~1) is and Two separate register shared by TimerA in six units.

11.4.10 TimerA Internal Trigger Event Selection Register 1 (TMRA_HTSSR1)

Register description: After TimerA 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, TimerA will be triggered by the event.

Offset address: 0x50

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								ICPTRGSEL[8:0]							
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger TMRA capture action 1: Allow public trigger events of AOS_COMTRG2 to trigger TMRA capture actions										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger TMRA capture action 1: Allow the public trigger event of AOS_COMTRG1 to trigger TMRA capture action										R/W		
b29~b9	Reserved	-	Read at "0", write at "0"										R/W		
b8~b0	ICPTRGSEL[8:0]	Capture action trigger event trigger source selection	Capture action trigger event trigger source number write Please refer to the chapter Interrupt Controller (INTC) for the specific number.										R/W		

11.4.11 OTS Trigger Source Selection Register (OTS_TRG)

Register description: After the OTS 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, the OTS will be triggered by the event.

Offset address: 0x54

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG2 from triggering OTS 1: Allow public trigger events of AOS_COMTRG2 to trigger OTS										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG1 from triggering OTS 1: Allow public trigger events of AOS_COMTRG1 to trigger OTS										R/W		
b29~b9	Reserved	-	Read at "0", write at "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Select the trigger source number when the hardware trigger starts, Please refer to the chapter Interrupt Controller (INTC) for the specific number.										R/W		

11.4.12 A/D1 Conversion Starts On-chip 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 by the event.

Offset address: 0x58, 0x5C

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	COMEN[1]	Common trigger enable	0: disable AOS_COMTRG2 common trigger event to trigger ADC1 1: Allow the common trigger event of AOS_COMTRG2 to trigger ADC1										R/W		
b30	COMEN[0]	Common trigger enable	0: disable AOS_COMTRG1 public trigger event to trigger ADC1 1: Allow the common trigger event of AOS_COMTRG1 to trigger ADC1										R/W		
b29~b9	Reserved	-	Read at "0", write at "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected Please refer to the chapter Interrupt Controller (INTC) for the specific number.										R/W		

11.4.13 A/D2 Conversion Starts On-chip 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 by the event.

Offset address: 0x60, 0x64

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]		Reserved													
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	COMEN[1]	Common trigger enable	0: disable AOS_COMTRG2 common trigger event to trigger ADC2 1: Allow the public trigger event of AOS_COMTRG2 to trigger ADC2												R/W
b30	COMEN[0]	Common trigger enable	0: disable AOS_COMTRG1 public trigger event to trigger ADC2 1: Allow the public trigger event of AOS_COMTRG1 to trigger ADC2												R/W
b29~b9	Reserved	-	Read at "0", write at "0"												R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected Please refer to the chapter Interrupt Controller (INTC) for the specific number.												R/W

11.4.14 Common Trigger Source Selection Register 1 (AOS_COMTRG1)

Register description: Write the number of the event to be triggered in AOS_COMTRG1. When the peripheral circuit event corresponding to the number occurs, if the COMEN[1] bit value of the dedicated trigger source selection register of one or more AOS targets is 1, then the peripheral circuit event corresponding to the number will trigger the start of one or more AOS targets.

Offset address: 0x68

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								COMTRG[8:0]							
<hr/>															
Bit	Marking	Place name	Function												Read and write
b31~b9	Reserved	-	Read at "0", write at "0"												R/W
b8~b0	COMTRG[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected Please refer to the chapter Interrupt Controller (INTC) for the specific number.												R/W

11.4.15 Common Trigger Source Selection Register 2 (AOS_COMTRG2)

Register description: Write the number of the event to be triggered in AOS_COMTRG2. When the peripheral circuit event corresponding to the number occurs, if the COMEN[0] bit value of the dedicated trigger source selection register of one or more AOS targets is 1, then the peripheral circuit event corresponding to the number will trigger the start of one or more AOS targets.

Offset address: 0x6C

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										COMTRG[8:0]					
Bit	Marking	Place name	Function										Read and write		
b31~b9	Reserved	-	Read at "0", write at "0"										R/W		
b8~b0	COMTRG[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected Please refer to the chapter Interrupt Controller (INTC) for the specific number.										R/W		

12 Keyboard Scan Control Module (KEYSCAN)

12.1 Introduction

This product is equipped with a keyboard control module (KEYSCAN) 1 unit. The KEYSCAN module supports keyboard array (row and column) scanning, the column is driven by an independent scan output KEYOUT_m ($m=0\sim 7$), and the row KEYIN_n ($n=0\sim 15$) is input as EIRQ_n ($n=0\sim 15$) is detected. This module realizes the key recognition function through the line scan query method.

KEYSCAN main features:

- EIRQ0~EIRQ15 can be independently selected as the row input of the keyboard array.
- KEYOUT can be selected by register.
- Output low level sequentially at regular intervals to scan the keyboard array.
- Scan time can be set.
- Stop scanning when the IRQ interrupt is detected, and locate the pressed key according to the SSR.INDEX value and the IRQ interrupt flag (INT_EIFR.EIFR).

12.2 KEYSCAN System Block Diagram

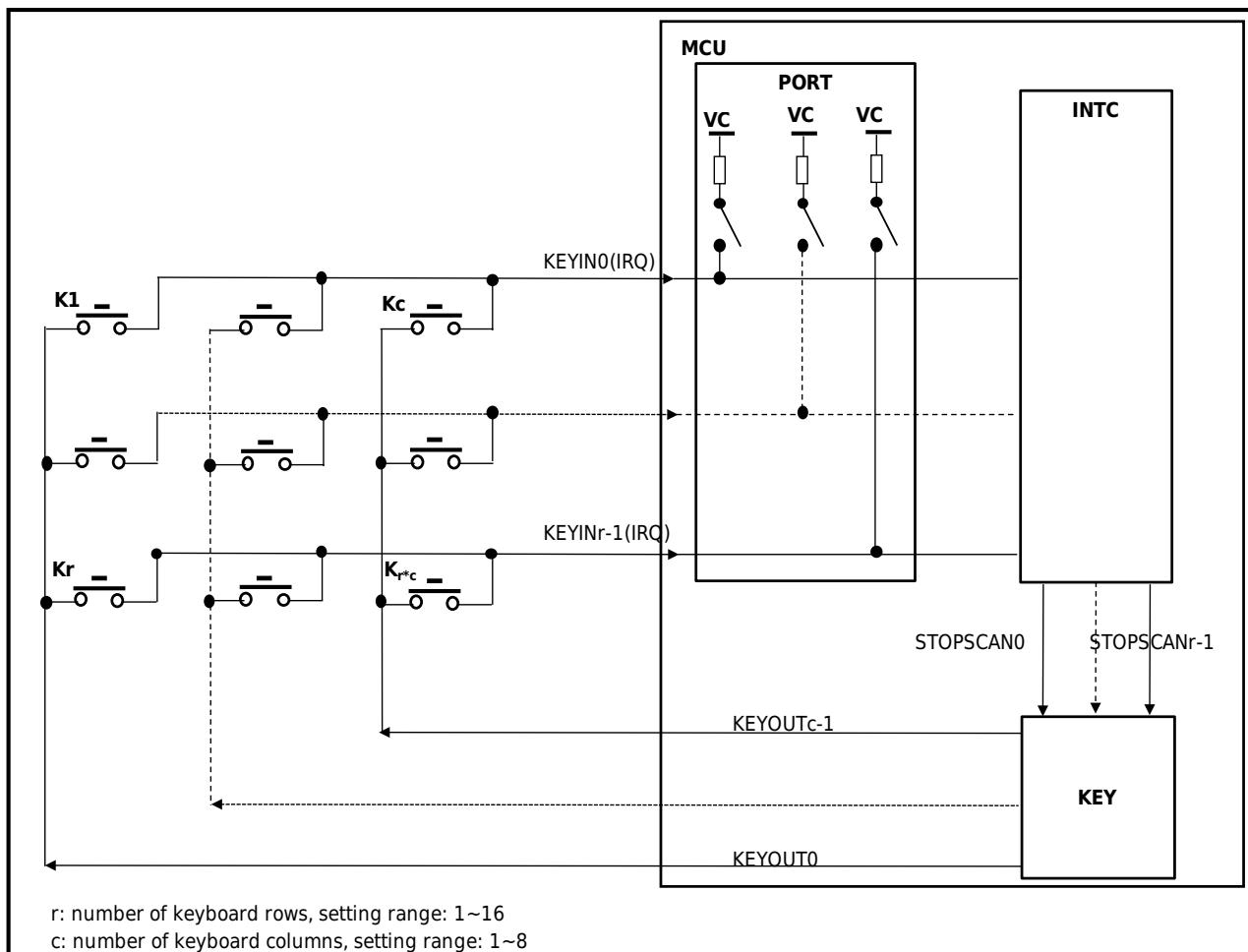


Figure 12-1 KEYSCAN System Block Diagram

12.3 Pin Description

Table 12-1 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

12.4 Functional Description

This chapter will describe the keyboard scan function and key recognition function in detail.

12.4.1 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 an EIRQ interrupt flag, which is located by comparing the value of the interrupt flag bit (INT_EIFR.EIFR) and SSR.INDEX[2:0]. The key is currently pressed.

Through the register SCR.KEYINSEL[15:0], KEYIN can be independently selected from EIRQ0~EIRQ15, and through the register SCR.KEYOUTSEL[2:0], the KEYOUT pin can be selected, so that the number of rows and columns of the keyboard can be flexibly selected. It can support a keyboard array of 16 rows*8 columns.

12.4.2 Keyboard Scan Function

The keyboard scanning function is: output low level to the columns of the keyboard array in a continuous cycle, 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, 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 scanning automatically restarts after the corresponding interrupt flag is cleared.

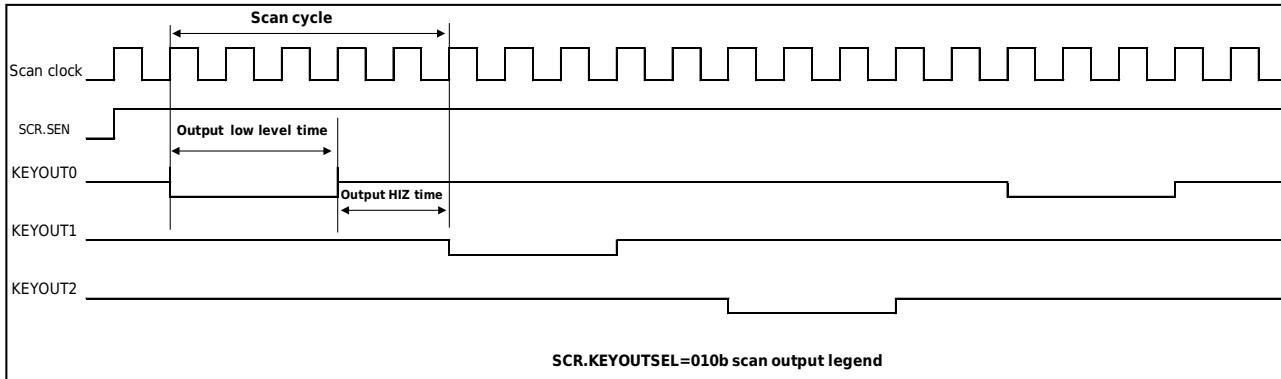


Figure 12-2 Schematic Diagram of Keyboard Scanning Function

12.4.3 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 open the digital filter function, and set the appropriate filter 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 you use the on-chip pull-up resistor, please refer to the PORT characteristics to select the appropriate scan time and filter time.

12.5 Register Description

KEYSCAN_BASE_ADDR: 0x4005_0C00

Table 12-2 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 status register	KEYSCAN_SSR	0x08	32	0x0000_0000

12.5.1 KEYS defense Scan Control Register (KEYSCAN_SCR)

Offset address: 0x00

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
T_HIZ[2:0]		T_LLEVEL[4:0]				-	-	CKSEL[1:0]	-	KEYOUTSEL[2:0]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
KEYINSEL[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b29	T_HIZ[2:0]	Output HIZ time	KEYOUT output low level HIZ time (number of scan clocks) Scan cycle = output low level time + output HIZ time Setting value: number of 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 level time	KEYOUT output low level time (number of scan clocks) Scan cycle = output low level time + output HIZ time Output low level time = 2 T_LLEVEL power 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, and the maximum settable value is 11000b	R/W											
b23~b22	Reserved	-	Read as "0", write as "0"	R											
b21~b20	CKSEL[1:0]	Scan clock source select bit	Scan clock source select bit 00b: System clock HCLK 01b: Internal low-speed oscillator LRC 10b: External low-speed oscillator XTAL32 11b: Prohibitions 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											
b18~b16	KEYOUTSEL[2:0]	Output selection	KEYOUT output selection bit Set value: Output selection 000b: Prohibition 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 selection bit, the selected row is used as the row of the keyboard array, and as EIRQn (n: 0~15) is detected KEYINSEL[n]=0: KEYINSEL[n] is not used as the line 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											

12.5.2 KEYS defense Scan Enable Register (KEYSCAN_SER)

Offset address: 0x04

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b30	Reserved	-	Read as "0", write as "0"	R
b0	SEN	scan enable bit 0: scan disabled 1: scan enable		R/W

12.5.3 KEYS defense Scan Status Register (KEYSCAN_SSR)

Offset address: 0x08

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R
b2~b0	INDEX[2:0]	Current working SCAN pindex bit	Current working SCAN pindex bit 000: The current working SCAN pin is KEYOUT0 001: The currently working SCAN pin is KEYOUT1 010: The currently working SCAN pin is KEYOUT2 011: The currently working SCAN pin is KEYOUT3 100: The currently working SCAN pin is KEYOUT4 101: The currently working SCAN pin is KEYOUT5 110: The currently working SCAN pin is KEYOUT6 111: The currently working SCAN pin is KEYOUT7 Note: The SSR.INDEX[2:0] bits are read-only registers, and the data read only when SER.SEN=1 is meaningful	R

13 Memory Protection Unit (MPU)

13.1 Introduction

The MPU can provide protection to the memory, and can improve the security of the system by preventing unauthorized access.

This chip has built-in four MPU units for host and one MPU unit for IP.

Modules	Content
ARM MPU	Memory Protection Unit of the CPU 8 areas, see ARM MPU description for details
System DMA_1 MPU: SMPU1	Memory protection unit of system DMA_1 16 areas, 8 areas dedicated to system DMA, 8 areas shared by all DMAs
System DMA_2 MPU: SMPU2	Memory protection unit of system DMA_2 16 areas, 8 areas dedicated to system DMA, 8 areas shared by all DMAs
USBFS-DMA MPU: FMPU	Storage Protection Unit of USBFS-DMA 8 regions, shared by 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 respectively provide system DMA_1/system DMA_2/USBFS-DMA to control the read and write access rights of the entire 4G address space. When accessing the prohibited space, the MPU action can be set to ignore/bus error/non-maskable interrupt/reset.

The IPMPU provides access control to system IP and security-related IP in non-privileged mode.

13.2 Functional Description

13.2.1 Locale Setting

The MPU manages the authority of the storage space in units of regions. Each area can independently set the base address and area size, the range of which can be set is 32Byte~4GByte, the size must be 2^n Byte ($n=5\sim 32$), and the corresponding lower n bits of the base address are 0.

The address space not covered by any region is called the background region.

13.2.2 Permission Settings

Each area, including the background area, can be independently set for each DMA to allow reading/prohibit reading and allow writing/prohibit writing. If address overlap occurs between different areas, the set prohibition takes precedence.

13.2.3 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

13.2.4 Start MPU

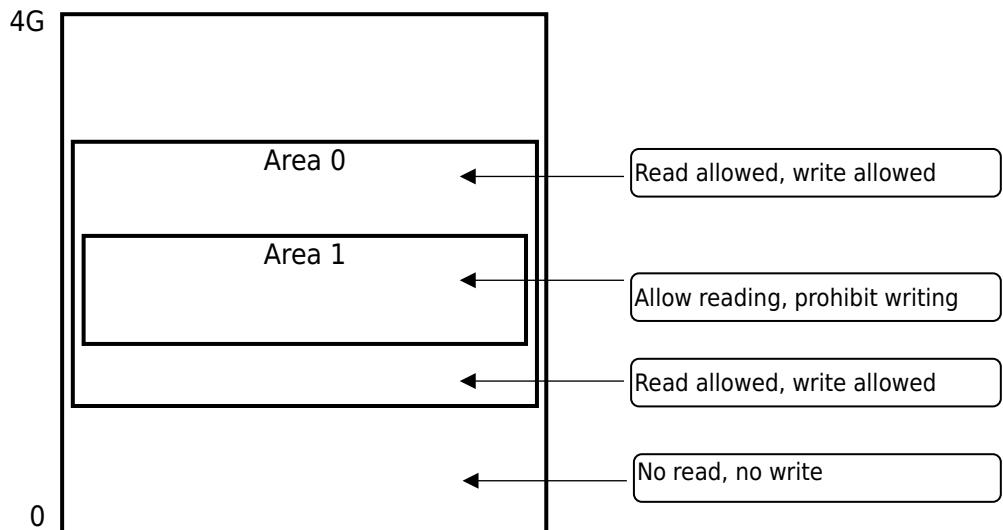
SMPU1/SMPU2/FMPU can be enabled independently.

It is recommended to enable the MPU after setting the area range/permission setting/action selection.

13.3 Application Examples

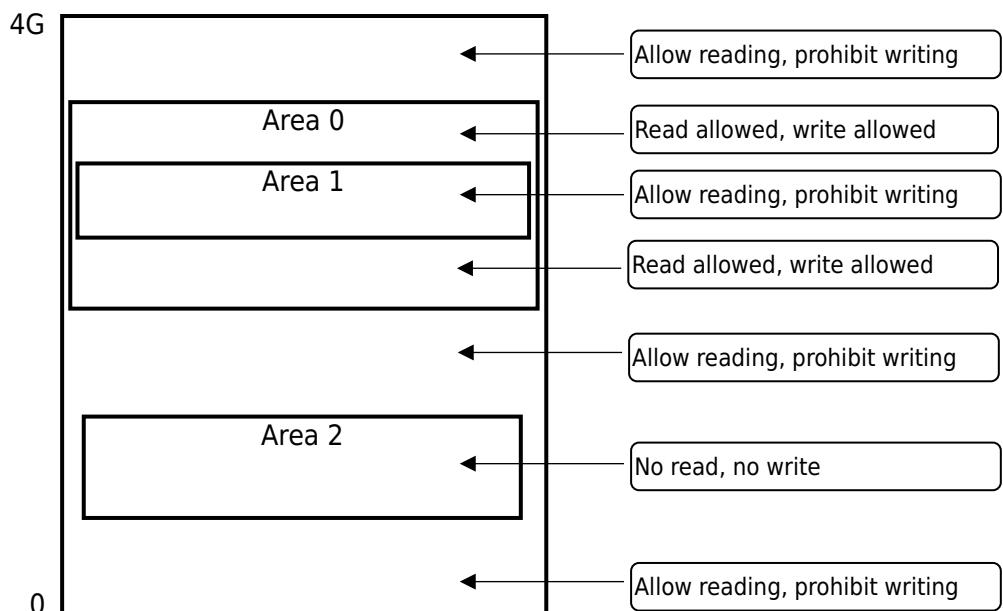
13.3.1 Only Allow Partial Space Access

Example: Set the permission of the background area to prohibit reading/writing, set area 0 to allow reading/allow writing, set area 1 to allow reading and prohibit writing, and the range of area 0 covers area 1.



13.3.2 Only Some Space Access is Prohibited

Example: Set the permission of the background area to allow reading/prohibiting writing, set area 0 to allow reading/allow writing, set area 1 to allow reading/prohibit writing, area 0 covers area 1, and set area 2 to prohibit reading/ prohibit writing.



13.4 Register Description

The registers of this module can only be set by the CPU.

MPU base address: 0x4005_0000

Offset address	Register name	Initial value	Name	Write protection
+00~+3C	MPU_RGD0~15	0x0000_0000	Area 0~15 range description register	MPUWE
+40~+7C	MPU_RGCR0~15	0x0000_0000	Area 0~15 Control Register	MPUWE
+80	MPU_CR	0x0000_0000	MPU control register	MPUWE
+84	MPU_SR	0x0000_0000	MPU status register	No
+88	MPU_ECLR	0x0000_0000	MPU Error Flag Clear Register	No
+8C	MPU_WP	0x0000_0000	MPU write protection register	WKEY

Base address: 0x40054000

Offset address	Register name	Initial value	Name	Write protection
+1C	MPU_IPPR	0x0000_0000	IP access protection register	SYS

13.4.1 Area scope Description Register MPU_RGDr (n=0~15)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
MPURGnADDR [31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MPURGnADDR[15:5]										MPURGnSIZE[4:0]					
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b5	MPURGnADDR[31:5]	Zone base address	Set the base address of area n, the effective number of digits is related to the area size, and the low (MPURGnSIZE+1) bit is fixed to 0	R/W											
b4~b0	MPURGnSIZE[4:0]	Area size	Set the size of the region n 00000~00011: Reserved, setting prohibited 00100: 32Byte 00101: 64Byte 11110: 2GByte 11111: 4GByte	R/W											

13.4.2 Regional Control Register MPU_RGCRn (n=0~15)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	FRGnE	-	-	-	-	-	FRGnWP	FRGnRP
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
S1RGnE	-	-	-	-	-	S1RGnWP	S1RGnRP	S2RGnE	-	-	-	-	-	S2RGnWP	S2RGnRP

Bit	Marking	Place name	Function	Read and write
b31~b24	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b23	FRGnE	FMPU region n enable	0: Region n of FMPU is invalid 1: FMPU area n is valid	R/W
b22~b18	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b17	FRGnWP	FMPU area n write permission	0: Area n allows USBFS-DMA write 1: Area n prohibits USBFS-DMA writing	R/W
b16	FRGnRP	FMPU area n read permissions	0: Area n allows USBFS-DMA read 1: Area n prohibits USBFS-DMA reading	R/W
b15	S1RGnE	SMPU1 zone n enable	0: Area n of SMPU1 is invalid 1: Area n of SMPU1 is valid	R/W
b14~b10	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b9	S1RGnWP	SMPU1 area n write permission	0: area n allows system DMA_1 to write 1: Area n prohibits system DMA_1 from writing	R/W
b8	S1RGnRP	SMPU1 area n read permission	0: Area n allows system DMA_1 to read 1: Area n prohibits system DMA_1 from reading	R/W
b7	S2RGnE	SMPU2 zone n enable	0: Area n of SMPU2 is invalid 1: Area n of SMPU2 is valid	R/W
b6~b2	reserved	-	Reserved bit, read as 0, write 0 when writing	R/W
b1	S2RGnWP	SMPU2 area n write permission	0: area n allows system DMA_2 to write 1: Area n prohibits system DMA_2 from writing	R/W
b0	S2RGnRP	SMPU2 area n read permission	0: Area n allows system DMA_2 to read 1: Area n prohibits system DMA_2 from reading	R/W

Note:

- The control registers b31~b16 of areas 8~15 are reserved bits.

13.4.3 Control Register MPU_CR

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	FMPUE	-	-	-	FMPUACT[1:0]	FMPUB WP	FMPUB RP	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SMPU1 E	-	-	-	SMPU1ACT[1:0]	SMPU1 BWP	SMPU1 BRP	SMPU2E	-	-	-	SMPU2ACT[1:0]	SMPU2 BWP	SMPU2 BRP		

Bit	Marking	Place name	Function	Read and write
b31~b24	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b23	FMPUE	FMPU enable	0: FMPU invalid 1: FMPU is valid	R/W
b22~b20	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b19~b18	FMPUACT[1:0]	FMPU action selection	Set the action when USBFS-DMA access is prohibited 00: ignore (read access reads 0, write access ignore) 01: ignore + bus error 10: ignore + unmaskable interrupt 11: reset	R/W
b17	FMPUBWP	FMPU background write permission setting	0: FMPU background space allows USBFS-DMA write 1: FMPU background space prohibits USBFS-DMA writing	R/W
b16	FMPUBRP	FMPU background read permission setting	0: FMPU background space allows USBFS-DMA read 1: FMPU background space prohibits USBFS-DMA reading	R/W
b15	SMPU1E	SMPU1 enable	0: SMPU1 invalid 1: SMPU1 effective	R/W
b14~b12	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b11~b10	SMPU1ACT[1:0]	SMPU1 action selection	Set the action when the system DMAC_1 is prohibited from accessing 00: ignore (read access reads 0, write access ignore) 01: ignore + bus error 10: ignore + unmaskable interrupt 11: reset	R/W
b9	SMPU1BWP	SMPU1 background write permission setting	0: SMPU1 background space allows system DMAC_1 to write 1: SMPU1 background space prohibits system DMAC_1 from writing	R/W
b8	SMPU1BRP	SMPU1 background read permission setting	0: SMPU1 background space allows system DMAC_1 to read 1: SMPU1 background space prohibits system DMAC_1 from reading	R/W
b7	SMPU2E	SMPU2 enable	0: SMPU2 invalid 1: SMPU2 effective	R/W
b6~b4	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b3~b2	SMPUACT[1:0]	SMPU action selection	Set the action when the system DMA_2 access is prohibited 00: ignore (read access reads 0, write access ignore) 01: ignore + bus error 10: ignore + unmaskable interrupt 11: reset	R/W
b1	SMPU2BWP	SMPU2 background write permission setting	0: SMPU2 background space allows system DMA_2 to write 1: SMPU2 background space prohibits system DMA_2 from 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 from reading	R/W

When multiple area settings overlap, the priority is: setting prohibited > setting allowed.

13.4.4 Status Flag Register MPU_SR

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FMPUE AF
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	SMPU1 EAF	-	-	-	-	-	-	-	SMPU2 EAF

Bit	Marking	Place name	Function	Read and write
b31~b17	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b16	FMPUEAF	FMPU error flag	0: USBFS-DMA has no error access 1: USBFS-DMA has a wrong access	R
b15~b9	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b8	SMPU1EAF	SMPU1 error flag	0: System DMA_1 has no error access 1: System DMA_1 has an error access	R
b7~b1	reserved	-	Reserved bit, read as 0, write 0 when writing	R
b0	SMPU2EAF	SMPU2 Error Flags	0: System DMA_2 has no error access 1: System DMA_2 has an error access	R

The write operation to this register will be ignored, please use MPUECLR to clear the error flag.

13.4.5 Flag Clear Register MPU_ECLR

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FMPUE CLR
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	SMPU1 ECLR	-	-	-	-	-	-	-	SMPU2 ECLR

Bit	Marking	Place name	Function	Read and write
b31~b17	Reserved	-	Reserved bit, read as 0, write 0 when writing	R
b16	FMPUECLR	FMPU error flag cleared	Writing 1 can clear FMPUEAF to 0	W
b15~b9	Reserved	-	Reserved bit, read as 0, write 0 when writing	R
b8	SMPU1ECLR	SMPU1 error flag cleared	Writing 1 can clear SMPU1EAF to 0	W
b7~b1	Reserved	-	Reserved bit, read as 0, write 0 when writing	R
b0	SMPU2ECLR	SMPU2 error flag cleared	Writing 1 can clear SMPU2EAF to 0	W

The read value of this register is fixed at 0x0000_0000.

13.4.6 Write protection register MPU_WP

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WKEY[15:1]															MPUWE

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Reserved bit, read as 0, write 0 when writing	R
b15~b1	WKEY[15:1]	Write code	When writing to MPUWE, it must be written to WKEY at the same time 'b1001_0110_1010_010, read as 0	W
b0	MPUWE	MPU register write enable	0: MPU address registers/control registers are not allowed to be written 1: MPU address register/control register allows writing	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.

13.4.7 IP Access Protection Register MPU_IPPR

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
BUSER RE	-	MSTPW RP	MSTPR DP	SYSCW RP	SYSCR DP	INTCW RP	INTCRD P	SRAMC WRP	SRAMC RDP	DMPU WRP	DMPUR DP	RTCWR P	RTCRD P	BKSRA MWRP	BKSRA MRDP
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SWDT WRP	SWDTR DP	WDTW RP	WDTRD P	-	-	EFCMCW RP	EFCMCR DP	CRCWR P	CRCRD P	TRNGW RP	TRNGR DP	HASHW RP	HASHR DP	AESWR P	AESRD P

Bit	Marking	Place name	Function	Read and write
b31	BUSERRE	Bus error enable	0: Ignore access to protected IP 1: When an access to the protected IP occurs, a bus error is returned	RW
b30	-	-	Reserved bit, read as 0, write 0 when writing	R
b29	MSTPWWRP	MSTP write protection	0: Allow write operation to registers PWC_FCG0/1/2/3, PWC_FCG0PC 1: Disable write operation to registers PWC_FCG0/1/2/3, PWC_FCG0PC	RW
b28	MSTPRDP	MSTP read protection	0: Allow read operation of registers PWC_FCG0/1/2/3, PWC_FCG0PC 1: Disable the read operation of registers PWC_FCG0/1/2/3, PWC_FCG0PC	RW
b27	SYSCWRP	SYSC write protection	0: Allow write operation to RMU/CMU/PWC 1: Disable writing to RMU/CMU/PWC	RW
b26	SYSCRDP	SYSC read protection	0: Allow read operation of RMU/CMU/PWC 1: Disable the read operation of RMU/CMU/PWC	RW
b25	INTCWRP	INTC write protection	0: Allow write operation to INTC 1: Prohibit writing to INTC	RW
b24	INTCRDP	INTC read protection	0: Allow read operation on INTC 1: Disable the read operation of INTC	RW
b23	SRAMCWRP	SRAMC write protection	0: Allow write operation to [10. Built-in SRAM] register 1: Prohibit writing to the register of [10. Built-in SRAM]	RW
b22	SRAMCRDP	SRAMC read protection	0: Allow the register read operation of [10. Built-in SRAM] 1: Disable the register read operation of [10. Built-in SRAM]	RW
b21	DMPUWRP	DMPU write protection	0: Allow write operation to SMPU1/SMPU2/FMPU/IPMPU 1: Disable writing to SMPU1/SMPU2/FMPU/IPMPU	RW
b20	DMPURDP	DMPU read protection	0: Allow read operation on SMPU1/SMPU2/FMPU/IPMPU 1: Disable read operation on SMPU1/SMPU2/FMPU/IPMPU	RW
b19	RTCWRP	RTC write protection	0: Allow writing to RTC 1: Disable writing to RTC	RW
b18	RTCRDP	RTC read protection	0: Allow read operation of RTC 1: Disable RTC read operation	RW
b17	BKSRAMWRP	BKSRAM write protection	0: Allow writing to Ret-SRAM 1: Disable writing to Ret-SRAM	RW
b16	BKSRAMRDP	BKSRAM read protection	0: Allow read operation of Ret-SRAM 1: Disable the read operation of Ret-SRAM	RW
b15	SWDTWRP	SWDT write protection	0: Allow writing to SWDT 1: Prohibit writing to SWDT	RW
b14	SWDTRDP	SWDT read protection	0: Allow SWDT read operation 1: Disable read operation on SWDT	RW
b13	WDTWRP	WDT write protection	0: Allow writing to WDT 1: Prohibit writing to WDT	RW
b12	WDTRDP	WDT read protection	0: Allow WDT read operation 1: Disable WDT read operation	RW
b11~b10	-	-	Reserved bit, read as 0, write 0 when writing	R
b9	EFCMCWRP	EFCM write protection	0: Allow register write operation to [9 Embedded Flash (EFM)] 1: Prohibit the register write operation to [9 Embedded Flash (EFM)]	RW
b8	EFCMCRDP	EFCM read protection	0: Allow the register read operation of [9 Embedded Flash (EFM)] 1: Prohibit the register read operation of [9 Embedded Flash (EFM)]	RW
b7	CRCWRP	CRC write protection	0: Allow writing to CRC 1: Prohibit writing to CRC	RW
b6	CRCRD	CRC read protection	0: Allow reading of CRC 1: Disable CRC read operation	RW
b5	TRNGWRP	TRNG write protection	0: Allow writing to TRNG 1: Prohibit writing to TRNG	RW

b4	TRNGRP	TRNG read protection	0: Allow TRNG read operation 1: Disable TRNG read operation	RW
b3	HASHWRP	HASH write protection	0: Allow writing to HASH 1: Forbid writing to HASH	RW
b2	HASHRDP	HASH read protection	0: Allow read operations on HASH 1: Disable HASH read operation	RW
b1	AESWRP	AES write protection	0: Allow writing to AES 1: Forbid writing to AES	RW
b0	AESRDP	AES read protection	0: Allow AES read operation 1: Disable AES read operation	RW

In the privileged mode, the object IP can be read and written without being affected by this register.

14 DMA Controller (DMA)

14.1 Introduction

DMA is used to transfer data between memory and peripheral functional modules. It can exchange data between memory, between memory and peripheral functional modules and between peripheral functional 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 8 independent channels, which can independently operate different DMA transfer functions
- For each channel, the source of the boot request is configured via a separate trigger source
- One block per request
- The data block is a minimum of one data and can be up to 1024 data
- The width of each data can be configured as 8bit, 16bit or 32bit
- Up to 65535 transmissions can be configured
- The source address and target address can be independently configured as fixed, auto-increment, auto-decrement, loop or jump with a specified offset
- Three types of interrupts can be generated: block transport complete interrupt, transport complete interrupt, and transport error interrupt. Each interrupt can be configured with a mask or not. Among them, the block transmission is completed, and the transmission completion can be used as an event output, which can be used as a trigger source for other peripheral modules
- Support for linked transmission to enable multiple blocks of data to be transmitted at a time
- Support channel reset triggered by external events
- You can set the module stop state to reduce power consumption when not in use

14.2 Module Diagram

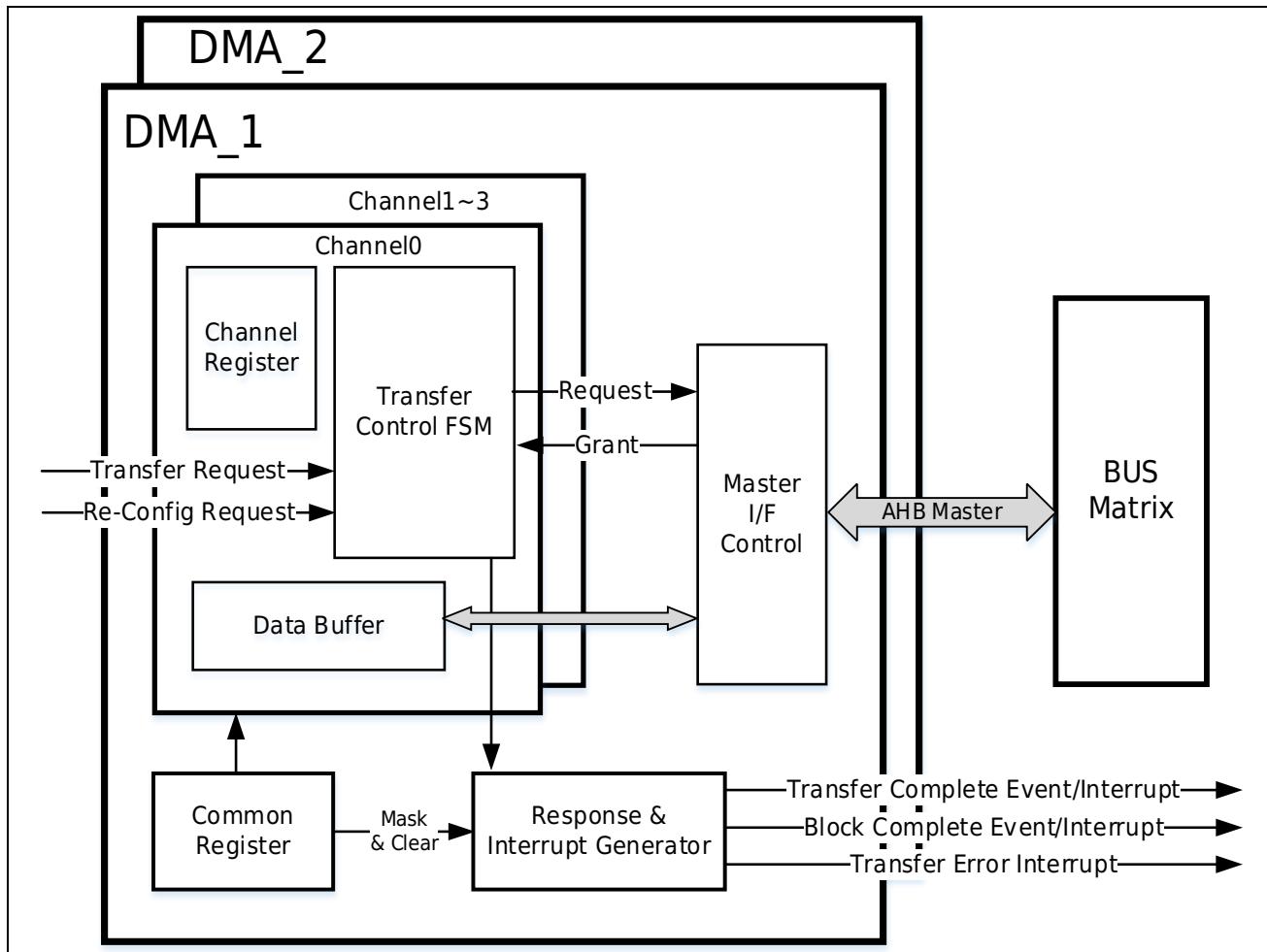


Figure 14-1 DMA Structure Diagram

14.3 Functional Description

14.3.1 Enable DMA controller

When using DMA, you need to write the register first to enable the DMA controller. The enabling method is to write the DMA_EN.EN bit of the DMA enable register.

Set DMA_EN.EN to 0 when DMA is not used or the chip needs to enter STOP mode. En Please confirm that register DMA_CHSTAT.DMAACT is 0 before writing 0 to ensure that DMA has completed all transfers.

14.3.2 Channel Selection and Channel Priority

Each DMA control unit contains 4 channels, and each channel can configure the transfer function independently.

The priority order of the 4 channels is: channel 0>channel 1>channel 2>channel 3.

When a DMA unit has multiple channel requests to transfer, it will be executed in priority order. But the channel that is already in transit will not be interrupted, the high priority channel will not start until the current channel transmission is complete.

14.3.3 Start DMA

The DMA is started by the request generated by the peripheral circuit. These requests are configured through the trigger source selection register DMA_TRGSELx (x=0~3), and the start request source of channel x can be configured. When the peripheral circuit generates a start request or the software writes the register to generate a start request, and the DMA transmission is enabled DMA_EN.EN=1, and the transmission channel is in the permitted state DMA_CHEN.CHEN[x]=1, Then the channel x transmission is started.

Before use, it is necessary to enable the peripheral circuit trigger function and DMA function enable position of the function clock control 0 register (FCG0).

14.3.4 Data Block

The amount of data transmitted by the DMA each time is represented by a 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 data width of each is determined by DMA_CHxCTL.HIZE.

14.3.5 Transmission Address Control

The source and destination addresses can be fixed, incremented, decremented, overloaded or discontinuous.

Fixed: The source address and destination address will be fixed during the transmission process.

Increment and decrement: The source ,destination addresses will jump forward or backward based on the HSIZE value after each transmission. For example, when HSIZE is 8bit, the address will increase/decrease by 1 each time, when it is 16bit, it will increase/decrease by 2 each time, and when it is 32bit, it will increase/decrease by 4 each time.

Overload: After transferring the specified amount of data, the source and destination addresses will return to the original address setting value. The amount of data that needs to be transferred before the address is reloaded, that is, the size of the repeated area is set by the register DMA_RPT.

Discontinuous address transfer: After transferring the specified amount of data, the source and destination addresses will skip the specified offset. The offset of the address jump and the amount of data to be transferred before the jump, that is, the size of the discontinuous area, are set by the register DMA_SNSEQCTL/DMA_DNSEQCTL. When the conditions of address overloading and discontinuous jump are met at the same time, address overloading is performed.

14.3.6 Number of Transmissions

The number of total data blocks transferred by DMA is set by the CNT bit of the data control register DMA_DTCTLx. The maximum number of transmissions can be set to 65535. Every time a data block is transmitted, the register value is reduced by 1. When the register value is reduced to 0, it means that all data transmission of this channel is completed, and the channel transmission permission bit DMA_CHEN.CHEN[x] is automatically cleared, and a transmission completion interrupt is generated. If the register is set to 0 at the beginning of the transmission, it means unlimited transmission, and each start requests to transmit a data block, but the channel transmission permission bit is not cleared, and the transmission completion interrupt will not be generated.

14.3.7 Interrupt and Event Signal Output

DMA controller can produce the following three interrupts:

Data Block Complete interrupt DMA_BTCx: Generated after completing a data block transfer.

Transmission complete interrupt DMA_TCx: Generated after completing the number of transmissions set by the register DMA_DTCTLx.CNT.

Transmission error interrupt DMA_ERR: When the start request overflows (that is, the channel triggers the start request again when the last request of the channel has not been responded), or when a bus error occurs during the transmission (such as accessing an illegal address or a protected address) An interrupt is generated, where in a bus error terminates the transfer immediately.

Except for the start request overflow error, the above interrupts can be enabled or disabled by setting the register DMA_CHxCTL.IE. In addition, all interrupts are equipped with independent MASK registers to mask the interrupts.

The above-mentioned DMA_BTCx, DMA_TCx interrupt can also be used as an event signal output, which can be used as a trigger source for other peripheral circuits. The event output is controlled by the MASK register, but not controlled by the interrupt per mission bit DMA_CHxCTL .IE

14.3.8 Chain Transmission

DMAcontroller has the function of linkage transmission. The chain transmission needs to configure the following eight registers, a total of eight words, called a descriptor (descriptor), which contains the source address, destination address, data control information, address control information, chain pointer and transmission control information of the chain transmission.

DMA_SARx
DMA_DARx
DMA_DTCTLx
DMA_RPTx
DMA_SNSEQCTLx
DMA_DNSEQCTLx
DMA_LLPx
DMA_CHxCTL

Where LLP is called a link-list Pointer, where the value represents the next descriptor's first address in the memory. When using chain transfer, first write LLPEN of the channel control register DMA_CHxCTLx to enable chain transfer, and write the descriptor information of the first transfer into the corresponding register. Descriptors for subsequent transfers are then initialized in memory in sequence. When it is necessary to end the chain transfer, set the LLPEN of DMA_CHxCTLx in the last descriptor of the channel to invalid, and the DMA controller will end the chain transfer after the transfer is completed.

When the last transfer of a descriptor ends,

If LLPEN=1, LLPRUN=0, BTC and TC interrupts will be generated according to the interrupt permission configuration, and the channel permission CHEN[x] will not be automatically cleared to 0. The next descriptor specified by LLP is loaded into the channel configuration register from the memory, waits for the next transmission request input, and starts the first transmission of the new descriptor.

If LLPEN=1, LLPRUN=1, BTC and TC interrupts will not be generated, channel permission CHEN[x] will not be automatically cleared to 0. The first transfer of a new descriptor begins directly after the next descriptor specified by the LLP is loaded from memory into the channel configuration register.

If LLPEN=0, the chain transmission ends, and BTC and TC interrupts are generated according to the interrupt permission configuration, and the channel permission CHEN[x] is automatically cleared to 0.

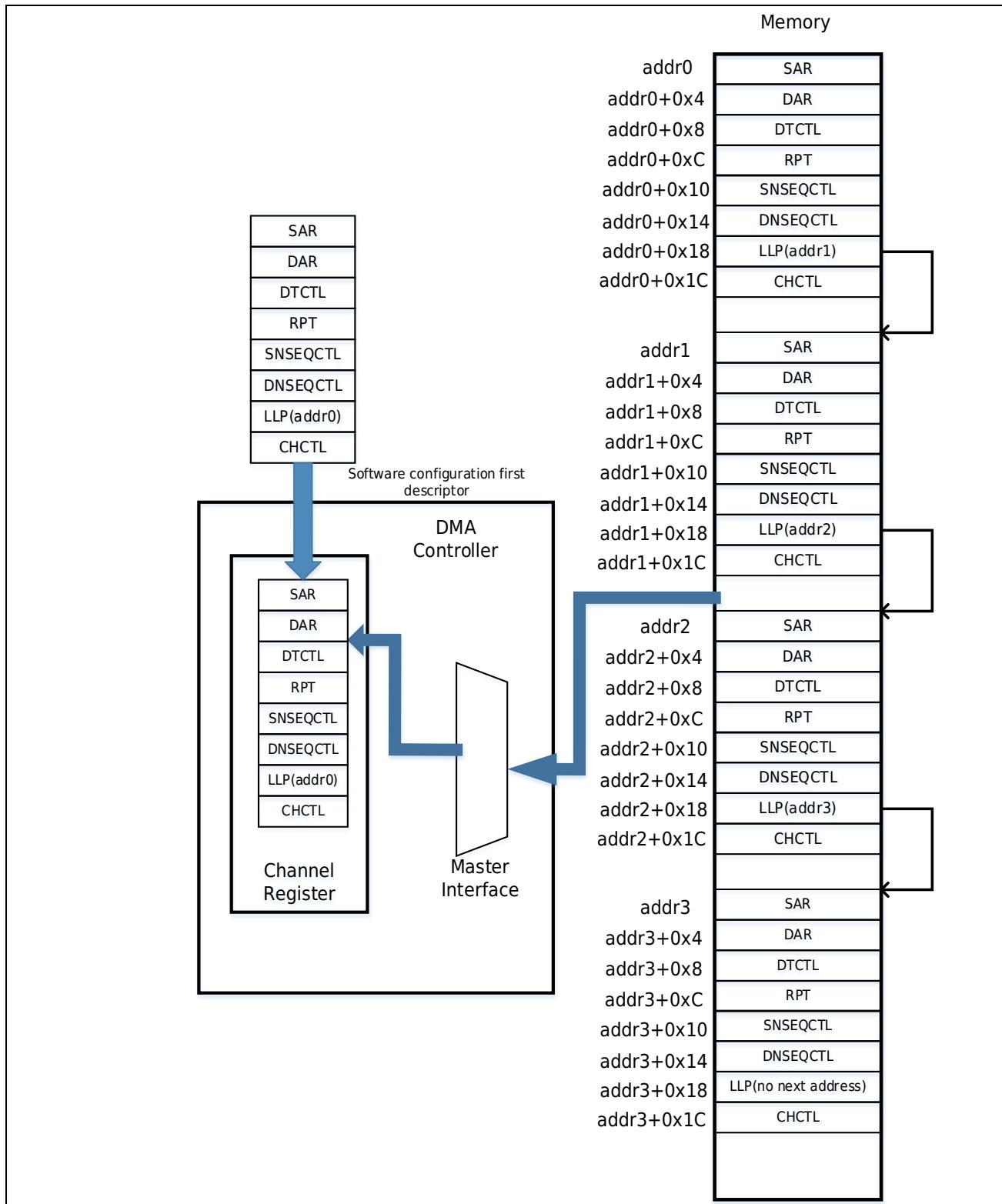


Figure 14-2 Chain transmission diagram

14.3.9 Discontinuous Address Transmission

Using discontinuous address transmission can realize source address and destination address to jump according to a certain offset after a certain amount of data is transmitted. The jump direction jumps forward or backward according to the settings of DMA_CHxCTL.SINC and DMA_CHxCTL.DINC. When in use, first set the channel control registers DMA_CHxCTL.SNSEQEN and DMA_CHxCTL.DNSEQEN to 1 to make the discontinuous address transfer valid. Then configure the discontinuous address transfer control registers DMA_SNSEQCTLx and DMA_DNSEQCTLx of the source device and the target device. The transmission process is shown in the following figure.

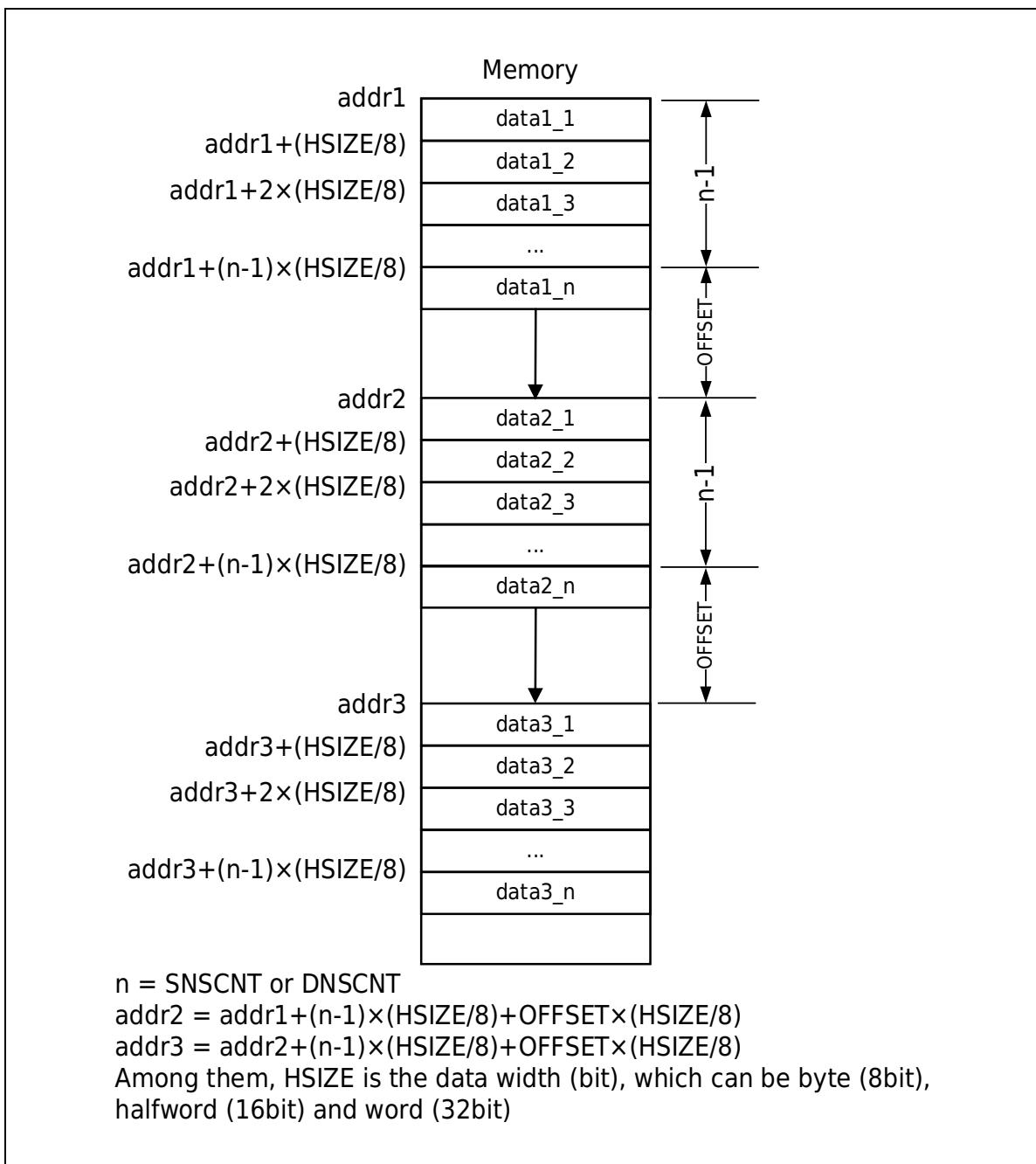


Figure 14-3 Schematic Diagram of Discontinuous Address Transmission

14.3.10 Channel Reset

The channel reset function refers to modifying the internal status register of the channel through the event request of the peripheral circuit, and reconfiguring the next data transmission mode. Set register DMA_RCFGCTL.RCFGGEN to 1 to allow channel reset. Select the reset request source through the trigger source selection register DMA_TRGSELRC. When the selected reset request source is input, the channel selected by the register DMA_RCFGCTL.RCFGCHS will be updated according to the specified method. A reset request only updates internal state and does not initiate the actual data transfer.

There are three ways to reset channels: chain pointer, discontinuous, and repeat.

When the chain pointer reset is selected, the descriptor and internal state of the channel are all updated to the new descriptor pointed to by the chain pointer LLP. Subsequent transfer requests are transferred according to the new descriptor.

When discontinuous, reload reset is selected, the internal state of the channel is updated as described in the table below.

Table 14-1 Channel Reset Description

Channel internal state	Reset method	
	Discontinuous	Heavy duty
Remaining Transfers Counter	Update to the value after the next address discontinuous jump occurs in the normal state	Update to the value after the next reload occurs in the normal state
Source/destination address for next transfer	Update to the first address of the next discontinuous transfer area	Update to the initial setting value of register DMA_SARx/DARx

Note:

- When the reset function is valid, the channel uses the registers DMA_RPTBx and DMA_SNSEQCTLBx, and DMA_DNSEQCTLBx controls the reload and discontinuous jump of the transfer address. Registers DMA_RPTx and DMA_SNSEQCTLx, DMA_DNSEQCTLx are invalid.

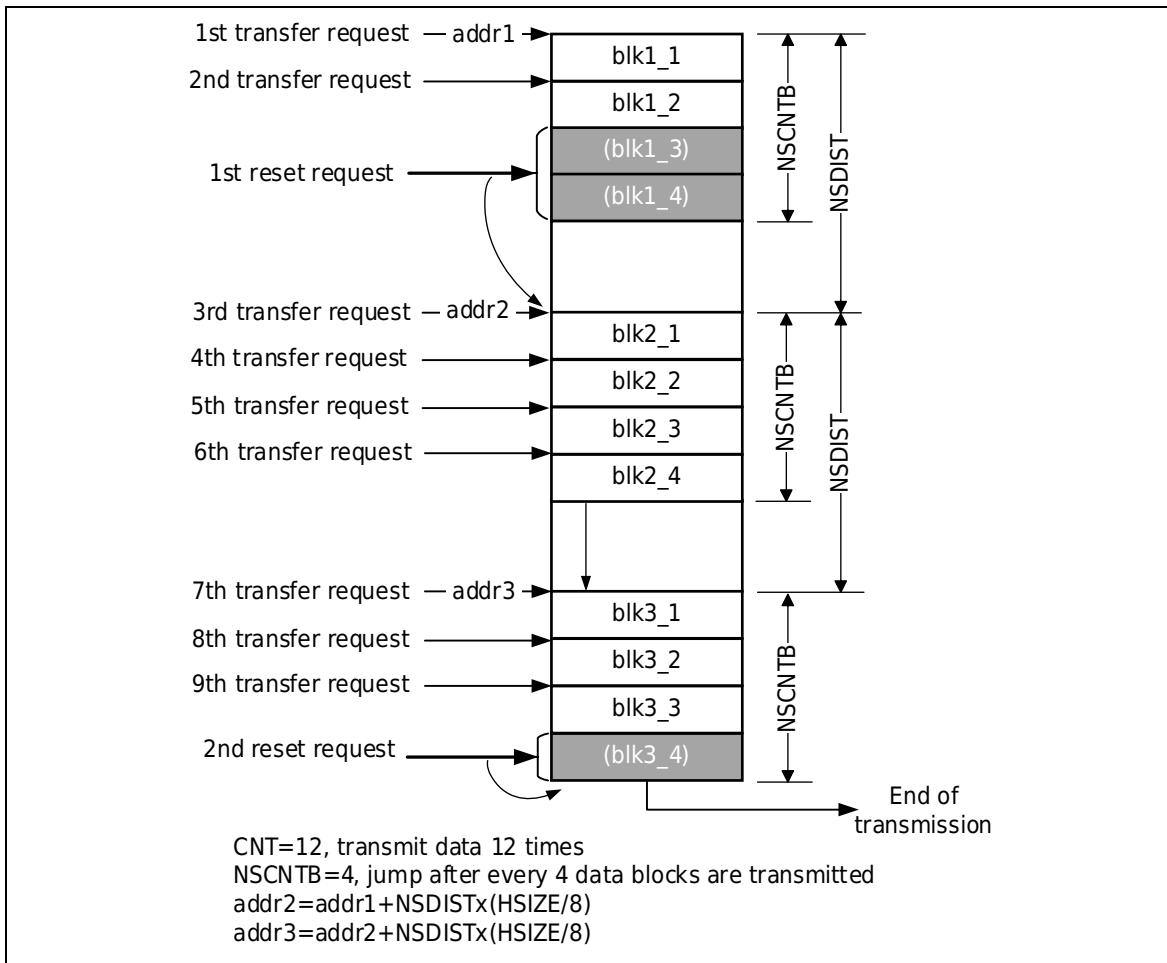


Figure 14-4 Schematic Diagram of Discontinuous Reset

Figure 14-4 In The DMA operation shown in Figure 14-4, each transfer request initiates the transfer of one block of data. After the first reset request occurs, the controller skips the data blocks blk1_3 and blk1_4, and the transmission address is updated to the first address of the next discontinuous area, which is addr2. After the second reset request occurs, the remaining number of transfers is updated to 0, that is, all data transfers are completed, the channel permission bit is automatically cleared to 0, and a transfer completion interrupt and event are generated.

14.3.11 Premature Interrupt of Transmission

During the transmission process, the channel enable register DMA_CHEN.CHENx remains valid. During non-chain transmission, the data control register DMA_DTCTLx is automatically set to invalid after the number of transmissions set by the data control register DMA_DTCTLx is complete ed. invalid. If the software writes DMA_CHEN.CHENx to 0 during the transmission, the DMA will terminate the transmission after completing the current data read and write.

Note:

- When the software writes the CHENx bit to 0 to terminate the transfer in advance, the DMA will not save the transfer status when it is terminated. Without resetting the state of the channel configuration register (descriptor), write CHENx 1 to enable the channel again. After the transmission request is input, the DMA will retransmit the terminated data block instead of resuming the breakpoint.

14.4 Application Examples

14.4.1 Memory-to-Memory Transmission

Target: Transfer 22 data from the RAM address 0x2000_0000 to 0x2000_1000 with a 32-bit data width.

1. Register setting

- Dma_EN.EN Write 1 Enables DMAcontroller
- Select a channel, such as channel 0, and configure the channel register to:
 - Write DMA_SAR0 configuration source address is SRAM area 0x2000_0000
 - Write DMA_DAR0 configuration source address is SRAM area 0x2000_1000
 - Write the DMA_DTCTL0 configuration data block size is 4, the number of transfers is 3, after each transfer one data block to produce block transfer complete interrupt, after 3 transfers complete one transfer complete interrupt
 - Write the DMA_RPT register to configure the size of the source address repetition area to be 6, that is, reload the initial source address after the transfer of 6 addresses is completed
 - Configure channel control register DMA_CH0CTL to achieve:
 - * Invalid link between source and destination
 - * Source address overload is valid, and destination address update is self-increasing
 - * Data width is word (32 bits)
 - * Interrupt enable is valid
 - Channel enable bit DMA_CHEN.CHEN0 write 1, enable channel 0
 - Configure trigger source controller DMA_TRGSEL0 and select software trigger as start request for DMA channel 0
- Write peripheral event software triggers registerINTSFTTRG STRG to 1, sends the first software startup request, DMA starts transmitting data

2. Transmission process

Since the size of the data block is 4, the first transmission begins when INTSFTTRG STRG is 1, and the number of transmissions is DMA_DTCTL0.CNT CH0CTL0 when a data block is transmitted.CNT minus 1, and produces a block transport complete interrupt, the software can continue to write INTSFTTRG STRG in the interrupt subroutine to start the second transport. In the second transmission, since the size of the source address repetition area is set to 6, the source address will be reloaded into the initial address 0x20000000 after the transmission of 2 addresses and continue to transmit the remaining 2 addresses. After the

second transfer is completed, the number of transfers DMA_DTCTL0.CNT is reduced by 1, and a block transfer completion interrupt is generated, and the software can continue to write INTSFTTRG STRG in the interrupt subroutine to start the third transfer. After the third transfer is completed, the number of transfers DMA_DTCTL0.CNT is reduced to 0, that is, this transfer is all completed, DMA generates a block transfer completion interrupt and a transfer completion interrupt, and the channel enable bit DMA_CHEN.CHENO will be automatically cleared.

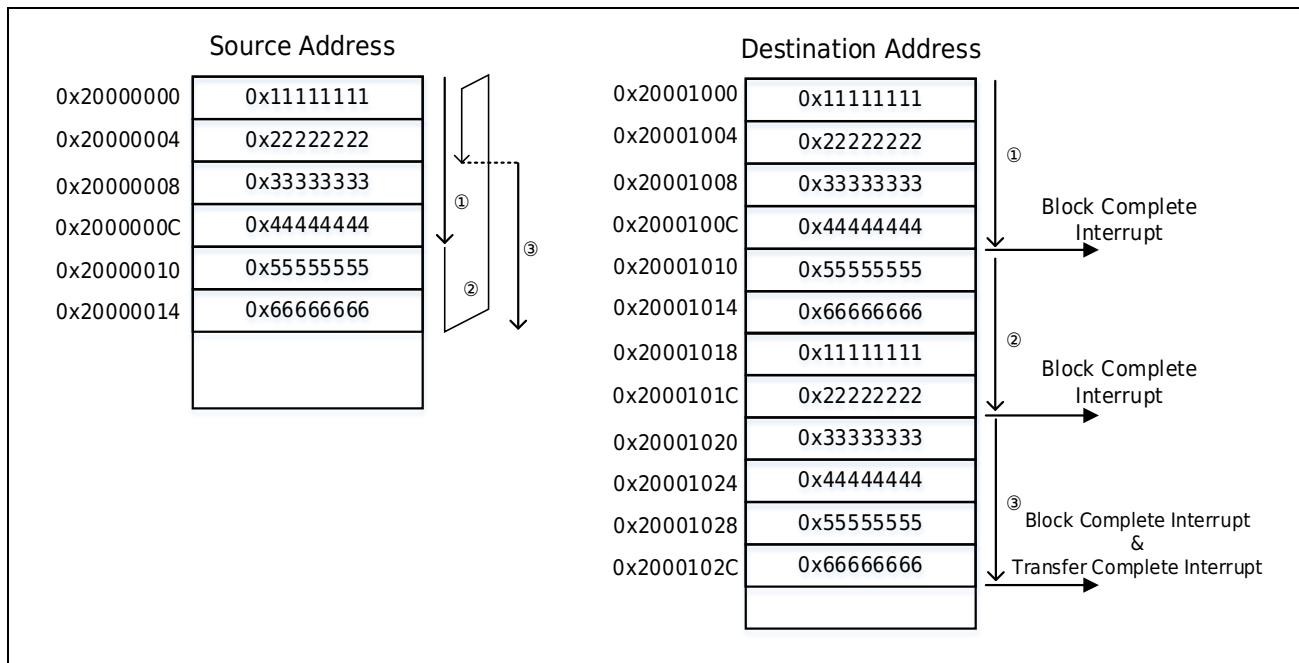


Figure 14-5 Application Example 1: Memory-to-Memory Transfer

14.4.2 Memory-to-Peripheral Transmission

Target: Transfer 10 pieces of data with a width of half-word from the RAM address 0x2000_0000 to the transmission buffer register of the communication module, each time the communication module completes the transmission request. When the last data is sent, DMA generates a transfer complete interrupt.

1. Register setting

- Dma_EN.EN Write 1 Enables DMAcontroller
- Configure the DMA_INTMSK register, mask the block transfer completion interrupt, and enable the transfer completion interrupt
- Select a channel to configure the channel register, for example, select channel 0
 - Write DMA_SAR0 configuration source address is SRAM area 0x2000_0000
 - Write DMA_DAR0 Configure source address to register address of peripheral circuit 0x4000_0000
 - Write DMA_DTCTL0Configuration data block size is 1, number of transfers is 10, each

transfer request is delivered once, one data at a time.

- Configure channel control register DMA_CH0CTL to achieve:
 - * Invalid link between source and destination
 - * Source address update mode is self-increment, target address is fixed
 - * Source and destination address data width is half word (16 bits)
 - * Interrupt enable is valid
- Configure the trigger source controller DMA_TRGSEL0, and select the event that the send register of the communication module is empty as the start request of DMA channel 0
- Channel enable bit DMA_CHEN.CHEN0 write 1, enable channel 0

2. Transmission process

After the channel is enabled, DMA waits for the transmission request from the communication module. After the transmission request is generated, the DMA transmits the data in the RAM to the transmission buffer register of the communication module and waits for the second transmission request from the communication module. Because the block transmission is shielded to complete interrupt, the DMA does not generate interrupt request at this time. When all 10 data transfers are completed, DMA generates a transfer completion interrupt, and the channel enable bit DMA_CHEN.CHEN0 will be automatically cleared.

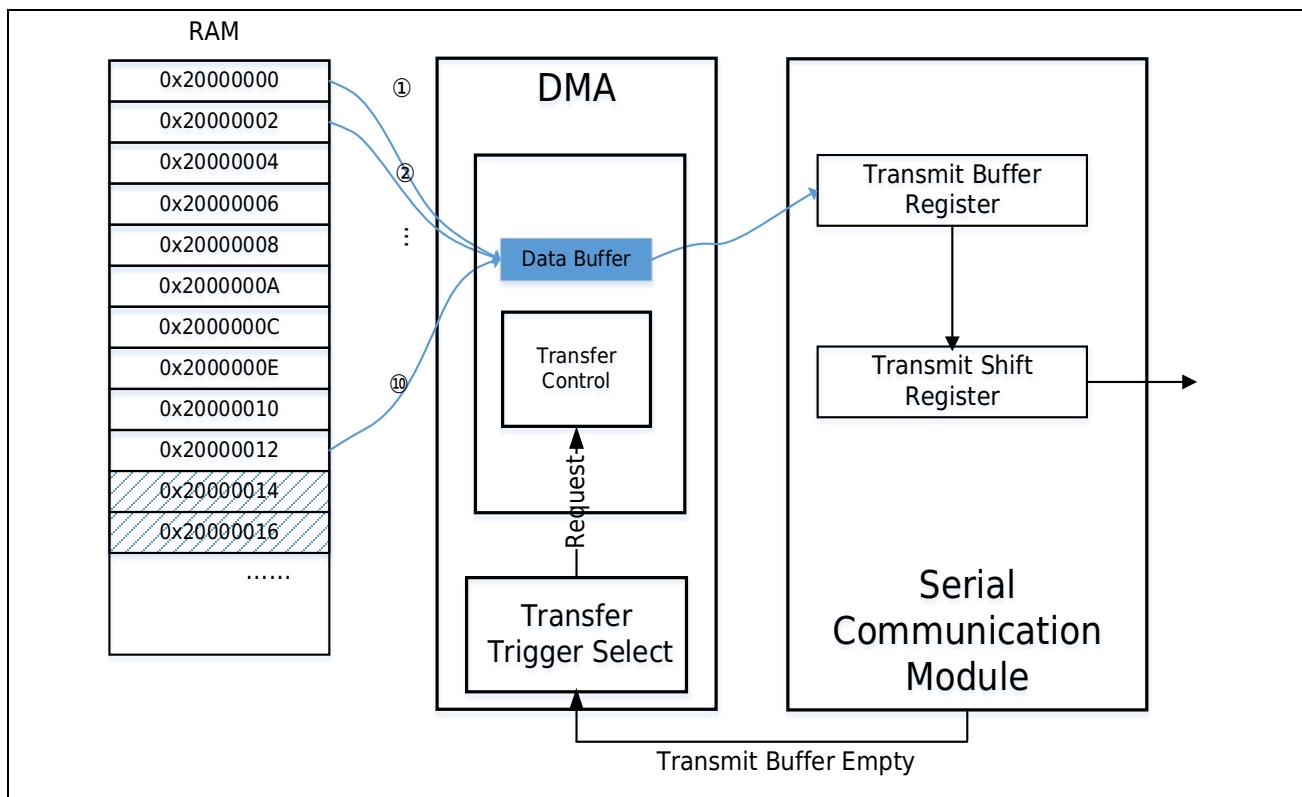


Figure 14-6 Application Example 2: Transfer from Memory to Peripheral Circuits

14.4.3 Memory-to-Memory Linkage

1. Register setting

- Dma_EN.EN Write 1 Enables DMAcontroller
- Select a channel, configure the channel register, for example, select channel 0, configure the descriptor for the first transmission (descriptor 0)
 - Write DMA_SAR0 configuration source address is SRAM area 0x2000_0000
 - Write DMA_DAR0 configuration source address is SRAM area 0x2000_1000
 - Write DMA_DTCTL0 configuration data block size is 10
 - Write the address 0x20002000 of the second descriptor (descriptor1) in the chain pointer register DMA_LLPO
 - Configure the channel control register DMA_CH0CTL, and configure the transmission parameters of the first data block to achieve:
 - * Linkage transmission efficiency
 - * Chain transmission mode is to start the next transmission directly
 - * Source and destination address updates are self-incremented
 - * Data width is word (32 bits)
 - * Interrupt enable invalid
- Configure the descriptor (descriptor 1) for the second transmission in the 0x2000_2000 address of the RAM space, including:
 - 0x2000_2000 writes 32-bit data 0x2000_0100, which is the source address of the second transmission
 - 0x2000_2004 writes 32-bit data 0x2000_1100, which is the target address of the second transmission
 - 0x2000_2008 Configuration data block size is 20
 - 0x2000_2018 writes 32-bit data 0x2000_2100, which is the address of the descriptor (descriptor 2) of the third transmission
 - 0x2000_021C writes control data for the second transmission, which implements:
 - * Linkage transmission efficiency
 - * Chain transmission mode is to start the next transmission directly
 - * Source and destination address updates are self-incremented
 - * Data width is half word (16 bits)

- * Interrupt enable invalid
- Configure the descriptor (descriptor 2) for the third transmission in the 0x2000_2100 address of the RAM space, including:
 - 0x2000_2100 writes 32-bit data 0x2000_0200, which is the source address of the third transmission
 - 0x2000_2104 writes 32-bit data 0x2000_1200, which is the target address for the third transmission
 - 0x2000_2108 Configuration data block size is 40
 - 0x2000_2118 writes 32-bit data 0x0, representing the last transmission of the link
 - 0x2000_211C write to the third transmission of the control data implementation:
 - * Chain transmission invalid
 - * Source and destination address updates are self-incremented
 - * Data width in bytes (8 bits)
 - * Interrupt enable is valid
- Channel enable bit DMA_CHEN.CHEN0 write 1, enable channel 0
- Configure the transfer start trigger source selection register DMA_TRGSEL0, and select the software trigger as the start request of DMA channel 0
- Write software triggers registerINTSFTTRG STRG to 1, sends a startup request, DMA starts transmitting data

2. Transmission process

The software starts DMA and starts transmission. After the first transmission is complete, the DMA reads the descriptor (descriptor 1) of the second transmission into the channel register because the linked transmission mode is set to start the next transmission directly and the interrupt is invalid. Start the second transfer directly according to the parameters configured by the descriptor. After the second transmission is complete, read the descriptor (descriptor 2) of the third transmission into the channel register. Starts the third transfer according to the parameters configured by the descriptor. After the third transmission is complete, the last transmission is linked according to the configuration information, and because the interrupt enable is valid, DMA will generate a transmission finish interrupt and clear the channel enable bit DMA _ CHEN.CHEN0.

14.5 Register Description

DMA_1 BASE_ADDR:0x4005_3000

DMA_2 BASE_ADDR:0x4005_3400

Register name	Symbol	Offset address	Bit width	Reset value
DMA enable register	DMA_EN	0x00	32	0x0000_0000
Interrupt state register0	DMA_INTSTAT0	0x04	32	0x0000_0000
Interrupt state register1	DMA_INTSTAT1	0x08	32	0x0000_0000
Interrupt shielded register0	DMA_INTMASK0	0x0C	32	0x0000_0000
Interrupt Shield register1	DMA_INTMASK1	0x10	32	0x0000_0000
Interrupt Reset register0	DMA_INTCLR0	0x14	32	0x0000_0000
Interrupt Reset register1	DMA_INTCLR1	0x18	32	0x0000_0000
Channel enable register	DMA_CHEN	0x1C	32	0x0000_0000
Transfer Request Status Register	DMA_REQSTAT	0x20	32	0x0000_0000
In-transit channel monitoring register	DMA_CHSTAT	0x24	32	0x0000_0000
Channel Reset Control Register	DMA_RCFGCTL	0x2c	32	0x0000_0000
Transmission source address register	DMA_SARx *1	0x40+0x40*x	32	0x0000_0000
Transmission destination address register	DMA_DARx	0x44+0x40*x	32	0x0000_0000
Data control register	DMA_DTCTLx	0x48+0x40*x	32	0x0000_0001
Repeat Region Size Register	DMA_RPTx	0x4C+0x40*x	32	0x0000_0000
Repeat Region Size Register B	DMA_RPTBx			
Source Device Discontinuous Address Transfer Control Register	DMA_SNSEQCTLx	0x50+0x40*x	32	0x0000_0000
Source Device Discontinuous Address Transfer Control Register B	DMA_SNSEQCTLBx			
Target Device Discontinuous Address Transfer Control Register	DMA_DNSEQCTLx	0x54+0x40*x	32	0x0000_0000
Target Device Discontinuous Address Transfer Control Register B	DMA_DNSEQCTBx			
Chain pointer register	DMA_LLpx	0x58+0x40*x	32	0x0000_0000
Channel control register	DMA_CHxCTL	0x5C+0x40*x	32	0x0000_1000
Transfer Source Address Monitor Register	DMA_MONSARx	0x60+0x40*x	32	0x0000_0000
Transfer Destination Address Monitor Register	DMA_MONDARx	0x64+0x40*x	32	0x0000_0000
Data Control Monitor Register	DMA_MONDTCTLx	0x68+0x40*x	32	0x0000_0001
Repeat Region Counter Monitor Register	DMA_MONRPTx	0x6C+0x40*x	32	0x0000_0000
Source Device Discontinuous Transfer Counter Monitor Register	DMA_MONSNSEQCTLx	0x70+0x40*x	32	0x0000_0000
Target Device Discontinuous Transfer Counter Monitor Register	DMA_MONDNSEQCTLx	0x74+0x40*x	32	0x0000_0000

14.5.1 DMA Enable Register (DMA _ EN)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

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

Bit	Marking	Place name	Function	Read and write
b31-b1	Reserved	-	Read as "0", write as "0"	R/W
b0	EN	DMA enable bit	0: DMA invalid 1: DMA enable	R/W

14.5.2 Interrupt State Register0 (DMA _ INTSTAT0)

Reset value: 0x0000_0000

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

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

Bit	Marking	Place name	Function	Read and write
b31-b20	Reserved	-	Read as "0", write as "0"	R/W
b19-b16	REQERR[3:0]	Transmission request overflow error interrupt bit	0: No transmission request overflow error occurred on the channel 1: The transmission request overflow error occurred on the channel, that is, the transmission request came again when the last request was still waiting	R
b15-b4	Reserved	-	Read as "0", write as "0"	R/W
b3-b0	TRNERR[3:0]	Transmission error interrupt bit	0: No transmission error occurred on the channel 1: The channel has a transmission error	R

14.5.3 Interrupt State Register0 (DMA _ INTSTAT1)

Reset value: 0x0000_0000

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

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

Bit	Marking	Place name	Function	Read and write
b31-b20	Reserved	-	Read as "0", write as "0"	R/W
b19-b16	BTC[3:0]	Block transport complete interrupt bit	The interrupt occurs after a block of data is transmitted 0: No block transport interrupt occurs on the channel 1: Block transport interrupt of this channel	R
b15-b4	Reserved	-	Read as "0", write as "0"	R/W
b3-b0	TC[3:0]	Transport completion interrupt bit	The interrupt occurs after the number of transmissions set by registerMA_CNTx is completed 0: The channel was not interrupted 1: The channel is interrupted	R

14.5.4 Interrupt Shield Register (DMA _ INTMASK0)

Reset value: 0x0000_0000

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

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

Bit	Marking	Place name	Function	Read and write
b31-b20	Reserved	-	Read as "0", write as "0"	R/W
b19-b16	MSKREQERR[3:0]	Transmission request overflow interrupt mask	0: Nomaskable transport request overflow interrupt 1: Shielded transport request overflow interrupt	R/W
b15-b4	Reserved	-	Read as "0", write as "0"	R/W
b3-b0	MSKTRNERR[3:0]	Transmission error interrupt mask	0: Nomaskable transport error interrupt 1: Mask transmission error interrupt	R/W

14.5.5 Interrupt Shield Register (DMA _ INTMASK1)

Reset value: 0x0000_0000

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

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

Bit	Marking	Place name	Function	Read and write
b31-b20	Reserved	-	Read as "0", write as "0"	R/W
b19-b16	MSKBTC[3:0]	Block transport complete interrupt mask	0: Nomaskable block transport complete interrupt 1: Block transport complete interrupt	R/W
b15-b4	Reserved	-	Read as "0", write as "0"	R/W
b3-b0	MSKTC[3:0]	Transport complete interrupt mask	0: Nomaskable transport complete interrupt 1: Shield transport complete interrupt	R/W

14.5.6 Interrupt Reset Register (DMA _ INTCLR0)

Reset value: 0x0000_0000

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

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

Bit	Marking	Place name	Function	Read and write
b31-b20	Reserved	-	Read as "0", write as "0"	R/W
b19-b16	CLRREQERR[3:0]	Transmission request overflow error interrupt reset	Write 0 has no effect. Write 1 Resets the transmission request overflow error interrupt status bit. Read always 0	W
b15-b4	Reserved	-	Read as "0", write as "0"	R/W
b3-b0	CLRTRNERR[3:0]	Transmission error interrupt reset	Write 0 has no effect, write 1 reset transmission error interrupt status bit Read always 0	W

14.5.7 Interrupt Reset Register (DMA _ INTCLR1)

Reset value: 0x0000_0000

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

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

Bit	Marking	Place name	Function	Read and write
b31-b20	Reserved	-	Read as "0", write as "0"	R/W
b19-b16	CLRBTC[3:0]	Block transport complete interrupt reset	Write 0 has no effect, write 1 reset block transport complete interrupt status bit Read always 0	W
b15-b4	Reserved	-	Read as "0", write as "0"	R/W
b3-b0	CLRTC[3:0]	Transmission complete interrupt reset	Write 0 has no effect. Write 1 resets the transmission to finish terminating the status bit. Read always 0	W

14.5.8 Channel Enable Register (DMA _ CHEN)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

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

Bit	Marking	Place name	Function	Read and write
b31-b4	Reserved	-	Read as "0", write as "0"	R/W
b3-b0	CHEN[3:0]	Channel enable bit	Each bit corresponds to a channel. To use this channel, set this bit to 1. The enable bit remains at 1 during transmission, and it will be automatically cleared to zero after the number of transmissions set by the transmission times register DMA_DTCTLx.CNT is completed. If DMA_DTCTLx.CNT is set to 0, it will not be automatically cleared after the transfer is completed, that is, unlimited transfers. 0: Invalid channel 1: The channel is valid	R/W

14.5.9 Channel Reset Control Register (DMA_RCFGCTL)

Reset value: 0x0000_0000

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	Marking	Place name	Function	Read and write
b31-b22	Reserved	-	Read as "0", write as "0"	R/W
b21-b20	CNTMD[1:0]	Remaining transfer count counter reset method	00: remain unchanged 01: By source address 10, 11: by destination address When the source address mode is selected, and the source address selects discontinuous reset, the remaining transfer times counter is updated to the state after the specified transfer times of DMA_SNSEQCTLBx.SNSCNTB; when the source address is repeated, the remaining transfer times counter is updated to DMA_RPTBx. SRPTB The state after the specified number of transfers. When the source address selection is held, the remaining transfer times counter also remains unchanged. When the target address mode is selected, it is similar to the source address mode.	R/W
b19-b18	DARMD[1:0]	Target address reset method	00: remain unchanged 01: Discontinuous reset The destination address for the next transfer is updated to addr_base + (DNSDIST x HSIZE(bit)/8) Among them: addr_base indicates the first address of the current discontinuous transmission area 10, 11: Repeated reset The target address of the next transfer is updated to the initial setting value of the DMA_DARx register. Note: It is only allowed to set DARMD[1:0] to 01 when the target address discontinuous transfer of this channel is valid (DMA_CHxCTL.DNSEQEN=1). It is only allowed to set DARMD[1] to 1 when the target address reload of this channel is valid (DMA_CHxCTL.DRPTEN=1).	R/W
b17-b16	SARMD[1:0]	Source address reset method	00: remain unchanged 01: Discontinuous reset The source address of the next transfer is updated to addr_base + (SNSDIST x HSIZE(bit)/8) Among them: addr_base indicates the first address of the current discontinuous transmission area 10, 11: Repeated reset The source address of the next transfer is updated to the initial setting value of the DMA_SARx register. Note: It is only allowed to set SARMD[1:0] to 01 when the source address discontinuous transmission of this channel is valid (DMA_CHxCTL.SNSEQEN=1). It is only allowed to set SARMD[1] to 1 when the source address reload of this channel is valid (DMA_CHxCTL.SRPTEN=1).	R/W
b15-b12	Reserved	-	Read as "0", write as "0"	R/W
b11-b8	RCFGCHS[3:0]	Reset channel selection	0x0: channel 0 0x1: channel 1 0x2: channel 2 0x3: channel 3 Others: Reserved, setting prohibited	R/W
b7-b2	Reserved	-	Read as "0", write as "0"	R/W
b1	RCFGLLP	Chain Pointer Channel Reset	0: Chain pointer reset is invalid 1: Chain pointer reset is valid Note: When RCFGPLL is set to 1, the channel will reload the new descriptor in the memory, so bit16-bit21 of this register are all invalid.	R/W
b0	RCFGEN	Channel Reset Permission	0: Disable event-triggered channel configuration register forced update 1: Enable event-triggered channel configuration register forced update	R/W

Note:

- Please set this register when DMA_EN.EN is 0, this register must be set before the first transmission of the reset channel.

14.5.10 Transfer Request Status Register (DMA_REQSTAT)

Reset value: 0x0000_0000

b31	b3 0	b29	b28	b2 7	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RCFGREQ

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

Bit	Marking	Place name	Function	Read and write
b31-b17	Reserved	-	Read as "0", write as "0"	R/W
b16	RCFGREQ	Channel reset request flag	When the external reset request input is set to 1, Cleared to 0 when channel reset is enabled, or when channel reset is disabled. 0: No channel reset request 1: There is a channel reset request	R
b15-b4	Reserved	-	Read as "0", write as "0"	R/W
b3~b0	CHREQ[3:0]	channel transfer request flag	Each bit corresponds to a channel. When the external transmission request is input, the corresponding position is 1, When the channel transmission starts, or a transmission error occurs, or the transmission permission bit (DMAEN or CHEN[x]) is written to 0, this bit is cleared to 0. When this bit is in 1 state, the transmission request of this channel is input again, and a transmission request overflow error occurs, and the second request is ignored. 0: There is no transmission request for this channel 1: The channel has a transfer request	R

14.5.11 Channel State Observation Register (DMA _ CHSTAT)

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
-	-	-	-	-	-	-	-							CHACT[3:0]	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RCFGACT	DMAACT
Bit	Marking		Place name				Function								Read and write
b31-b20	Reserved		-				Read as "0", write as "0"								R/W
b19-b16	CHACT[3:0]		Channel monitoring bit				Each bit corresponds to a channel. 0: The channel is idle 1: The channel is in motion								R
b15-b2	Reserved		-				Read as "0", write as "0"								R/W
b1	RCFGACT		DMA channel reset action monitoring bit				0: DMA is not in channel reset action 1: DMA is in channel reset action								R
b0	DMAACT		Monitoring bit in DMA action				0: DMA not in transmission 1: DMA in transmission								R

14.5.12 Transmission Source Address Register (DMA _ SARx) (x = 0 ~ 3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SAR[31:16]															
SAR[15:0]															
Bit	Marking		Place name				Function								Read and write
b31-b0	SAR[31:0]		Transmission source address				Set source address Note: - When the transmission data width is 16 bits, that is, DMA _ CHxctl.HSIZE = 01, SAR [0] is invalid. When the transmission data width is 32 bits, that is, DMA _ CHxCTL.HSIZE = 1x, SAR [1: 0] is invalid.								R/W

14.5.13 Transport Destination Address Register (DMA_DARx) ($x = 0 \sim 3$)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DAR[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31-b0	DAR[31:0]	Transmission destination address	Set transmission destination address Note: - When the transmission data width is 16 bits, that is, DMA_CHxCTL.HSIZE = 01, DAR [0] is invalid. When the transmission data width is 32 bits, that is, DMA_CHxCTL.HSIZE = 1x, DAR [1: 0] is invalid.	R/W

14.5.14 Data Control Register (DMA_DTCTLx) ($x=0\sim 3$)

Reset value: 0x0000_0001

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CNT[15:0]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	BLKSIZE[9:0]								

Bit	Marking	Place name	Function	Read and write
b31-b16	CNT[15:0]	Number of transmissions	The total number of transfers, each request starts the transfer of a data block, the number of transfers is decremented by 1 when it is completed, and a transfer completion interrupt occurs when it is reduced to 0. If it is set to 0, it means unlimited transmission. Each time a data block is requested to be transmitted, the transmission times counter will remain at 0 when it is completed, and no transmission 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 pieces of data can be configured. The width of each data is determined by the HSIZE bit of the DMA_CHxCTLRegister. If the register value is set to 1, 1 data is transmitted each time, and if it is set to 0, 1024 data are transmitted each time.	R/W

14.5.15 Repeat Region Size Register (DMA_RPTx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	DRPT[9:0]

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SRPT[9:0]

Bit	Marking	Place name	Function	Read and write
b31-b26	Reserved	-	Read as "0", write as "0"	R/W
b25-b16	DRPT[9:0]	Destination address repeat area size	Set target address repeat area size The target device reloads the target address to the initial setting value of the DMA_DARx register after each DRPT data transfer. If the register is set to 10, the address will be reloaded after every 10 data transmissions, and if it is set to 0, the address will be reloaded after every 1024 data transmissions. The width of each data is determined by the HSIZE bit of the DMA_CHxCTLregister.	R/W
b15-b10	Reserved	-	Read as "0", write as "0"	R/W
b9-b0	SRPT[9:0]	Source Address Duplication Area Size	Set the source address repeat area size The source device reloads the source address to the initial setting value of the DMA_SARx register after each transmission of SRPT data. If the register is set to 10, the address will be reloaded after every 10 data transmissions, and if it is set to 0, the address will be reloaded after every 1024 data transmissions. The width of each data is determined by the HSIZE bit of the DMA_CHxCTLregister.	R/W

This register configures the size of the source and destination address duplication region. The use of repeated address transmission needs to configure the SRPTEN/DRPEN bit of the DMA_CHxCTL register to be valid, and configure the SINC/DINC bit of the DMA_CHxCTL register to make the address update mode self-increment or self-decrement. If it is fixed, the address reload function is invalid.

The two registers DMA_RPTx and DMA_RPTBx share the same address and are used to define the size of the repeated area. Which one to use depends on whether the channel has reset enabled or not. DMA_RPTx is used when reset is not enabled, and DMA_PRTBx is used when reset is enabled.

14.5.16 Repeat area size register B (DMA_RPTBx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	DRPT[9:0]

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SRPT[9:0]

Bit	Marking	Place name	Function	Read and write
b31-b26	Reserved	-	Read as "0", write as "0"	R/W
b25-b16	DRPTB[9:0]	Destination address repeat area size	Set target address repeat area size The target device reloads the target address to the initial setting value of the DMA_DARx register after each transmission of DRPTB data blocks. The data block size is determined by DMA_DTCTLx.BLKSIZE and DMA_CHxCTL.HSIZE.	R/W
b15-b10	Reserved	-	Read as "0", write as "0"	R/W
b9-b0	SRPTB[9:0]	Source Address Duplication Area Size	Set the source address repeat area size The source device reloads the source address to the initial setting value of the DMA_SARx register after each transmission of SRPTB data blocks. The data block size is determined by DMA_DTCTLx.BLKSIZE and DMA_CHxCTL.HSIZE.	R/W

This register configures the size of the source and destination address duplication region. The use of repeated address transmission needs to configure the SRPTEN/DRPEN bit of the DMA_CHxCTL register to be valid, and configure the SINC/DINC bit of the DMA_CHxCTL register to make the address update mode self-increment or self-decrement. If it is fixed, the repeated address transmission function is invalid.

The two registers DMA_RPTx and DMA_RPTBx share the same address and are used to define the size of the repeated area. Which one to use depends on whether the channel has reset enabled or not. DMA_RPTx is used when reset is not enabled, and DMA_RPTBx is used when reset is enabled.

14.5.17 Source Device Discontinuous Address Transfer Control Register (DMA_SNSEQCTLx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SNSCNT[11:0]												SOFFSET[19:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SOFFSET[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31-b20	SNSCNT[11:0]	The amount of data redirected by the source address	Set the amount of data transferred before the source address jumps. After the source device transmits SNSCNT data, the source address jumps according to the offset specified by SOFFSET. If the register is set to 10, the address will jump after every 10 data transmissions, if it is set to 0, the address will jump after every 4096 data transmissions.	R/W											
b19-b0	SOFFSET[19:0]	The address offset of the source address jump	When discontinuous address transmission, set the offset of the source address jump. The offset is relative to the current transfer address, that is, the last transfer address before the jump. The jump direction jumps forward or backward according to the value of the channel control register DMA_CHxCTL.SINC. reference imageFigure 14-3. When DMA_CHxCTL.SINC is set to fixed address, discontinuous address transfer is invalid. The jump address will be calculated according to the number of bits set by the data width (DMA_CHxCTL.HSIZE) and the value of SOFFSET. Address offset=SOFFSET×(HSIZE(bit)/8) For example, when SOFFSET is set to 10 and HSIZE is a word (32bit), the address offset is $10 \times 4 = 40$; if HSIZE is a halfword (16bit), the offset is $10 \times 2 = 20$; if HSIZE is Byte (8bit), the offset is $10 \times 1 = 10$. The source address of the next transfer = the source address of the current transfer ± address offset	R/W											

The discontinuous transmission of the source device needs to configure the SNSEQEN bit of the DMA_CHxCTL register to be valid, and configure the SINC bit of the DMA_CHxCTL register to make the address update mode be self-increment or self-decrement.

The two registers DMA_SNSEQCTLx, DMA_SNSEQCTLBx share the same address and are used to define discontinuous transmission. Which one to use depends on whether the channel has reset enabled or not. Use DMA_SNSEQCTLx when reset is not enabled, use DMA_SNSEQCTLBx when reset is enabled.

14.5.18 Source Device Discontinuous Address Transfer Control Register B (DMA_SNSEQCTLBx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SNSCNTB[11:0]												SNSDIST[19:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SNSDIST[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31-b20	SNSCNTB[11:0]	The amount of data redirected by the source address	Set the amount of data transferred before the source address jumps. After the source device transmits SNSCNTB data blocks, the source address jumps according to the address interval specified by SNSDIST. The data block size is determined by DMA_DCTRLx.BLKSIZE and DMA_CHxCTL.HSIZE.	R/W											
b19-b0	SNSDIST[19:0]	Source discontinuous area address spacing	When discontinuous address transmission, set the distance between two discontinuous areas of the source device. The jump direction jumps forward or backward according to the value of the channel control register DMA_CHxCTL.SINC. reference imageFigure 14-4. When DMA_CHxCTL.SINC is set to fixed address, discontinuous address transfer is invalid. The address spacing will be calculated based on the number of bits set by the data width (DMA_CHxCTL.HSIZE) and the value of SNSDIST. Address spacing=SNSDIST×(HSIZE(bit)/8) For example, when SNSDIST is set to 10 and HSIZE is word (32bit), the address spacing is $10 \times 4 = 40$; if HSIZE is halfword (16bit), the spacing is $10 \times 2 = 20$; if HSIZE is byte (8bit), the spacing is $10 \times 1 = 10$. The source address of the next transmission = the first address of the current source discontinuous area ± address distance	R/W											

The discontinuous transmission of the source device needs to configure the SNSEQEN bit of the DMA_CHxCTL register to be valid, and configure the SINC bit of the DMA_CHxCTL register to make the address update mode be self-increment or self-decrement.

The two registers DMA_SNSEQCTLx, DMA_SNSEQCTLBx share the same address and are used to define discontinuous transmission. Which one to use depends on whether the channel has reset enabled or not. DMA_SNSEQCTLx is used when reset is not enabled, and DMA_SNSEQCTLBx is used when reset is enabled.

14.5.19 Target Device Discontinuous Address Transfer Control Register (DMA_DNSEQCTLx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DNSEQNT[11:0]												DOFFSET[19:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DOFFSET[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31-b20	DNSEQNT[11:0]	The amount of data that the target address jumps to	Set the amount of data transferred before the target address jumps. After the target device transmits DNSEQNT data, the target address jumps according to the offset specified by DOFFSET. If the register is set to 10, the address will jump after every 10 data transmissions, if it is set to 0, the address will jump after every 4096 data transmissions.	R/W											
b19-b0	DOFFSET[19:0]	The address offset of the target address jump	When discontinuous address transfer, set the offset of the target address jump. The offset is relative to the current transfer address, that is, the last transfer address before the jump. The direction of the jump jumps forward or backward according to the value of the channel control register DMA_CHxCTL.DINC. reference imageFigure 14-3. When DMA_CHxCTL.DINC is set to fixed address, discontinuous address transfer will be invalid. The jump address will be calculated according to the number of bits set by the data width (DMA_CHxCTL.HSIZE) and the value of DOFFSET. Address offset=DOFFSET×(HSIZE(bit)/8) For example, when DOFFSET is set to 10 and HSIZE is word (32bit), the address offset is $10 \times 4 = 40$; if HSIZE is halfword (16bit), the offset is $10 \times 2 = 20$; if HSIZE is Byte (8bit), the offset is $10 \times 1 = 10$. The target address of the next transfer = the target address of the current transfer ± address offset	R/W											

The discontinuous transmission of the target device needs to configure the DNSEQEN bit of the DMA_CHxCTL register to be valid, and configure the DINC bit of the DMA_CHxCTL register to make the address update mode auto-increment or auto-decrement.

The two registers DMA_DNSEQCTLx, DMA_DNSEQCTLBx share the same address and are used to define discontinuous transmission. Which one to use depends on whether the channel has reset enabled or not. DMA_DNSEQCTLx is used when reset is not enabled, and DMA_DNSEQCTLBx is used when reset is enabled.

14.5.20 Target Device Discontinuous Address Transfer Control Register B (DMA_DNSEQCTLBx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	DNSCNTB[11:0]		DNSDIST[19:16]	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	DNSDIST[15:0]			
<hr/>																			
<hr/>																			
Bit	Marking			Place name				Function								Read and write			
b31-b20	DNSCNTB[11:0]			The amount of data that the target address jumps to				Set the amount of data transferred before the target address jumps. After the target device transmits DNSCNTB data blocks, the target address jumps according to the address interval specified by DNSDIST. The data block size is determined by DMA_DTCCTLx.BLKSIZE and DMA_CHxCTL.HSIZE.								R/W			
b19-b0	DNSDIST[19:0]			Target Discontinuous Area Address Spacing				When discontinuous address transmission, set the distance between two discontinuous areas of the target device. The direction of the jump jumps forward or backward according to the value of the channel control register DMA_CHxCTL.DINC. reference imageFigure 14-4. When DMA_CHxCTL.DINC is set to fixed address, discontinuous address transfer will be invalid. The address spacing will be calculated based on the number of bits set by the data width (DMA_CHxCTL.HSIZE) and the value of DNSDIST. Address distance=DNSDIST×(HSIZE(bit)/8) For example, when DNSDIST is set to 10 and HSIZE is word (32bit), the address spacing is $10 \times 4 = 40$; if HSIZE is halfword (16bit), the spacing is $10 \times 2 = 20$; if HSIZE is byte (8bit), the spacing is $10 \times 1 = 10$. The target address of the next transmission = the first address of the current target discontinuous area ± address distance								R/W			

The discontinuous transmission of the target device needs to configure the DNSEQEN bit of the DMA_CHxCTL register to be valid, and configure the DINC bit of the DMA_CHxCTL register to make the address update mode auto-increment or auto-decrement.

The two registers DMA_DNSEQCTLx, DMA_DNSEQCTLBx share the same address and are used to define discontinuous transmission. Which one to use depends on whether the channel has reset enabled or not. DMA_DNSEQCTLx is used when reset is not enabled, and DMA_DNSEQCTLBx is used when reset is enabled.

14.5.21 Chain Pointer Register (DMA_LLPx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LLP[31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LLP[15:2]															

Bit	Marking	Place name	Function	Read and write
b31-b2	LLP[31:2]	Chain pointer	When the chain transmission is valid, set the address of the next transmission descriptor to word alignment, that is, LLP [1: 0] is fixed to 0	R/W
b1-b0	Reserved	-	Read as "0", write as "0"	R/W

14.5.22 Channel Control Register (DMA _ CHxCTL) ($x = 0 \sim 3$)

Reset value: 0x0000_1000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	IE	LLPRUN	LLPEN	HSIZE[1:0]	DNSEQEN	SNSEQEN	DRPTEN	SRPTEN	DINC[1:0]	SINC[1:0]			

Bit	Marking	Place name	Function	Read and write
b31-b13	Reserved	-	Read as "0", write as "0"	R/W
b12	IE	Interrupt enable bit	Configure whether the channel generates an interrupt. 0: This channel does not generate an interrupt 1: The channel generates an interrupt	R/W
b11	LLPRUN	Linkage mode selection	When the chain transfer is active, set whether to start the transfer corresponding to the new descriptor immediately after loading the new descriptor specified by the chain pointer when the current transfer is complete 0: Do not transmit immediately, wait for the next transmission request to start 1: Start transmission immediately after the new descriptor is loaded	R/W
b10	LLPEN	Chain transmission enable	0: Invalid chain transmission 1: Linkage transmission is effective	R/W
b9-b8	HSIZE[1:0]	Width of transmitted data	00: 8bit 01: 16bit 10, 11: 32bit	R/W
b7	DNSEQEN	Destination Address Discontinuous Transfer Enable	0: Do not allow discontinuous address transmission 1: Allow discontinuous address transmission	R/W
b6	SNSEQEN	Source address discontinuous transfer enable	0: Do not allow discontinuous address transmission 1: Allow discontinuous address transmission	R/W
b5	DRPTEN	Target repeat transfer function enable bit	Set whether to allow the target address to reload the initial value 0: no reload 1: Overload	R/W
b4	SRPTEN	Source repeat transfer function enable bit	Set whether to allow the source address to reload the initial value 0: no reload 1: Overload	R/W
b3-b2	DINC[1:0]	Update mode of destination address	00: Fixed 01: Incremental 10, 11: decrement	R/W
b1-b0	SINC[1:0]	Source address update	00: Fixed 01: Incremental 10, 11: decrement	R/W

14.5.23 Channel Monitor Registers (DMA_MONSAR_x, DMA_MONDAR_x, DMA_MONDTCTL_x, DMA_MONRPT_x, DMA_MONSEQCTL_x, DMA_MONDSEQCTL_x) (x=0~3)

These monitoring registers correspond to the corresponding channel configuration registers, and the register bit configurations are consistent, but all are read-only registers.

The channel configuration register remains unchanged before and after the DMA transfer, and the channel monitoring register will be updated after each transfer corresponding to a request is completed by the DMA, that is, after each data block transfer is completed. The update content and method are as follows:

- DMA_MONSAR_x.SAR[31:0], DMA_MONDAR_x.DAR[31:0]: Update the address of the next transmission according to the fixed/increment/decrement/reload/discontinuous jump mode set by the channel configuration register.
- DMA_MONDTCTL_x.CNT[15:0]: Subtract 1, if it is already 0, keep it as 0.
- DMA_MONRPT_x.SRPT[9:0], DRPT[9:0]: When the channel reset is invalid, the block size is subtracted, and when it is reduced to 0 or the value is less than the block size, the DMA_RPT_x setting value is reloaded. When reset is active, decrement by 1, and reload DMA_RPTB_x setting value when decremented to 0.
- DMA_MONSEQCTL_x.SNSCNT[11:0], DMA_MONDNSEQCTL_x.DNSCNT[11:0]: When the channel reset is invalid, the block size is subtracted, and when it is reduced to 0 or the value is less than the block size value, the original setting value of DMA_SNSEQCTL_x/DMA_DNSEQCTL_x. When reset is valid, decrement 1, and reload DMA_SNSEQCTLB_x/DMA_DNSEQCTLB_x setting value when decremented to 0.

Monitor register bits other than the above remain the same as the configuration register.

14.6 Precautions for Use

- 1 DMA registers only support 32bit read and write, and 8/16bit read and write operations are invalid.
- 2 Writing to channel configuration registers during DMA transfers has no effect. Channel configuration registers include: DMA_SARx, DMA_DARx, DMA_DTCTLx, DMA_RPTx, DMA_RPTBx, DMA_SNSEQCTLx, DMA_SNSEQCTLBx, DMA_DNSEQCTLx, DMA_DNSEQCTLBx, DMA_LLPx, DMA_CHxCTL (x=0~3). Please make sure that the above registers are written when the DMA is in an idle state, or read out after writing the registers to confirm whether the writing is successful.

15 Voltage Comparator (CMP)

15.1 Introduction

The voltage comparator (Comparator, hereinafter referred to as CMP) is a peripheral module that compares two analog voltages (positive terminal voltage INP and negative terminal voltage INM) and outputs the comparison result. It has 3 independent comparison channels. There are 4 input sources for the positive terminal voltage and negative terminal voltage of each comparison channel. When using it, you can select one for single comparison, or you can select multiple positive terminal voltages to scan and compare with the same negative terminal voltage. The comparison result can be read through registers, can also be output to external pins, and can also generate interrupts and events.

This series is also equipped with two 8-bit digital-to-analog converters (hereinafter referred to as 8bitDAC), whose analog output can be used as a negative terminal voltage input source for CMP.

Table 15-1 CMP Detailed Specifications

Item	Specifications
compare channels	3 comparison channels: CMP1~3
Positive terminal voltage input source	12 input sources: 10 external analog inputs, 2 internal PGA outputs
Negative terminal voltage input source	7 input sources: 4 external analog inputs, 1 internal Vref, 2 internal D/A outputs
Comparing results	Can be read from the register or output to the external pin VCOUT with selectable polarity
Operating mode	Single compare mode and scan compare mode
Digital filtering	7 sampling frequencies are optional, and can be turned off when no filtering is required
Interrupt and events	Detects valid edges of compare results and generates interrupts and peripheral trigger events
Low power control	Module stop function reduces system power consumption

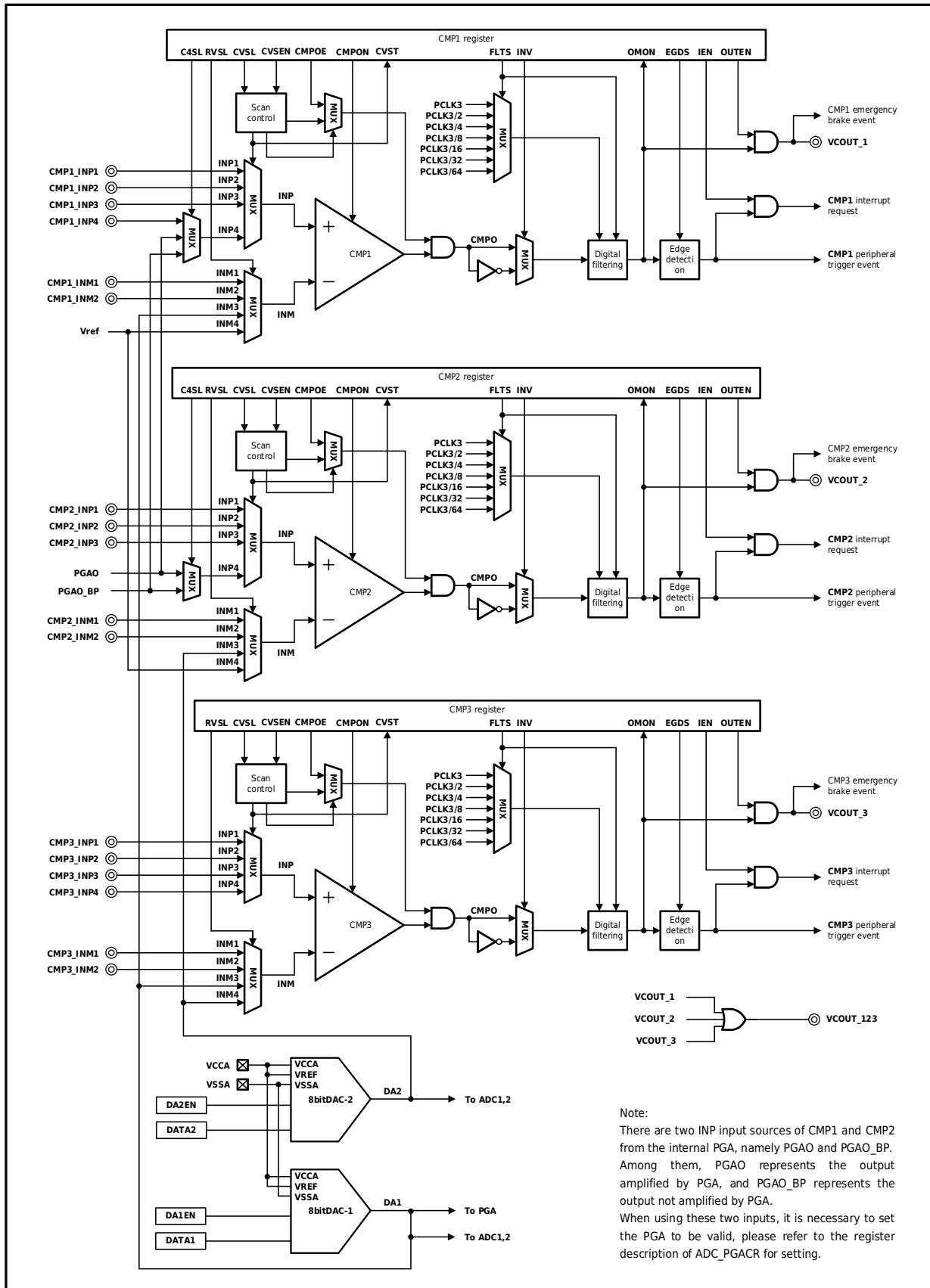


Figure 15-1 CMP, 8bitDAC function connection diagram

15.2 Functional Description

15.2.1 Voltage Comparison

When CMP works in a single comparison mode, select any comparison channel and select the positive terminal voltage INP and the negative terminal INM for it, and then the comparison of these two voltages can be realized. The working diagram of CMP is shown in Figure 15-2. If positive polarity output is set, CMP outputs high level when INP is higher than INM, and CMP outputs low level when INP is lower than INM. Conversely, if negative polarity output is set, CMP outputs low level when INP is higher than INM, and outputs high level when INP is lower than INM. Finally, the comparison result can be read from the respective comparison result monitoring register CMPMON.OMON bit of each comparison channel, and can also be output to the external pin VCOUT.

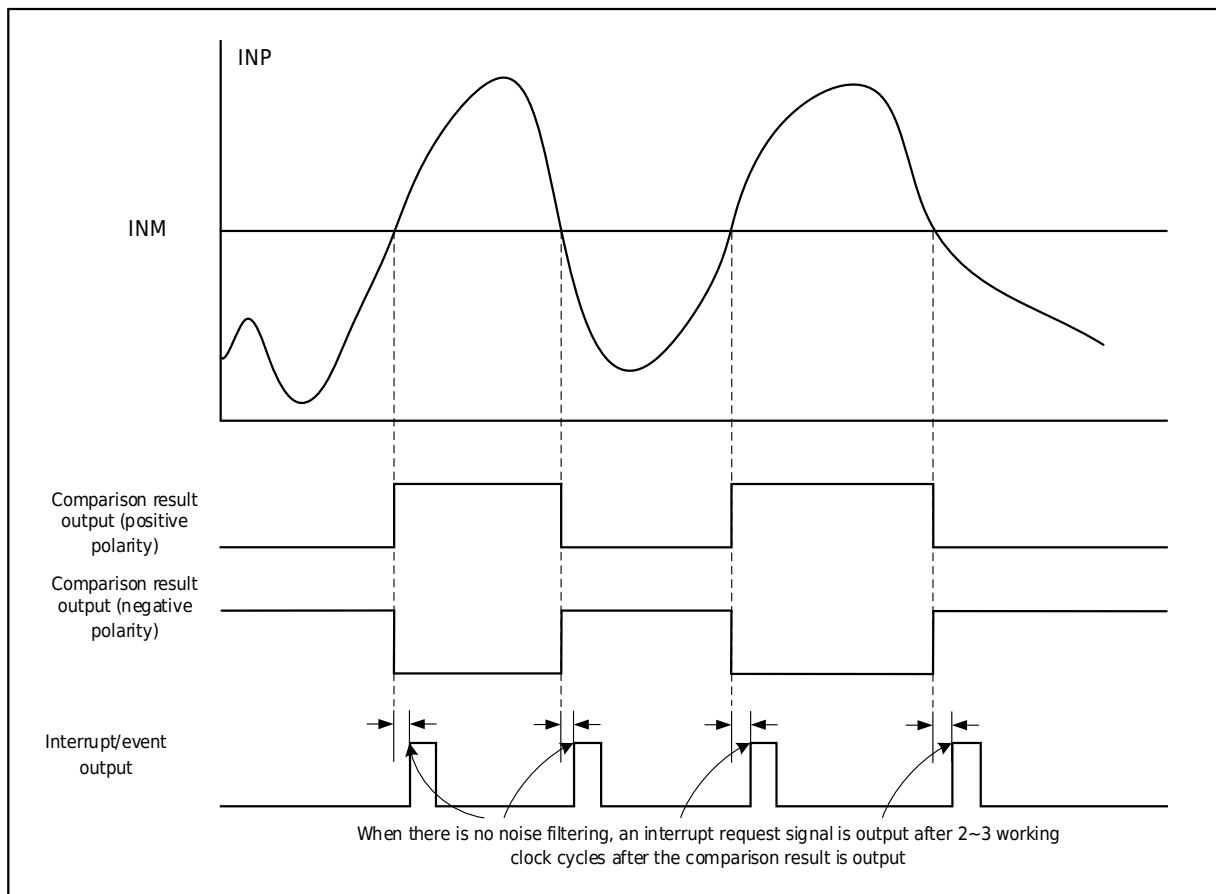


Figure 15-2 Schematic diagram of CMP work

15.2.2 Digital Filtering

Each comparison channel has a digital filter circuit, which can perform noise filter processing on the comparison results. The filter circuit compares the output CMPO of the comparator after three samples. If the results of the three samples are consistent, the sampled result is output as the comparison result. If the results of the three samples are inconsistent, the output of the comparison result remains unchanged. There are 7 options for sampling frequency, please refer to the function description of the register CMP_CTRL.FLTS[2:0] for details. The filter function can also be closed, and the comparison result output is exactly the same as the comparator output CMPO at this time.

15.2.3 Interrupt and Events

When the output of the comparison result changes, an interrupt request can be generated and an event triggered by other peripherals can be triggered. The edge that generates the interrupt and event can choose the rising edge, falling edge or both edges of the comparison result output. Among them, the interrupt request of comparison channel 1 (CMP1) can also wake up/stop the low power consumption mode, but the interrupt edge must be selected as a rising edge and the digital filter function must be turned off.

When using compare interrupts and events for the first time, please refer to the following procedure to set.

- 1) Set CMP_VLTSEL to select INP and INM input sources.
- 2) Set CMP_CTRL.INV to select the comparator output polarity.
- 3) Set CMP_CTRL.EDGSL[1:0] to select the interrupt edge.
- 4) Set CMP_CTRL.CMPON, enable the comparator and wait for 300ns stabilization time.
- 5) Set CMP_CTRL.CMPOE to enable comparator output.
- 6) If interrupts are used, set CMP_CTRL.IEN to enable interrupts.
If using events, select the trigger event of the enabled peripheral as the CMP compare event.

15.2.4 Scan Compare Mode

When setting register CMP_VLTSEL.CVSL[3:0] to select two or more INPs as comparison objects, CMP enters scan comparison mode, and then sets register CMP_CTRL.CVSEN to "1" to start scanning. When CMP works in the scan comparison mode, the INM input remains unchanged, and the INP input will automatically switch according to the set scan cycle. If the valid edge selected by the register CMP_CTRL.EDGSL[1:0] is detected during the scan, the register CMP_CTRL.CVSEN bit will be automatically cleared and the scan will be suspended. At this time, the register CMP_OUTMON.CVST[3:0] bits can be read to determine the current INP input. When the register CMP_CTRL.CVSEN bit is written to "1" again, the scanning operation continues.

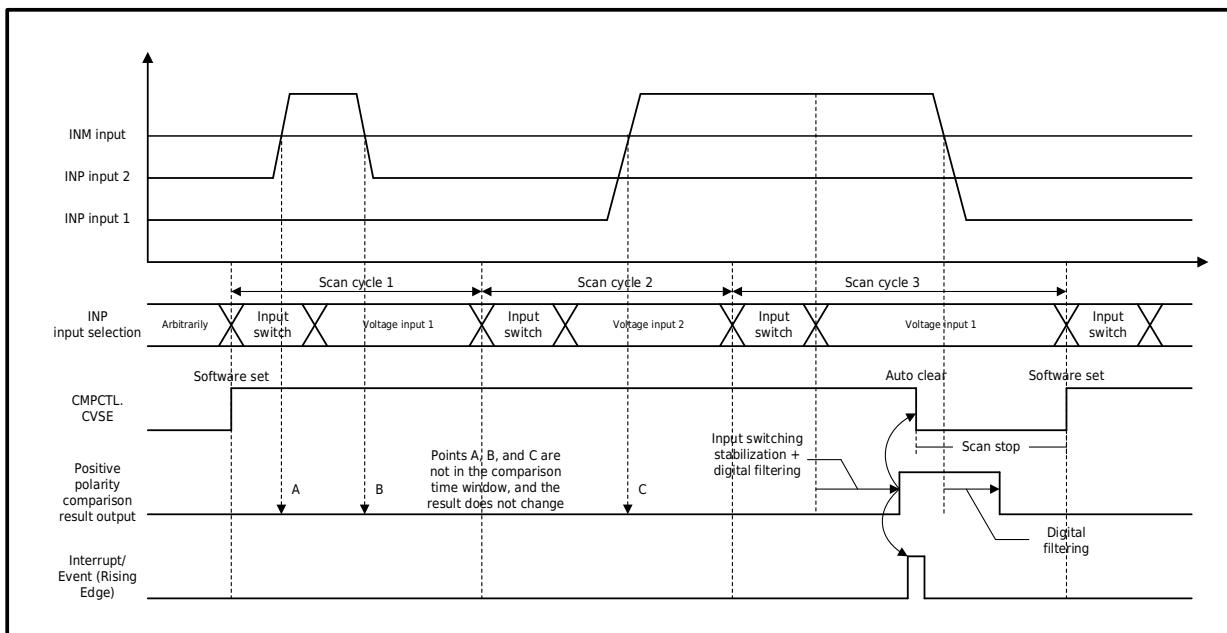


Figure 15-3 Schematic Diagram of Scanning Mode Action

The figure above is a schematic diagram of scanning two INP inputs. In the first two scanning cycles, since the changes of INP input 1 and INP input 2 did not occur within the respective comparison time windows, the output of the comparison result remains unchanged. Entering scan cycle 3, INP input 1 is higher than INM to cause the comparator to reverse, and the output is passed to the comparison result output after digital filtering. At this time, the scan action stops automatically and interrupts and events are generated. Next, CMP keeps working in the state where INP input 1 is compared with INM until the software restarts the scan mode.

Please refer to the following process to set up using scan comparison mode.

- 1) Set CMP_VLTSEL[3:0] to select INP and INM input sources (INP selects two or more).
- 2) Set CMP_CVSSTB to set the voltage switching stabilization time.
- 3) Set CMP_CVSPRD to set the voltage scan period.
- 4) Set CMP_CTRL.INV to select the comparator output polarity.
- 5) Set CMP_CTRL.FLTSIL[2:0] to select the digital filter sampling frequency.

- 6) Set CMP_CTRL.EDGSL[1:0] to select the interrupt edge.
- 7) Set CMP_CTRL. IEN to enable interrupts.
- 8) Set CMP_CTRL.CMPON, enable the comparator and wait for 300ns stabilization time.
- 9) Set CMP_CTRL.CMPOE to enable comparator output.
- 10) Set CMP_CTRL.CVSEN to start scanning.

The voltage switching stabilization time and the voltage scanning cycle correspond to the number of PCLK3 cycles. In order to ensure the correct action, the set value must meet the condition of "scan period > switching stabilization time + filter sampling period × 4 + 5".

15.2.5 8bit-DAC Setting

Two 8bit-DACs provide two internal voltages DA1 and DA2 for CMP. The output voltage at work is:

$$\text{Output voltage} = \text{VCCA} \times \text{conversion data} / 255$$

The conversion data of DA1 and DA2 are respectively set by CMP_DADDR1 and CMP_DADDR2.

When using, please refer to the following process for setting:

- 1) Set CMP_DADDR1, CMP_DADDR2, and set conversion data.
- 2) Set CMP_DACR to start DA1 and DA2.
- 3) Wait for the DA conversion stabilization time.
- 4) Set the comparator.

15.3 Precautions

15.3.1 Module Stop Function

CMP has module stop function, by setting module stop register can turn off the digital part of the module. CMP is initially stopped and is not accessible until the module is set to work. For details, please refer to the chapter on low power consumption.

15.3.2 Act of Stopping a Module.

When the CMP enters the stop state from the working state, the analog part of the CMP continues to work, and the power consumption is equal to the working state. To reduce power consumption, first set CMP_CTRL.CMPON to "0" to stop the CMP analog part.

Similarly, when the 8bit-DAC enters the stop state in the working state, the analog part of the DA continues to work, and the power consumption is equal to the working state. To reduce power consumption, please first set DA1EN and DA2EN of DACR to "0" to stop the work of the 8bit-DAC analog part.

15.3.3 Stop Action in Low Power Mode

When the HC32F46xx enters the stop low power consumption mode in the CMP working state, the CMP will continue to work, and the power consumption is equal to the level before entering the stop low power consumption mode. If you need to further reduce power consumption in stop low power mode, please clear CMP_CTRL.CMPON to "0" before entering stop low power mode to stop CMP from working.

Similarly, when the HC32F46xx enters the stop low power consumption mode during 8bit-DAC conversion, the D/A output will be maintained, and the power consumption is equal to the level before entering the stop low power consumption mode. If you need to further reduce power consumption in the stop low power mode, please set the DA1EN and DA2EN bits of DACR to "0" before entering the stop low power mode to stop the 8bit-DAC from working.

15.3.4 Actions in Power-down Low-power Mode

When HC32F46xx enters power-down low-power consumption mode, both CMP and 8bit-DAC will stop working. The CMP comparison result outputs Low, and the 8bit-DAC output becomes a high-impedance state.

15.4 Register Description

CMP1 base address: 0x4004_A000

CMP2 base address: 0x4004_A010

CMP3 base address: 0x4004_A020

Table 15-2 List of CMP Registers

Register name	Symbol	Offset address	Bit width	Reset value
CMP control register	CMP_CTRL	0x000	16	0x0000
CMP Voltage Select Register	CMP_VLTSEL	0x002	16	0x0000
CMP Result Monitor Register	CMP_OUTMON	0x004	16	0x0000
CMP Settling Time Register	CMP_CVSSTB	0x006	16	0x0005
CMP Scan Period Register	CMP_CVSPRD	0x008	16	0x000F
8bitDAC data register 1 for CMP	CMP_DADR1	0x100	16	0x0000
8bitDAC data register 2 for CMP	CMP_DADR2	0x102	16	0x0000
8bitDAC control register for CMP	CMP_DACR	0x108	16	0x0000
CMP internal reference voltage AD conversion register	CMP_RVADC	0x10C	16	0x0000

Note:

- CMP_DADR1, CMP_DADR2, and CMP_DACR are shared registers that only exist in CMP1.

15.4.1 CMP Control Register (CMP_CTRL)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CMPON	CMPOE	INV	OUTEN	-	-	-	CVSEN	IEN	EDGSL[1:0]	-	-	-	-	FLTSL[2:0]	

Bit	Marking	Place name	Function	Read and write
b15	CMPON	Comparator Work Permit	0: Comparator stop (comparator output Low) (reset value) 1: Comparator works Note: - After the comparator work permit CMPON is set to "1", it is necessary to wait for about 300ns of work stabilization time before subsequent operations can be performed.	R/W
b14	CMPOE	Comparator Output Permission	0: Comparator output disabled (comparator output Low) (reset value) 1: Comparator output enable	R/W
b13	INV	Comparator output polarity selection	0: Comparator positive polarity output (reset value) 1: Comparator negative polarity output (comparator output is inverted) Note: - Override INV when comparator output is disabled (i.e. CMPOE bit is "0"). Changing the INV 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
b12	OUTEN	Comparison result output permission	0: Disable comparison result output (comparison result output Low) (reset value) 1: Allow comparison result output	R/W
b11~b9	Reserved	-	Read as "0", write as "0"	R
b8	CVSEN	Compare Voltage Scan Permission	0: Comparison voltage scan stops (reset value) 1: compare voltage scan starts Note: - When the valid edge of the comparison result is detected, CVSEN will be automatically cleared, and if you want to continue scanning, please write "1" again.	R/W
b7	IEN	Compare Interrupt Licenses	0: Disable compare interrupt (reset value) 1: Enable compare interrupt	R/W
b6~b5	EDGSL [1:0]	Comparison result active edge selection	0 0: Does not detect the edge of the comparison result (reset value) 0 1: detect the rising edge of the comparison result 1 0: Detect the falling edge of the comparison result 1 1: detect the rising edge and falling edge of the comparison result	R/W
b4~b3	Reserved	-	Read as "0", write as "0"	R
b2~b0	FLTSL [2:0]	Filter sample selection	0 0 0: Do not use noise filtering (reset value) 0 0 1: Sampling using PCLK3 0 1 0: Divide-by-2 sampling using PCLK3 0 1 1: divide by 4 sampling using PCLK3 1 0 0: divide-by-8 sampling using PCLK3 1 0 1: divide-by-16 sampling using PCLK3 1 1 0: divide-by-32 sampling using PCLK3 1 1 1: Divide-by-64 sampling using PCLK3 Note: - Please rewrite FLTSL[2:0] when the comparator output is disabled (that is, the CMPOE bit is "0"). When FLTSL [2: 0] switches from '000b' to a different value, use the filtered output after 4 samples as a interrupt request or a peripheral trigger event. - Changing the FLTSL [2: 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

15.4.2 CMP Voltage Select Register (CMP_VLTSEL)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	C4SL[2:0]		CVSL[3:0]		-	-	-	-	-	-		RVSL[3:0]			
Bit	Marking	Place name	Function										Read and write		
b15	Reserved	-	Read as "0", write as "0"										R		
b14~b12	C4SL[2:0]	INP4 input selection	0 0 0: No input (reset value) 0 0 1: Select internal PGAO output (only valid for CMP1 and CMP2) 0 1 0: Select internal PGAO_BP output (only valid for CMP1 and CMP2) 1 0 0: Select external CMP1_INP4 input (only valid for CMP1) Note: 1. The setting is invalid for CMP3. 2. Setting other values is prohibited. 3. When rewriting C4SL[2:0], please refer to the following steps: (1) Clear the CMP_CTRL.CMPOE bit to "0". (2) Write a new value to the C4SL[2:0] bit (note that only 1 bit is "1"). (3) Wait for the input to stabilize. (4) CMP_CTRL.CMPOE bit "1". (5) Clear the corresponding interrupt flag bit.										R/W		
b11~b8	CVSL[3:0]	INP input selection	0 0 0 0: No input (reset value) 0 0 0 1: selectINP1 0 0 1 0: Select INP2 0 0 1 1: Select INP1, INP2 (scanning mode) 0 1 0 0: select INP3 0 1 0 1: Select INP1, INP3 (scanning mode) 0 1 1 0: Select INP2, INP3 (scanning mode) 0 1 1 1: Select INP1, INP2, INP3 (scan mode) 1 0 0 0: Select INP4 1 0 0 1: Select INP1, INP4 (scanning mode) 1 0 1 0: Select INP2, INP4 (scanning mode) 1 0 1 1: Select INP1, INP2, INP4 (scan mode) 1 1 0 0: Select INP3, INP4 (scanning mode) 1 1 0 1: Select INP1, INP3, INP4 (scan mode) 1 1 1 0: Select INP2, INP3, INP4 (scan mode) 1 1 1 1: Select INP1, INP 2, INP 3, INP 4 (scan mode)										R/W		
b7~b4	Reserved	-	Read as "0", write as "0"										R		
b3~b0	RVSL[3:0]	INM input selection	0 0 0 0: No input (reset value) 0 0 0 1: Select INM1 0 0 1 0: Select INM2 0 1 0 0: Select INM3 1 0 0 0: Select INM4 Note: 1. See "Figure 15-1" for the specific input source. 2. When selecting INM4, please set C4SL[2:0] to select the input source. 3. When rewriting RVSL[3:0], please refer to the following steps: (1) Clear the CMP_CTRL.CMPOE bit to "0". (2) RVSL[3:0] bits write new values. (3) Wait for the input to stabilize. (4) CMP_CTRL.CMPOE bit "1". (5) Clear the corresponding interrupt flag bit.										R/W		

15.4.3 CMP Result Monitor Register (CMP_OUTMON)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
															OMON	
Bit	Marking	Place name		Function											Read and write	
b15~b12	Reserved	CVST[3:0]											Read as "0", write as "0"			R
b11~b8	CVST[3:0]	INP input status bit		0 0 0 0: The current INP input selects INP1 0 0 0 1: The current INP input selects INP1 0 0 1 0: The current INP input selects INP2 0 1 0 0: The current INP input selects INP3 1 0 0 0: The current INP input selects INP4 Note: 1. This register is read-only, write all "0" when writing. 2. See "Figure 15-1" for the specific input source.											R	
b7~b1	Reserved	-		Read as "0", write as "0"											R	
b0	OMON	Comparison result monitor bit		0: The comparison result output is "0" (reset value) 1: The output of the comparison result is "1" Note: 1. This register is read-only, write all "0" when writing. 2. When setting the comparator to work under the condition of CMP_CTRL.FLTS[2:0]=000b (noise filter circuit is not used), please use the method of reading the OMON bit twice to confirm the comparison status.											R	

15.4.4 CMP Stabilization Time Register (CMP_CVSSTB)

Reset value: 0x0005

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
															STB[3:0]	
Bit	Marking	Place name		Function											Read and write	
b15~b4	Reserved	-		Read as "0", write as "0"											R	
b3~b0	STB[3:0]	Input Switching Settling Time		Set the output stabilization time of CMP when switching INP input in scan compare mode. The reset value is 0x5, and the set value can be any value between 0x0 ~ 0xF. CMP output stabilization time = PCLK3 period × STB set value For example, PCLK3 is 40MHz, STB is set to 0x5, then Scan period = 25(nS) × 0x5 = 125(nS) Note: 1. Please refer to "Comparator Characteristics - Input Channel Switching Settling Time" in the Electrical Characteristics section of the Data Manual. 2. To rewrite STB, please stop scanning and switch to single comparison mode.											R/W	

15.4.5 CMP Compare Voltage Scan Period Register (CMP_CVSPRD)

Reset value: 0x000F

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PRD[7:0]
Bit	Marking	Place name												Function	Read and write
b15~b8	Reserved	-												Read at "0", write at "0"	R
b7~b0	PRD[7:0]	Compare Voltage Sweep Period												Set the time interval for switching the INP input in the scan comparison mode, that is, the scan cycle. The reset value is 0x0F, and the set value can be any value between 0x0F ~ 0xFF. Scan period = PCLK3 period × PRD set value For example, PCLK3 is 40MHz, PRD is set to 0x50, then Scan period = $25(\text{nS}) \times 0x50 = 2(\mu\text{s})$	R/W
		Note: 1. In order to ensure correct operation, please confirm when setting PRD Scan period > switching settling time + filter sampling period $\times 4 + 5$ 2. To rewrite the PRD, please stop scanning and switch to single comparison mode.													

15.4.6 8bit-DAC Control Register for CMP (CMP_DACR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	DA2 EN	DA1 EN
Bit	Marking	Place name												Function	Read and write
b15~b2	Reserved	-												Read "0" when reading, write "0" when writing	R
b1	DA2EN	DA2 start bit												0: DA2 stopped (reset value) 1: DA2 counting work	R/W
b0	DA1EN	DA1 enable bit												0: DA1 stop (reset value) 1: DA1 works	R/W

15.4.7 8bit-DAC Data Registers for CMP (CMP_DADDR1, CMP_DADDR2)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	DATA[7:0]
Bit	Marking	Place name												Function	Read and write
b15~b8	Reserved	-												Read "0" when reading, write "0" when writing	R
b7~b0	DATA[7:0]	DA conversion data												Any value between 8'h00 (reset value) ~ 8'hFF	R/W

15.4.8 CMP Internal Reference Voltage AD Conversion Register (CMP_RVADC)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
								-	-	-	VREF SW	-	-	DA2 SW	DA1 SW
Bit	Marking	Place name	Function												Read and write
b15~b8	WPRT[7:0]	write protection bit	8'h55: VREFSW, DA2SW, DA1SW bits write valid others: VREFSW, DA2SW, DA1SW bits write invalid Note: WPRT will revert to "8'h00" when writing any value other than "8'h55".												R/W
b7~b5	Reserved	-	Read "0" when reading, write "0" when writing												R
b4	VREFSW	Internal Vref AD conversion switch	0: The internal Vref AD conversion path is disconnected 1: Internal Vref AD conversion path connection Note: VREFSW, DA2SW, DA1SW can only be "1".												R/W
b3~b2	Reserved	-	Read "0" when reading, write "0" when writing												R
b1	DA2SW	DA2 AD conversion switch	0: DA2 AD conversion channel is disconnected 1: DA2 AD conversion channel connection Note: VREFSW, DA2SW, DA1SW can only be "1".												R/W
b0	DA1SW	DA1 AD conversion switch	0: DA1 AD conversion channel is disconnected 1: DA1 AD conversion channel connection Note: VREFSW, DA2SW, DA1SW can only be "1".												R/W

16 Analog-to-Digital Conversion Module (ADC)

16.1 Introduction

The ADC of this product is an analog-to-digital converter using successive approximation, with a maximum resolution of 12 bits, and supports up to 17 analog input channels, which can convert analog signals from external pins and inside the chip. These analog input channels can be randomly combined into a sequence, which can be either a single scan conversion or a continuous scan conversion. It supports multiple consecutive conversions on any specified channel and averages the conversion results. The ADC module also carries the analog watchdog function, monitors the conversion result of any specified channel, and detects whether it exceeds the threshold set by the user.

Main Features of ADC:

- High performance
 - Configurable 12-bit, 10-bit and 8-bit resolution
 - The frequency ratio of digital interface clock PCLK4 and A/D conversion clock ADCLK can be selected:
PCLK4: ADCLK=1: 1, 2: 1, 4: 1, 8: 1, 1: 2, 1: 4
ADCLK can select the PLL clock asynchronous with the system clock HCLK, at this time
PCLK4: ADCLK=1: 1
 - Sampling rate: 2.5MSPS (PCLK4=ADCLK=60MHz, 12 bits, sampling 11 cycles)
 - The sampling time of each channel is independently programmed
 - Channel-independent data register
 - Data register configurable data alignment
 - Consecutive multiple conversion average function
 - Simulation watchdog, monitoring conversion results
 - The ADC module can be set to stop when not in use
- Analog input channel
 - Maximum 16 external analog input channels
 - 1 internal reference voltage/8bitDAC output detection channel
- Conversion start condition
 - Software Setup Conversion Started
 - External event triggers conversion start
 - External Pin Triggering Start
- Conversion mode
 - 2 scan sequences A and B, optionally specifying a single or multiple channels
 - Sequence A single scan
 - Sequence A continuous scanning

- Double sequence scanning, sequence A and B independently select trigger source, sequence B has higher priority than A
- Synchronous mode (for devices with two or three ADCs)
- Interrupt and Event Signal Output
 - Sequence A Scan End Interrupt and Event ADC_EOCA
 - Sequence B Scan End Interrupt and Event ADC_EOCB
 - Analog watchdog channel compare interrupt and event ADC_CHCMP, sequence compare interrupt and event ADC_SEQCMP
 - All four of these event outputs can start the DMA

16.2ADC System Block Diagram

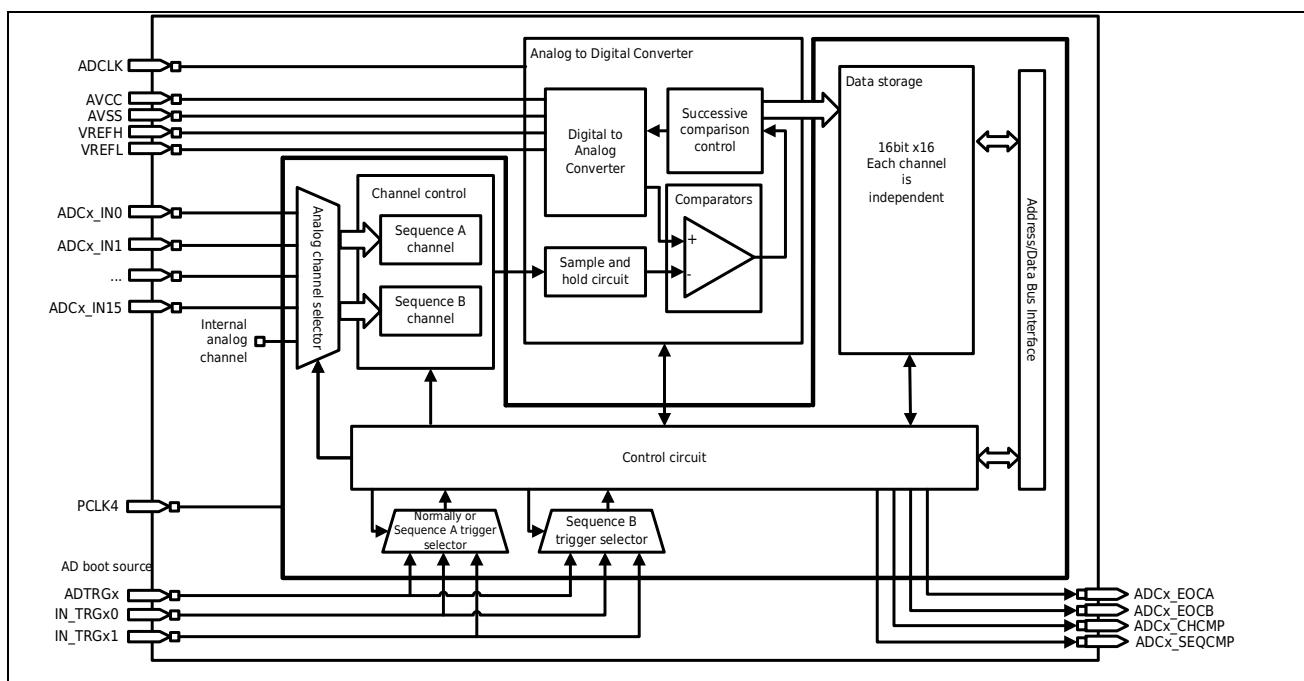


Figure 16-1 ADC Block Diagram

This chip is equipped with 2 ADC module units, and the configuration of each unit is different. For details, refer to the following table:

Table 16-1 Specifications of each ADC unit

Item		Unit 1 (ADC1)	Unit 2 (ADC2)
Power supply		AVCC	
		AVSS/VREFL	
The reference voltage		VREFH *1	
Analog channels *2	CH0	ADC1_IN0	ADC12_IN4
	CH1	ADC1_IN1	ADC12_IN5
	CH2	ADC1_IN2	ADC12_IN6
	CH3	ADC1_IN3	ADC12_IN7
	CH4	ADC12_IN4	ADC12_IN8
	CH5	ADC12_IN5	ADC12_IN9
	CH6	ADC12_IN6	ADC12_IN10
	CH7	ADC12_IN7	ADC12_IN11
	CH8	ADC12_IN8	Internal analog channel (reference voltage/8bitDAC output)
	CH9	ADC12_IN9	-
	CH10	ADC12_IN10	-
	CH11	ADC12_IN11	-
	CH12	ADC1_IN12	-
	CH13	ADC1_IN13	-
	CH14	ADC1_IN14	-
	CH15	ADC1_IN15	-
	CH16	Internal analog channel (reference voltage/8bitDAC output)	
PGA		Any 1 channel of ADC1_IN0~3, ADC12_IN4~7, 8bitDAC_1 output	
Hardware trigger source	External pin	ADTRG1	ADTRG2
	On-chip peripheral s	IN_TRG10	IN_TRG20
		IN_TRG11	IN_TRG21

Note:

- VREFH is available under the LQFP100 package, but not available under other packages, and AVCC is used instead of VREFH.
- The ADC analog channels CH0~CH15 and the actual input ADCx_INy can be freely mapped by setting registers. This table shows the default mapping relationship after reset.

16.3 Functional Description

16.3.1 ADC Clock

The ADC module needs to use two clocks: the analog circuit clock ADCLK, and the digital interface clock PCLK4.

Both clocks come from frequency dividers in the clock controller. ADCLK is equivalent to PCLK2 and is in a synchronous relationship with PCLK4. The frequency ratio relationship between PCLK4 and ADCLK is 1:1, 2:1, 4:1, 8:1, 1:2, 1:4.

ADCLK can choose a PLL clock source that is asynchronous to the system clock HCLK. At this time, PCLK4 is the same as ADCLK, which is a synchronous frequency relationship. The maximum frequency of ADCLK is 60MHz.

16.3.2 Channel Selection

The ADC module has multiple channels and can be configured as two sequences: sequence A, sequence B for conversion. Sequences A and B are equipped with independent channel selection registers ADC_CHSELRA, ADC_CHSELRB. Register each represents a channel, such as bit0 bit write 1 to convert CH0 and write 0 to not convert CH0. Two sequences can be converted independently by selecting any one or more channels. For example: ADC_CHSELRA is set to 0x0055, ADC_CHSELRB is set to 0x0002, when the trigger condition of sequence A occurs, the four channels CH0, CH2, CH4 and CH6 will be converted in sequence. When the trigger condition for sequence B occurs, CH1 is converted.

For the internal analog channel, there are 3 optional analog quantities: 8bitDAC_1 output, 8bitDAC_2 output, and internal reference voltage (about 1.1V). As shown below. When selecting the internal analog channel as the ADC conversion object, you need to set the internal channel selection register first, and then set the ADC channel selection register ADC_CHSELRA/ADC_CHSELRB to select the internal channel before ADC conversion can be started ed.

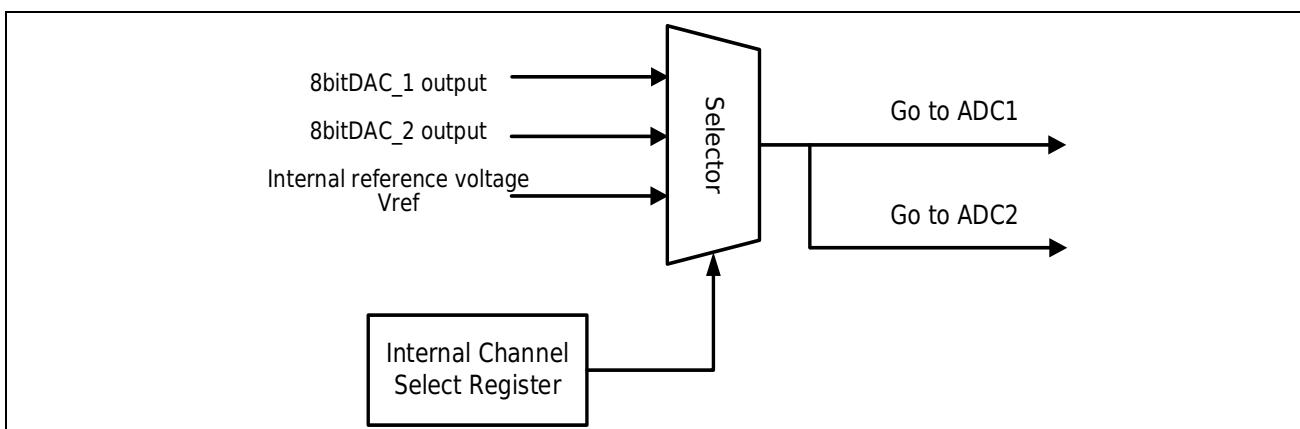


Figure 16-2 Internal Analog Channel Selection

For the specific setting method of the internal channel selection register, please refer to the description of the register CMP_RVADC in the chapter [Voltage Comparator (CMP)] and the description of the register PWC_PWCMR in the chapter [Power Control (PWC)].

Table 16-2 Register Setting Method when Switching Internal Channels

Conversion target	ADC_CHSELR	CMP_RVADC	CMP_DADR	CMP_DACR	PWC_PWCMR
8bitDAC_1 -> ADC1	ADC1_CHSELRA1 or ADC1_CHSELRB1 bit0 write 1, select CH16	Write 0x0001 *1	Write CMP_DADR1 to specify 8bitDAC_1 output voltage value	Write 1 to bit0 to allow 8bitDAC_1 output	—
8bitDAC_2 -> ADC1		Write 0x0002 *1	Write CMP_DADR1 to specify 8bitDAC_2 output voltage value	Write 1 to bit1 to allow 8bitDAC_2 output	—
Internal reference voltage vref -> ADC1		Write 0x0010 *1	—	—	Write 0x80 *2
8bitDAC_1 -> ADC2	ADC2_CHSELRA0 or ADC2_CHSELRB0 Write 1 to bit8, select CH8	Write 0x0001 *1	Write CMP_DADR1 to specify 8bitDAC_1 output voltage value	Write 1 to bit0 to allow 8bitDAC_1 output	—
8bitDAC_2 -> ADC2		Write 0x0002 *1	Write CMP_DADR1 to specify 8bitDAC_2 output voltage value	Write 1 to bit1 to allow 8bitDAC_2 output	—
Internal reference voltage vref -> ADC2		Write 0x0010 *1	—	—	Write 0x80 *2

Note:

- CMP_RVADC needs to write 0x5500 first, and then write the target setting value.
- The PWC_PWCMR register needs to turn on the protection bit PWC_FPRC.FPRCB1 first.

NOTE: Do not select the same channel in sequence A and B. For channels that do not exist, please do not set the corresponding registers and keep the status after reset.

CH0 represents channel 0, and the corresponding relationship with the actual analog input channel ADCx_INy can be freely set through the register ADC_CHMUXR. For example: For ADC1, CH00MUX is set to 0x0 to indicate that CH0 is mapped to ADC1_IN0, and set to 0xf to indicate that CH0 is mapped to ADC1_IN15. For ADC2, setting CH00MUX to 0x0 means that CH0 is mapped to ADC12_IN4, and setting it to 0xF means invalid mapping. Similarly, CH1~CH15 can be channel mapped through the corresponding ADC_CHMUXR register.

16.3.3 Trigger Source Selection

Sequence A, sequence B independently selects the trigger source. 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_ITRGSEL0,1, which can select rich event sources inside the chip. In addition, write register ADC_STR generates a

software trigger signal that can only be used when the ADC is in standby state and is only applicable to sequence A.

16.3.4 Sequence A Single Scan Mode

The A/D control register ADC_CR0.MS[1:0] is set to 00b to select the 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.STRT bit writes 1 software trigger, the ADC starts, and all channels selected in the sequence A channel selection register ADC_CHSELRA are sequentially sampled and converted, and the conversion results are stored in the corresponding data register ADC_DR. ADC_STR.STRT remains 1 during the ADC conversion process, and is automatically cleared to 0 when all channel conversions are completed, and the ADC enters the conversion standby state, waiting for the next trigger condition to occur.

When all channel conversion ends, the sequence A conversion end flag bit ADC_ISR.EOCAF is set to 1 and the sequence A conversion end event ADC_EOCA is generated, which can be used to start the DMA. If the ADC_ICR.EOCAIEN is 1, the interrupt request is generated in the state of interrupt enable.

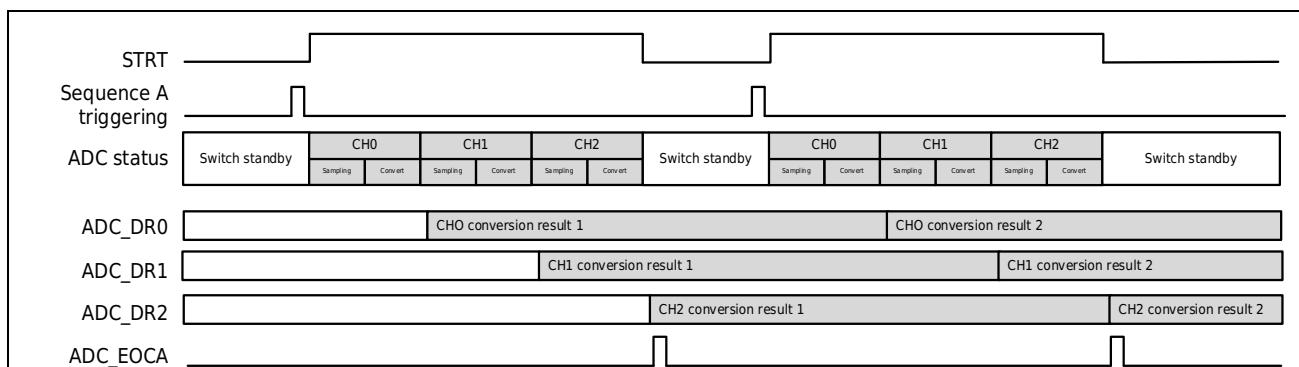


Figure 16-3 Sequence A Single Sweep Mode

Sequence A single scan mode software process:

1. Confirm that ADC_STR.STRT is 0, ADC is in conversion standby state.
2. The A/D control register ADC_CR0.MS[1:0] is set to 00b to select the sequence A single-scan mode.
3. Set the sequence A channel select register ADC_CHSELRA.
4. Set the sampling time register ADC_SSTR.
5. ADC_STR.STRT Write 1 software trigger sequence A or set register ADC_TRGSR select sequence A trigger condition.
6. Query sequence A conversion end flag bit EOCAF.
7. Read each channel data register ADC_DR.
8. Write 0 clears the EOCAF flag bit to prepare for the next conversion.

The CPU query 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 read data using the ADC_EOCA event.

16.3.5 Sequence A Continuous Scanning Mode

The A/D control register ADCR0.MS[1:0] is set to 01b to select sequence A continuous scan mode.

Sequence A continuous scan mode is similar to sequence A single scan mode. The difference lies in that continuous scan mode does not enter conversion standby state after channel conversion, but starts to convert sequence A. STRT bits also do not automatically clear 0.

Write 0 to the STRT bit and read STRT to confirm 0 to determine that the ADC enters the conversion standby state when a continuous scan needs to be stopped.

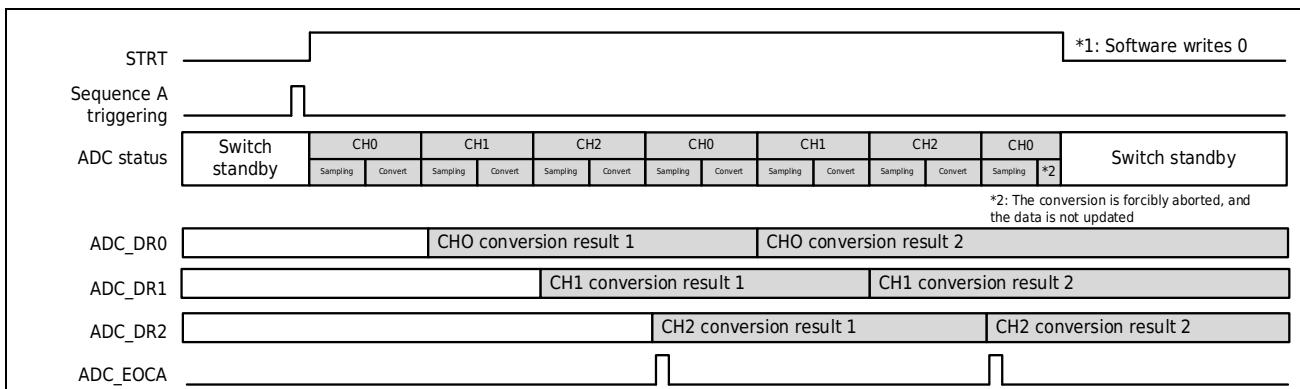


Figure 16-4 Continuous Scanning

Sequence A Continuous Scan Mode software flow:

1. Confirm that ADC_STR.STRT is 0, ADC is in conversion standby state.
2. The A/D control register ADC_CR0.MS[1:0] is set to 01b to select sequence A continuous scan mode.
3. Set the sequence A channel select register ADC_CHSELRA.
4. Set the sampling time register ADC_SSTR.
5. ADC_STR.STRT Write 1 software trigger sequence A or set register ADC_TRGSR select sequence A trigger condition.
6. Query sequence A conversion end flag bit EOCAF.
7. Read each channel data register ADC_DR.
8. Write 0 clears the EOCAF flag bit to prepare for the next conversion.
9. Write 0 to the STRT bit when you do not need to continue the conversion, and read STRT to confirm that 0 to determine if the ADC is in conversion standby.

The above-mentioned query methods in steps 6 to 8 can also be replaced with interrupt methods, which can be used to process the converted data. Or start DMA read data 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 intime processing in query mode.

16.3.6 Double Sequence Scanning Mode

The A/D control register ADC_CR0.MS[1:0] is set to 10b or 11b to select the dual-sequence scan mode, that is, both sequence A and sequence B can start scanning 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.

In sequence A, the trigger source is selected by ADC_TRGSR.TRGSEL_A[2:0], and the converted channel is selected by ADC_CHSELRA. In sequence B, the trigger source is selected by ADC_TRGSR.TRGSEL_B[2:0], and the converted channel is selected by ADC_CHSELRB.

The sequence A conversion end flag bit ADC_ISR.EOCAF is set to 1 and the sequence A conversion end event ADC_EOCA is generated. If the sequence A conversion end flag bit ADC_ISR.EOCAIEN is set to 1, the sequence A conversion end interrupt request is generated. When the conversion of all channels of sequence B ends, the sequence B conversion end flag ADC_ISR.EOCBF is set to 1, and the sequence B conversion end event ADC_EOCB is generated. If ADC_ISCR.EOCBIEN is 1, the interrupt is enabled and the sequence B conversion end interrupt is generated at the same time. ask.

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 16-3 Sequences A and B of Various Competitions

A/D conversion	Trigger signal occurrence	Mode of treatment	
		ADC_CR1.RSCHSEL=0	ADC_CR1.RSCHSEL=1
Sequence A conversion in process	Sequence A triggering	Invalid trigger signal	
	Sequence B triggering	1) The sequence A conversion is interrupted and the sequence B conversion is started. 2) After the conversion of sequence B is complete, sequence A continues to be converted from the channel interruptd.	1) The sequence A conversion is interruptd and the sequence B conversion is started. 2) After the conversion of sequence B is complete, sequence A is reconverted from the first channel.
Sequence B conversion in process	Sequence A triggering	Sequence B All channel conversion is complete and sequence A conversion is started.	
	Sequence B triggering	Invalid trigger signal	
In AD idle, sequence A and B are triggered simultaneously		Sequence B starts first, after all channel conversions are complete, sequence A conversions are started.	

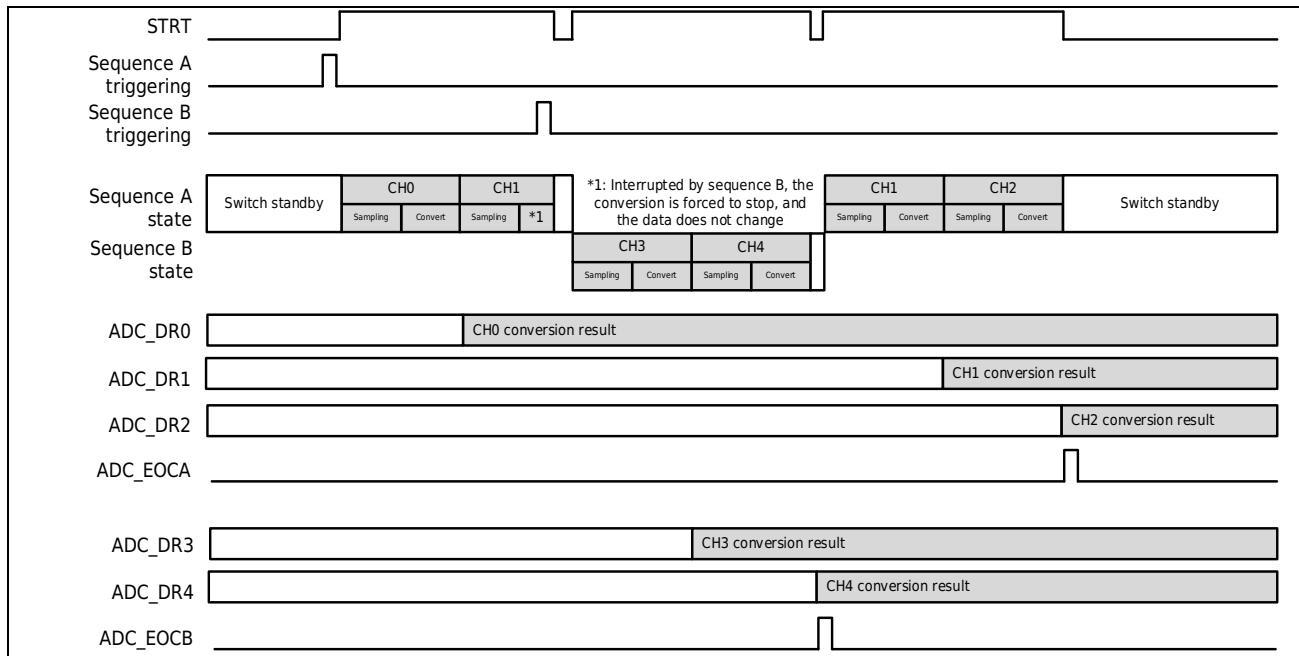


Figure 16-5 Double sequence scan mode (sequence A restarts from interrupted channel)

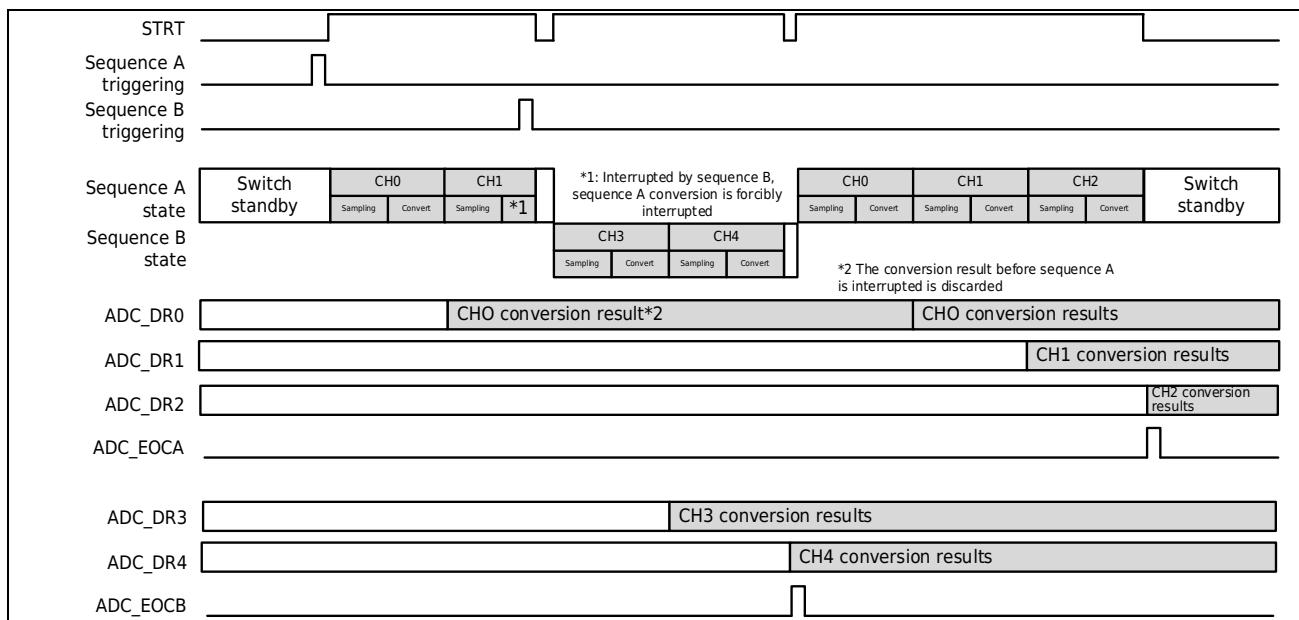


Figure 16-6 Double sequence scan mode (sequence A restarts from the first channel)

The software process of double sequence scanning mode:

1. Confirm that ADC_STR.STRT is 0, ADC is in conversion standby state.
2. The A/D control register ADC_CR0.MS[1:0] is set to 10b or 11b to select the dual sequence scan mode.
3. Set register ADC_CR1.RSCHSEL selects the start mode after sequence A is broken.
4. Set the sequence A channel select register ADC_CHSELRA.
5. Set the sequence B channel select register ADC_CHSELRB.
6. Set the sampling time register ADC_SSTR.

7. Set the register ADC _ TRGSR selection sequence A and B trigger conditions.
8. By querying EOCAF, EOCBF, or ADC _ EOCA, or ADC _ EOCB interrupt, or starting DMA to process the converted data after the sequence A or B conversion ends.

Note:

- Do not select the same channel in sequence A and B. Do not select the same trigger source for sequences A and B.

16.3.7 Analog watchdog Function

The analog watchdog function refers to comparing the conversion results at the end of the A/D conversion of the channel, as shown in the figure below, if the conversion result is within the protection area, a channel comparison interrupt and event ADC_CHCMP will be generated . And at the end of the entire sequence scan, a sequence comparison interrupt and event ADC_SEQCMP are generated according to the comparison results of each channel. You can choose to compare any single or multiple channels. Multiple channels are more effective, and if any channel is consistent, sequence comparison interrupts and events will be generated.

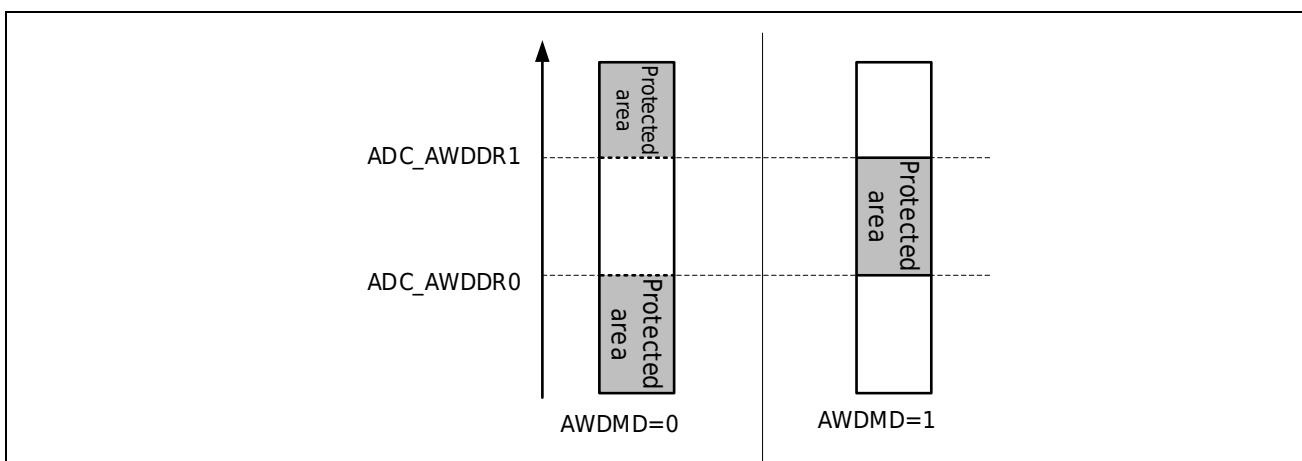


Figure 16-7 Analog Watchdog Protection Area (comparison conditions)

Software flow for using the analog watchdog function:

1. Set the threshold register AD_AWDDR0, ADC_AWDDR1
2. Set the comparison channel register ADC_AWDCHSR to select any single or multiple channels to be compared
3. Set ADC_AWDCR.AWDMD to select the comparison condition
4. Set ADC_AWDCR.AWDSS[1:0] to select sequence comparison interrupt and event output, and set ADC_AWDCR.AWDIEN interrupt enable bit.
5. Setting ADC_AWDCR.AWDEN allows analog watchdog function
6. According to the previous text, set scan mode, start AD for conversion.
7. During the ADC_CHCMP/ADC_SEQCMP interrupt, or after the A/D conversion is completed, query the comparison status register ADC_AWDSR, and perform corresponding processing on the comparison result.

16.3.8 Sampling Time and Conversion Time of Analog Input

In single-scan mode, A/D conversion can select software settings, internal trigger IN_TRGx0,1 and external pin trigger ADTRGx start mode. After the scan conversion delay time t_D , the ADC module starts to sample and convert the analog channel. After all the conversions are completed, the conversion end delay time t_{ED} enters the standby state, and a scan is finally completed. The continuous scanning mode is similar to a single scanning mode, except that there is no t_D time at the second time of the sequence and at the start of the sequence.

The conversion time of a single channel is $t_{CONV} = t_{SPL} + t_{CMP}$. Among them, t_{SPL} represents the sampling time of the analog input, and the number of sampling cycles can be adjusted according to the input impedance setting register ADC_SSTRx. t_{CMP} means successive comparison time, 13 ADCLKs for 12-bit precision, 11 ADCLKs for 10-bit precision, and 9 ADCLKs for 8-bit precision.

The time of one scan conversion $t_{SCAN} = t_D + \sum t_{CONV} + t_{ED}$. Among them, $\sum t_{CONV}$ 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.

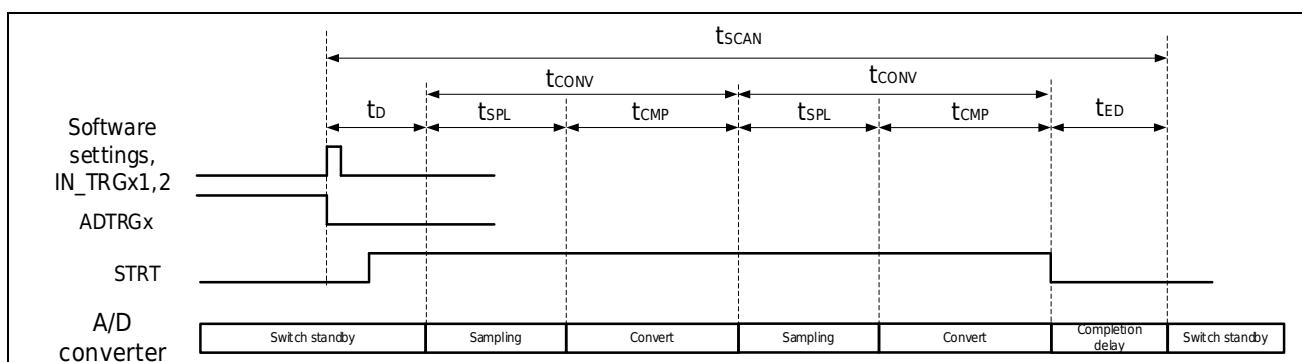


Figure 16-8 A/D conversion time

Table 16-4 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	1 PCLK4 + 4 ADCLK	3 PCLK4 + 4 ADCLK + 1 PCLK4_SYNC	3 PCLK4 + 4 ADCLK	4 ADCLK
		Interrupted in sequence A conversion, starting sequence B conversion	2 PCLK4 + 6 ADCLK	4 PCLK4 + 6 ADCLK + 1 PCLK4_SYNC	4 PCLK4 + 6 ADCLK	-
t_{CONV}	t_{SPL}	Sampling time		ADSSTRx.SST[7:0] x ADCLK		
	t_{CMP}	Successive conversion time	12 bit resolution	13 ADCLK		
			10-bit resolution	11 ADCLK		
			8 bit resolution	9 ADCLK		
t_{ED}		Scan completion time		1 PCLK4 + 3 ADCLK		
t_{TD}		Minimum continuous trigger interval		$\Sigma t_{CONV} + 2 PCLK4 + 5 ADCLK$		

Note:

- Asynchronous peripheral triggering refers to the peripheral triggering situation when the ADC module selects the PLL clock operation that is asynchronous to the system clock. At this time, the peripheral module clock and the ADC module clock are in an asynchronous relationship. PCLK4_SYNC indicates the original synchronous clock of the ADC module. At this time, PCLK4 and ADCLK are the same, and both are asynchronous PLL clocks.

16.3.9 Automatic Clearing Function of A/D Data Register

When ADC_CR0.CLREN is '1' and 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" and auto-purge is disabled, the A/D conversion result (0x0222) is not updated to the data register ADC_DR for some reason, and the ADC_DR register continues to maintain 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 updated, you need to store the previous conversion value in RAM to determine by comparing the conversion results.
- If ADC_CR0.CLRE is "1", when 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.

16.3.10 Conversion 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 saving the conversion results to the data register after averaging. Using the average calculation function can remove certain noise components to make the conversion result more accurate.

The register ADC_CR0.AVCNT[2:0] sets the number of continuous conversions, and the register ADC_AVCHSEL selects any one or more channels that need to be averaged.

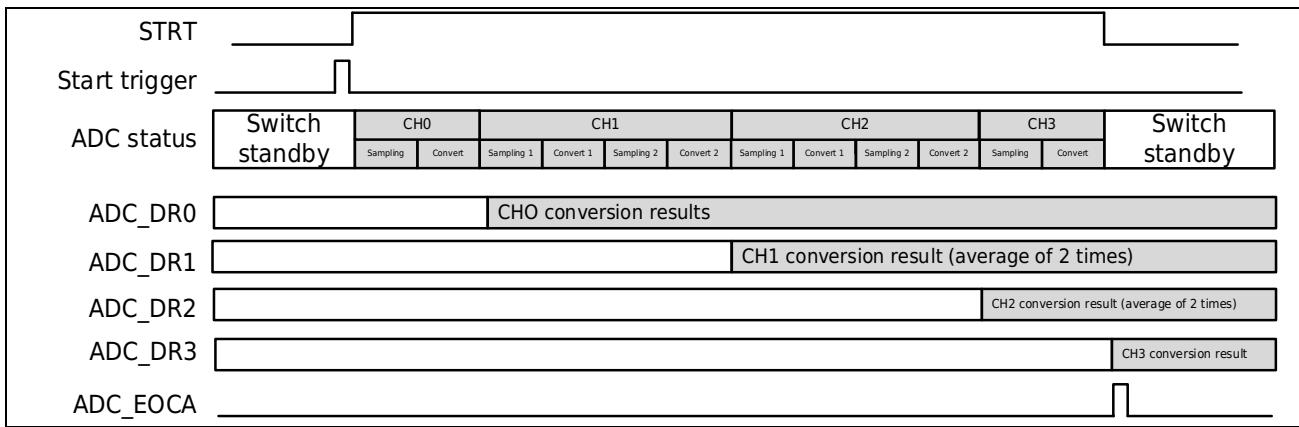


Figure 16-9 Conversion action when averaging function is valid

In Figure 16-9, sequence A single scan mode, choose to convert the 4 channels CH0~3 , among which CH1 and 2 are set to the 2 -time average mode. During the scanning process, CH1 and 2 will perform two consecutive conversions, and save the averaged results to the data registers ADC_DR1 and 2 of the corresponding channel.

16.3.11 Programmable Gain Amplifier PGA

When equipped with a programmable gain amplifier PGA, you can set the register ADC_PGAINSR to select the input source of the PGA, set the register ADC_PGACR to make the PGA circuit effective, set the register ADC_PGAGSR to select the gain multiple, and the gain range x2~x32 can be selected. At this time, the analog input is first amplified by the PGA circuit, and then input to the ADC module for conversion.

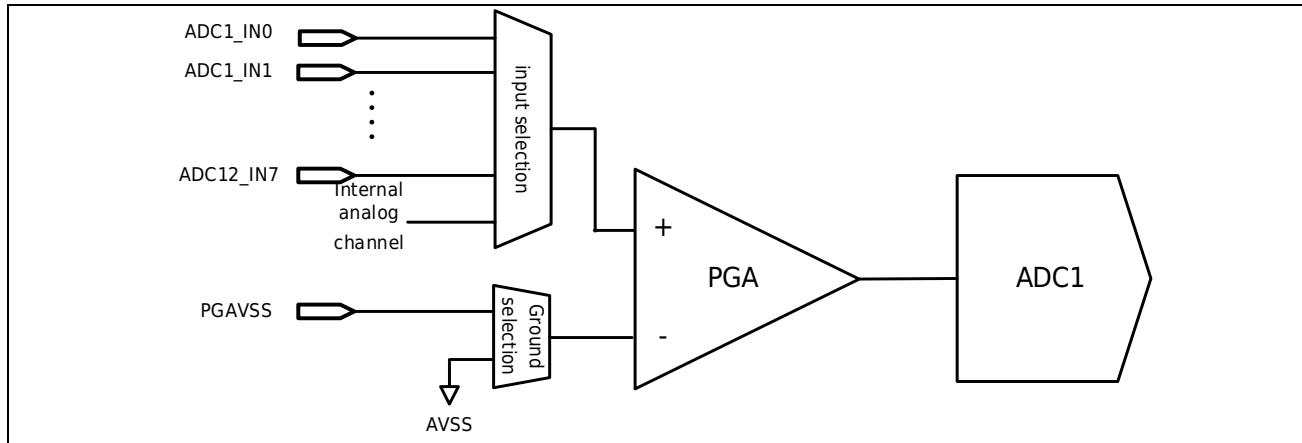


Figure 16-10 Schematic diagram of PGA work

For example, if you need to convert ADC1_IN2 after amplification, you can set the channel selection register ADC_CHSELRA0 to 0x4 to select CH2 (ADC1_IN2), set the PGA input source register ADC_PGAINS to 0x4 to select ADC1_IN2, set the PGA to be valid and the amplification factor and then start the conversion. Similarly, if you need to amplify and convert the internal analog channel, you can set ADC_CHSELRA1 to 0x1 to select CH16, set ADC_PGAINS to 0x100, and then start the conversion.

Note:

- When the PGA input source selects the internal analog channel, the analog value is fixed as 8bitDAC_1 output, regardless of the CMP_RVADC setting.
- Only ADC1 supports PGA.

16.3.12 Multi-ADC Cooperative Working Mode

On chips with two or three ADC blocks, ADC cooperative mode can be used.

In the ADC cooperation mode, the conversion of ADC2 and ADC3 is synchronized by the trigger signal of ADC1. That is, the setting of the sequence A trigger source selection register ADC_TRGSR.TRGSEL[2:0] of ADC2 and ADC3 is invalid. All ADC modules are triggered by the trigger source selected by the Sequence A trigger source selection register of ADC1. In this mode, writing 1 to the ADC_STR.START register will not start the conversion, that is, the software start is invalid.

When using the cooperative work mode, please prohibit the sequence B action, so as not to disturb the synchronization.

You can set the two ADC modules ADC1 and ADC2 to work together, or you can set the three ADC modules ADC1, ADC2 and ADC3 to work together. Depending on the specific specifications of the product, ADC3 may not be equipped.

The ADC can be configured into the following four cooperative working modes:

- One-Shot Parallel Trigger Mode

- One-shot delayed trigger mode
- Loop Parallel Trigger Mode
- or cyclic delay trigger mode

One-Shot Parallel Trigger Mode

The Sequence A trigger condition of ADC1 triggers all ADC modules in cooperative mode at the same time, and triggers only once.

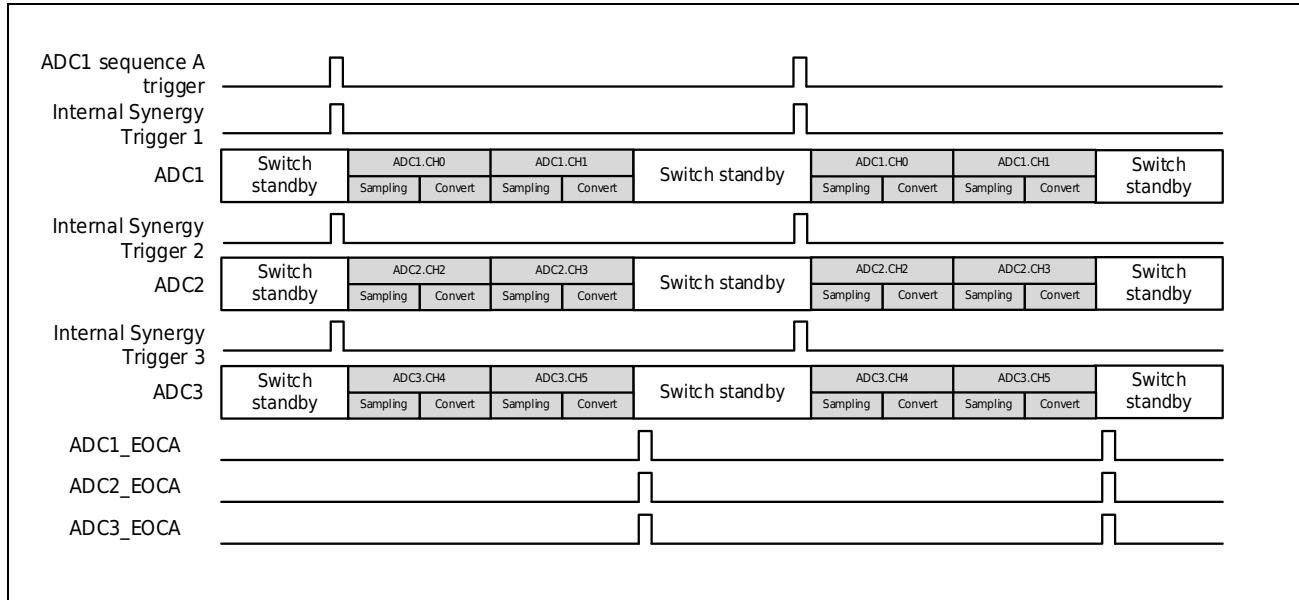


Figure 16-11 Single Parallel Trigger Mode (Three ADCs)

Note:

- It is forbidden for multiple ADCs to convert the same analog input at the same time, and one analog channel can only sample one ADC module at the same time, the same below.

The software setting process of this mode is as follows:

1. Write 0 to the collaborative work permission register ADC_SYNCCR.SYNCEN to confirm that the collaborative work is invalid.
2. Set up the ADC1 module
 - a) Confirm that ADC1_STR.STRT is 0, and ADC1 is in conversion standby state.
 - b) Set the control register ADC1_CR0.MS[1:0] to 00b: sequence A single scan mode, or 01b: sequence A continuous scan mode
 - c) Set Sequence A Channel Select Register ADC1_CHSELRA
 - d) Set the sampling time register ADC1_SSTR
 - e) Set the sequence A trigger source selection register ADC1_TRGSR
3. Set up the ADC2 module
 - a) Confirm that ADC2_STR.STRT is 0, and ADC1 is in conversion standby 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 as far as possible. The specific channels do not need to be the same, as long as the number of channels and the sampling time of the corresponding channels are consistent.

4. Set the ADC3 module (when three ADCs work together)

- Confirm that ADC3_STR.SRT is 0, and ADC2 is in conversion standby state.
- Set the control register ADC3_CR0.MS[1:0], the channel selection register ADC3_CHSELRA, and the channel sampling time register ADC3_SSTR.

Note:

- 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 much as possible.

- Set the synergy mode control register ADC_SYNCCR.SYNCMD[2:0], write 010b: ADC1 and ADC2 work together. Or write 011b: ADC1, ADC2, ADC3 three ADCs work together.
- Write 1 to the collaborative work permission register ADC_SYNCCR.SYNCEN, and the collaborative work is valid.
- Wait for ADC1 sequence A trigger source input, and process the result after ADC1, ADC2, ADC3 complete conversion.

One-shot delayed trigger mode

After the sequence A trigger condition of ADC1 triggers ADC1, it triggers ADC2 to start conversion after a set delay, and then triggers ADC3 to start conversion after a set delay. Each ADC module is only triggered once.

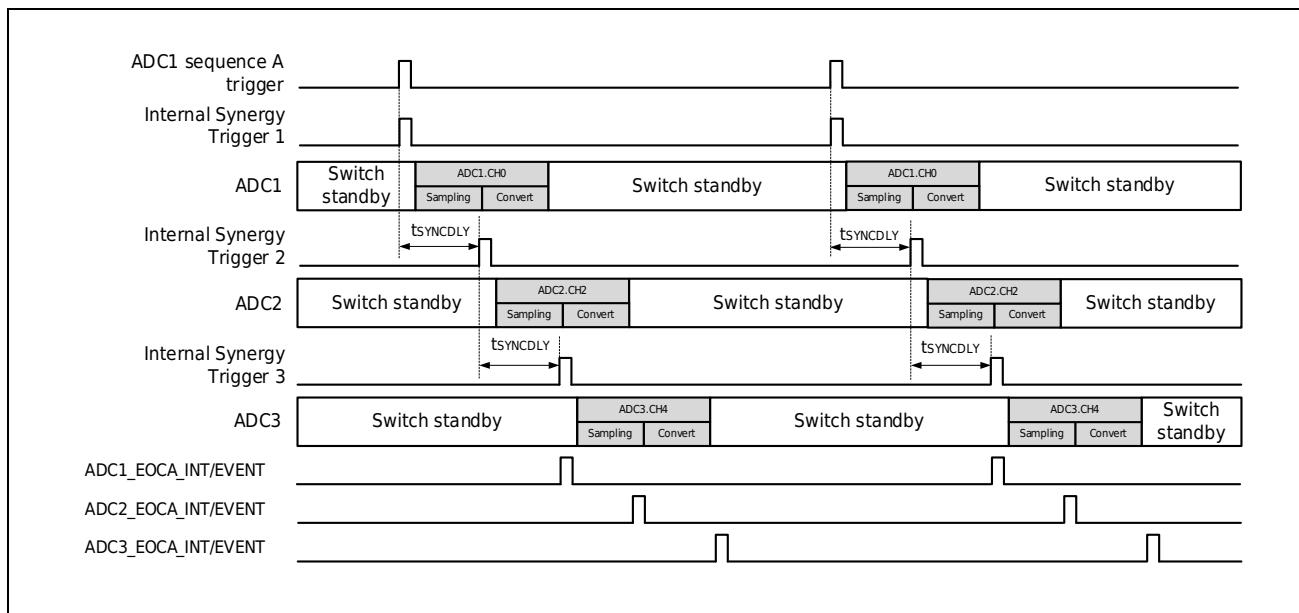


Figure 16-12 Single delay trigger mode (three ADCs)

Note:

- After the ADC1 sequence A trigger is input for the first time, before the ADC3 coordinated 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 opening sampling time t_{SPL} need to satisfy: $t_{SYNCDLY} > t_{SPL}$.

The software setting process of this mode is as follows:

1. Write 0 to the collaborative work permission register ADC_SYNCCR.SYNCEN to confirm that the collaborative work is invalid.
2. Set ADC1, ADC2, ADC3 modules (refer to single parallel mode)
3. Set the cooperative mode control register ADC_SYNCCR.SYNCDLY[7:0] to set the startup delay of the two ADCs.
4. Set the cooperative mode control register ADC_SYNCCR.SYNCMD[2:0], write 000b: ADC1 and ADC2 work together. Or write 001b: ADC1, ADC2, ADC3 and three ADCs work together.
5. Write 1 to the collaborative work permission register ADC_SYNCCR.SYNCEN, and the collaborative work is valid.
6. Wait for ADC1 sequence A trigger source input, and process the result after ADC1, ADC2, ADC3 complete conversion.

Loop Parallel Trigger Mode

The Sequence A trigger condition of ADC1 triggers all ADC modules in cooperative mode simultaneously, and then triggers all ADC modules simultaneously again after each specified delay. Until the user initiates the software to stop the ADC1 module, or prohibits the cooperative work mode.

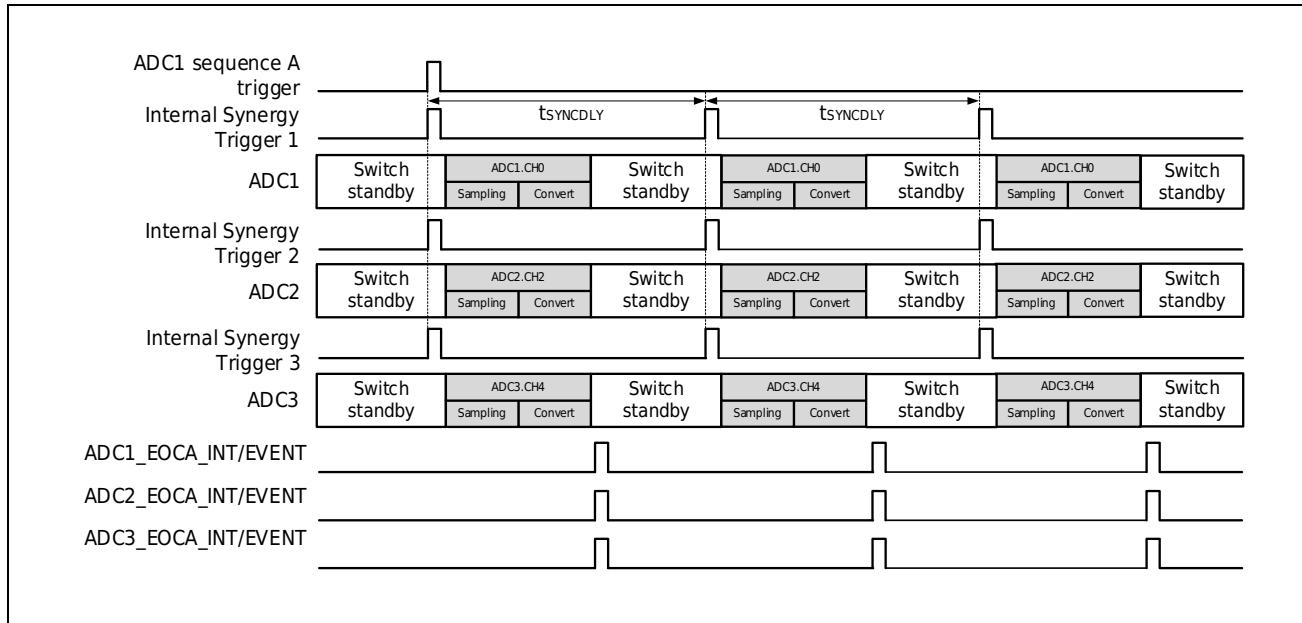


Figure 16-13 Cyclic Parallel Trigger Mode (Three ADCs)

Note:

- The delay time $t_{SYNCDLY}$ and the time t_{SCAN} of a scan conversion need to satisfy: $t_{SYNCDLY} > t_{SCAN}$. The software setting process of this mode is as follows:

1. Write 0 to the collaborative work permission register ADC_SYNCCR.SYNCEN to confirm that the collaborative work is invalid.
2. Set ADC1, ADC2, ADC3 modules, refer to single parallel mode. ADC_CR0.MS[1:0] set to 00b: Sequence A single scan mode
3. Set the cooperative mode control register ADC_SYNCCR.SYNCDLY[7:0] to set the delay of each parallel trigger.
4. Set the cooperative mode control register ADC_SYNCCR.SYNCMD[2:0], write 110b: ADC1 and ADC2 work together. Or write 111b: ADC1, ADC2, ADC3 and three ADCs work together.
5. Write 1 to the collaborative work permission register ADC_SYNCCR.SYNCEN, and the collaborative work is valid.
6. Wait for ADC1 sequence A trigger source input, and process the result after ADC1, ADC2, ADC3 complete conversion.

Cyclic delay trigger mode

After the sequence A trigger condition of ADC1 triggers ADC1, after each set delay, it will continuously trigger ADC2, ADC3, ADC1, ADC2... until the user actively software stops the ADC1 module or disables the cooperative working mode.

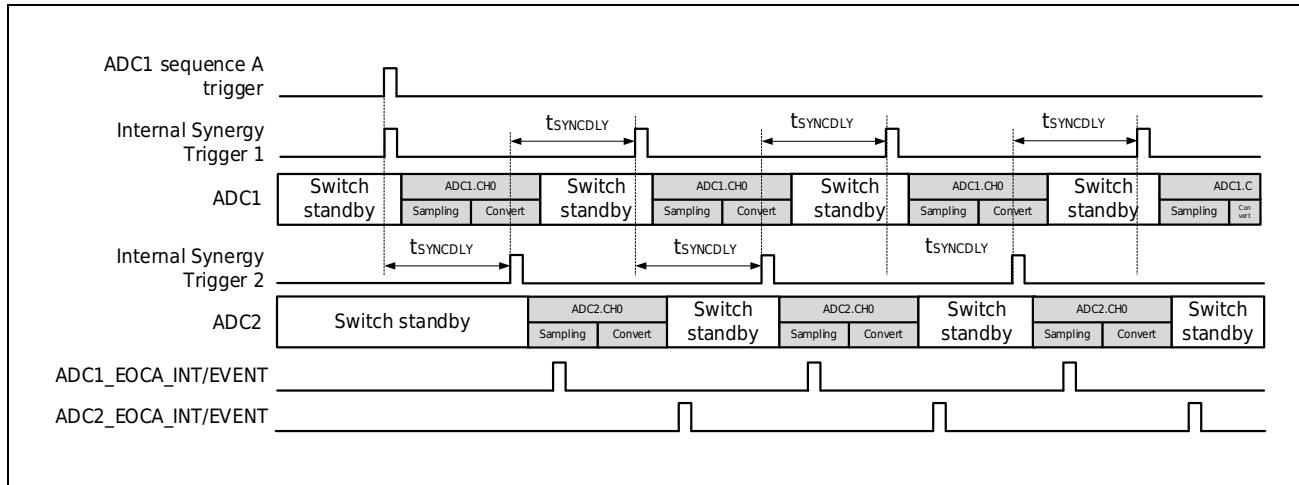


Figure 16-14 Cyclic Delay Trigger Mode (Two ADCs)

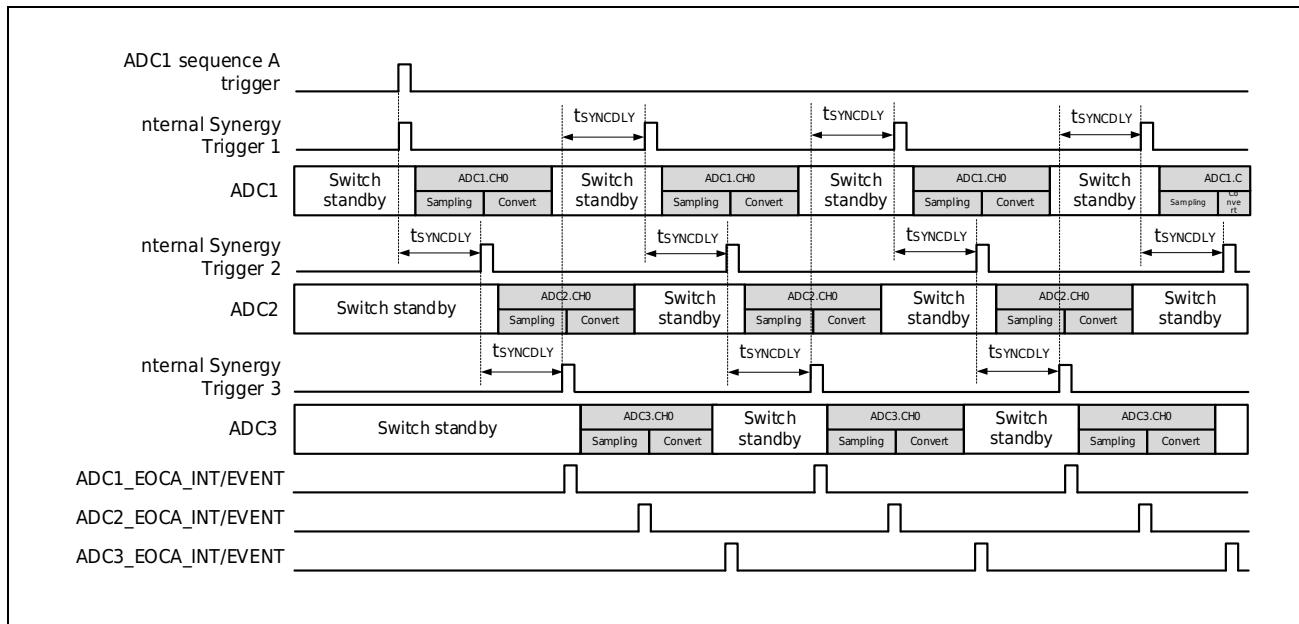


Figure 16-15 Cyclic Delay Trigger Mode (Three ADCs)

Note:

- When the two ADCs work together, The delay time $t_{SYNCDLY}$ and the time t_{SCAN} of a scan conversion need to satisfy: $t_{SYNCDLY} > t_{SCAN}/2$. When the three ADCs work together, it needs to satisfy: $t_{SYNCDLY} > t_{SCAN}/3$. At the same time, if ADC1, ADC2, and ADC3 convert the same analog channel, the sampling time needs to be staggered, that is, $t_{SYNCDLY} > t_{SPL}$.

The software setting process of this mode is as follows:

- Write 0 to the collaborative work permission register ADC_SYNCCR.SYNCEN to confirm that the collaborative work is invalid.

2. Set ADC1, ADC2, ADC3 modules, reference cycle parallel trigger mode.
3. Set the cooperative mode control register ADC_SYNCCR.SYNCDLY[7:0] to set the delay of each trigger.
4. Set the cooperative mode control register ADC_SYNCCR.SYNCMD[2:0], write 100b: ADC1 and ADC2 work together. Or write 101b: ADC1, ADC2, ADC3 and three ADCs work together.
5. Write 1 to the collaborative work permission register ADC_SYNCCR.SYNCEN, and the collaborative work is valid.
6. Wait for ADC1 sequence A trigger source input, and process the result after ADC1, ADC2, ADC3 complete conversion.

16.3.13 Interrupt and Event Signal Output

The ADC module can produce the following four types of event outputs. When each event occurs, if the corresponding interrupt permission register is set to 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. Channel comparison ADC_CHCMP, corresponding to the interrupt permission register ADC_AWDCR.AWDIEN
4. Sequence comparison ADC_SEQCMP, corresponding to the interrupt permission register ADC_AWDCR.AWDIEN

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, completely implemented by hardware, reducing 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, as long as the condition occurs, it will be output.

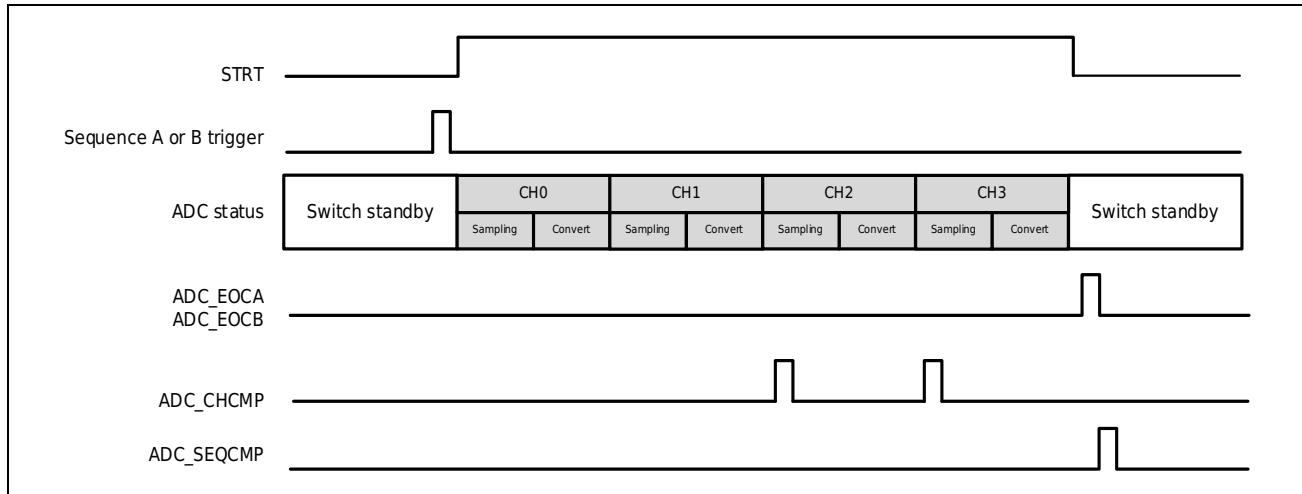


Figure 16-16 ADC Interrupt and Event Output Timing

In the figure above, the comparison conditions of CH1 and CH2 watchdogs are satisfied. The channel compare event occurs at the end of each channel conversion, and the sequence compare event occurs at the end of the sequence scan, one cycle later than the sequence scan event ADC_EOCA/B.

16.4 Register Description

Register overview

Unit 1 BASE_ADDR: 0x4004_0000

Unit 2 BASE_ADDR: 0x4004_0400

Table 16-5 List of ADC Registers 1/2

Register name	Symbol	Offset address	Bit width	Reset value
A/D boot register	ADC_STR	0x00	8	0x00
A/D control register0	ADC_CR0	0x02	16	0x0000
A/D control register1	ADC_CR1	0x04	16	0x0000
The A/D conversion starts triggering the register	ADC_TRGSR	0x0A	16	0x0000
A/D channel selection registerA0	ADC_CHSELRA0	0x0C	16	0x0000
A/D channel selection register A1	ADC_CHSELRA1	0x0E	16	0x0000
A/D channel selection registerB0	ADC_CHSELRB0	0x10	16	0x0000
A/D channel selection register B1	ADC_CHSELRB1	0x12	16	0x0000
A/D average channel selection register 0	ADC_AVCHSELR0	0x14	16	0x0000
A/D average channel selection register 1	ADC_AVCHSELR1	0x16	16	0x0000
A/D sampling period	ADC_SSTRx	0x20+x	8	0x0B
	ADC_SSTRL	0x30	8	0x0B
A/D channel mapping control register 0	CHMUXR0	0x38	16	0x3210
A/D channel mapping control register 1	CHMUXR1	0x3A	16	0x7654
A/D channel mapping control register 2	CHMUXR2	0x3C	16	0xBA98
A/D channel mapping control register 3	CHMUXR3	0x3E	16	0xFEDC
A/D interrupt state register	ADC_ISR	0x46	8	0x00
A/D interrupt enable register	ADC_ICR	0x47	8	0x03
A/D cooperative mode control register	ADC_SYNCCR	0x4C	16	0x0C00
A/D data register	ADC_DRx	0x50+2*x	16	0x0000
Analog watchdog control register	ADC_AWDCR	0xA0	16	0x0000
Analog Watchdog Threshold Register	ADC_AWDDR0	0xA4	16	0x0000
	ADC_AWDDR1	0xA6	16	0x0000
Analog Watchdog Compare Channel Select Register	ADC_AWDCHSR0	0xAC	16	0x0000
Analog Watchdog Compare Channel Select Register 1	ADC_AWDCHSR1	0xAE	16	0x0000
Analog watchdog status register	ADC_AWDSR0	0xB0	16	0x0000
Analog Watchdog Status Register 1	ADC_AWDSR1	0xB2	16	0x0000
A/D Programmable Gain Amplifier Control Register	ADC_PGACR	0xC0	16	0x0000
A/D Programmable Gain Multiplier Register	ADC_PGAGSR	0xC2	16	0x0000
A/D Programmable Gain Amplifier Input Selection Register	ADC_PGAISNR0	0xCC	16	0x0000
A/D Programmable Gain Amplifier Input Selection Register 1	ADC_PGAISNR1	0xCE	16	0x0000

16.4.1 A/D Launches Register (ADC _ STR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
Bit	Marking	Place name	Function				R/W
b7-b1	—	—	Read at 0, write at 0				R/W
			0: Stop conversion 1: Start conversion Set the "1" condition: (1) Software setup (2) The selected trigger condition occurs. (3) A/D conversion "0" condition: (1) Software "0" (2) "0" will be cleared automatically after conversion.				
b0	STRT	Start of AD conversion	Note: <ul style="list-style-type: none"> – Write 1 generates a software trigger when STRT is 0 (ADC is idle), starting sequence A – Write 1 is invalid when STRT is 1 (in ADC action). – Write 0 when STRT is 1 to force the stop of AD conversion. If the ADC _ TRGSR sets a value other than 0x0 and does not want the ADC to restart, set the ADC _ TRGSR to 0 before writing 0 to the STRT. – Write 0 is invalid when STRT is 0. 				R/W

16.4.2 A/D Control Register 0 ADC_CR0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
—	—	—	—	—	AVCNT[2:0]		
b7	b6	b5	b4	b3	b2	b1	b0
DFMT	CLREN	ACCSEL[1:0]		—	—	MS[1:0]	

Bit	Marking	Place name	Function	R/W
b15-b11	—	—	Read at 0, write at 0	R/W
b10-b8	AVCNT[2:0]	Times selection	0 0 0: Average of 2 consecutive conversions 0 0 1: continuous conversion 4 average 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 purge	0: Automatic removal of prohibitions 1: Automatic clearance Note: After the CLREN bit is set, registerADC_DRx is automatically cleared after 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	0 0: 12-bit resolution 0 1: 10 bit resolution 1 0: 8 bit resolution 1 1: Prohibitions	R/W
b3-b2	—	—	Read at 0, write at 0	R/W
b1-b0	MS[1:0]	Pattern 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".

16.4.3 A/D Control Register1 (ADC _ CR1)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
—	—	—	—	—	—	—	—
b7	b6	b5	b4	b3	b2	b1	b0
—	—	—	—	—	RSCHSEL	—	—

Bit	Marking	Place name	Function	R/W
b15-b3	—	—	Read at 0, write at 0	R/W
b2	RSCHSEL	Sequence A Restart Channel Selection	0: After sequence Binterrupt, sequence A continues to scan from the interrupt channel when it restarts 1: Rescan from the first channel when sequence A restarts after sequence Binterrupt	R/W
b1-b0	—	—	Read at 0, write at 0	R/W

Note:

- Set this register when ADC_STR. STRT is "0".

16.4.4 A/D Conversion Starts Triggering Register (ADC _ TRGSR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
TRGENB	—	—	—	—	TRGSELB[2]	TRGSELB[1]	TRGSELB[0]
b7	b6	b5	b4	b3	b2	b1	b0
TRGENA	—	—	—	—	TRGSELB[2]	TRGSELB[1]	TRGSELB[0]

Bit	Marking	Place name	Function	R/W
b15	TRGENB	Sequence B trigger enable	0: Sequence B in-chip or external pin trigger disable 1: Sequence B in-chip or external pin trigger enable Note: Select External Pin Trigger is valid. If ADTRGx is changed from "High" to "Low" and a falling edge is detected, the scan conversion begins. Keep "Low" 1.5 * PCLK4 cycles above.	R/W
b14-b11	—	—	Read at 0, write at 0	R/W
b10-b8	TRGSELB[2:0]	Sequence B trigger condition selection	In sequence B active mode (ADC _ CR0.MS [1] = 1) as trigger condition of sequence B 000b: ADTRGx 001b: IN_TRGx0 010b: IN_~TRGx1 011b: IN_TRGx0 + IN_~TRGx1 Other: not selected x=1,2 Note: Only valid in sequence B active mode. Other mode settings are invalid. The interval between triggers must be greater than or equal to the scan period tSCAN, and if less, the trigger is invalid.	R/W
b7	TRGENA	Sequence A trigger enable	0: Sequence A in-chip or external pin trigger disable 1: Sequence A in-chip or external pin trigger enable Note: Select External Pin Trigger is valid. If ADTRGx is changed from "High" to "Low" and a falling edge is detected, the scan conversion begins. Keep "Low" 1.5 * PCLK4 cycles above.	R/W
b6-b3	—	—	Read at 0, write at 0	R/W
b2-b0	TRGSELB[2:0]	Sequence A trigger condition selection	Trigger condition for sequence A. 000b:ADTRGx 001b:IN_TRGx0 010b:IN_~TRGx1 011b: IN_TRGx0 + IN_~TRGx1 Other: not selected n=1,2 Note: ADC _ STR.START Write 1 software triggers, disregarding TRGENA or TRGSELB [2: 0] settings, starting A/D conversion directly. The interval between triggers must be greater than or equal to the scan period tSCAN, and if less, the trigger is invalid.	R/W

Note:

- Set this register when ADC _ STR. START is "0".

16.4.5 A/D Channel Select RegisterA (ADC_CHSELRA0)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CHSEL[A15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
CHSEL[A7:0]							

Bit	Marking	Place name	Function	R/W
b15-b0	CHSEL[A15:0]	Switching channel selection	Channel selection for sequence A, each representing a channel, CHSEL[A] [x] representing a channel CHx, can be any combination. 0: No channel selected 1: Select Corresponding Channels The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. Note: Do not select the same channel in sequence A and sequence B.	R/W

Note:

- Set this register when ADC_STR. STRT is "0".

16.4.6 A/D Channel Select Register A 1 ADC_CHSELRA1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CHSEL[A31:24]							
b7	b6	b5	b4	b3	b2	b1	b0
CHSEL[A23:16]							

Bit	Marking	Place name	Function	R/W
b15-b0	CHSEL[A31:16]	Switching channel selection	Channel selection for sequence A, each representing a channel, CHSEL[A] [x] representing a channel CHx, can be any combination. 0: No channel selected 1: Select Corresponding Channels The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. Note: Do not select the same channel in sequence A and sequence B.	R/W

Note:

- Set this register when ADC_STR. STRT is "0".

16.4.7 A/D Channel Selection RegisterB (ADC _ CHSELRB0)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CHSELB[15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
CHSELB[7:0]							

Bit	Marking	Place name	Function	R/W
b15-b0	CHSELB[15:0]	Switching channel selection	Channel selection for sequence B, each representing a channel, CHSELB [x] representing a channel CHx, can be any combination. Only valid in double sequence scan mode. 0: No channel selected 1: Select Corresponding Channels The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. Note: Do not select the same channel in sequence A and sequence B.	R/W

Note:

- Set this register when ADC_STR. STRT is "0".

16.4.8 A/D Channel Select Register B1 ADC_CHSELRB1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CHSELB[31:24]							
b7	b6	b5	b4	b3	b2	b1	b0
CHSELB[23:16]							

Bit	Marking	Place name	Function	R/W
b15-b0	CHSELB[31:16]	Switching channel selection	Channel selection for sequence B, each representing a channel, CHSELB [x] representing a channel CHx, can be any combination. Only valid in double sequence scan mode. 0: No channel selected 1: Select Corresponding Channels The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. Note: Do not select the same channel in sequence A and sequence B.	R/W

Note:

- Set this register when ADC_STR. STRT is "0".

16.4.9 A/D Average Channel Selection Register ADC_AVCHSELRO

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AVCHSEL[15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
AVCHSEL[7:0]							

Bit	Marking	Place name	Function	R/W
b15-b0	AVCHSEL[15:0]	Average channel selection	Each bit represents a channel, and AVCHSEL[x] represents the channel CHx, and any combination can be selected. 0: No channel selected 1: Select Corresponding Channels The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. Note: When AVCHSEL and the corresponding channel of ADC_CHSELRA or ADC_CHSELB are selected at the same time, the channel will perform the set number of A/D conversions continuously 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 conversion.	R/W

Note:

- Set this register when ADC_STR. STRT is "0".

16.4.10 A/D Average Channel Selection Register 1 ADC_AVCHSELR1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AVCHSEL[31:24]							
b7	b6	b5	b4	b3	b2	b1	b0
AVCHSEL[23:16]							

Bit	Marking	Place name	Function	R/W
b15-b0	ADAVSEL[31:16]	Average channel selection	Each bit represents a channel, and AVCHSEL[x] represents the channel CHx, and any combination can be selected. 0: No channel selected 1: Select Corresponding Channels The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. Note: When AVCHSEL and the corresponding channel of ADC_CHSELRA or ADC_CHSELB are selected at the same time, the channel will perform the set number of A/D conversions continuously 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 conversion.	R/W

Note:

- Set this register when ADC_STR. STRT is "0".

16.4.11 A/D Sampling Status Register (ADC_SSTR)

Reset value: 0x0B

b7	b6	b5	b4	b3	b2	b1	b0
SST[7:0]							
Bit	Marking	Place name	Function				
b7-b0	SST[7:0]	Number of sampling cycles	The number of sampling cycles can be set from 5 to 255 cycles. Channels CH0~15 are set by ADC_SSTRx, x=0~15, and other channels are set by ADC_SSTRL. Note: When ADCLK frequency is 50 MHz, one sampling period is 20 ns, and the initial conversion state has 11 sampling periods. Register can be set to adjust sampling time when external input impedance RAIN is too large or ADCLK frequency is low. The sampling time should not be less than 5 cycles. $SST \geq (RAIN + R_{ADC}) * C_{ADC} * \ln(2^{N+2}) * f_{ADC} + 1$ Where: RAIN represents external input impedance (Ω), R_{ADC} represents internal sampling switch resistance (Ω), C_{ADC} represents internal sampling and retention capacitance (F), N represents AD resolution (12/10/8), and f_{ADC} represents ADCLK frequency (Hz). Refer to the description of electrical characteristics.				R/W

Note:

- Set this register when ADC_STR. STRT is "0".

16.4.12 A/D Channel Mapping Control Register ADC_CHMUXR

ADC_CHMUXR0 reset value: 0x3210

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CH03MUX[3:0]				CH02MUX[3:0]				CH01MUX[3:0]				CH00MUX[3:0]			

ADC_CHMUXR1 reset value: 0x7654

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CH07MUX[3:0]				CH06MUX[3:0]				CH05MUX[3:0]				CH04MUX[3:0]			

ADC_CHMUXR2 reset value: 0xBA98

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CH11MUX[3:0]				CH10MUX[3:0]				CH09MUX[3:0]				CH08MUX[3:0]			

ADC_CHMUXR3 reset value: 0xFEDC

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CH15MUX[3:0]				CH14MUX[3:0]				CH13MUX[3:0]				CH12MUX[3:0]			

Bit	Marking	Place name	Function	R/W
The corresponding bit of the channel that does not exist is 0 when reading, and writes 0 when writing.				
The mapping relationship for 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	
	0x0	ADC1_IN0	ADC12_IN4	
	0x1	ADC1_IN1	ADC12_IN5	
	0x2	ADC1_IN2	ADC12_IN6	
	0x3	ADC1_IN3	ADC12_IN7	
	0x4	ADC12_IN4	ADC12_IN8	
	0x5	ADC12_IN5	ADC12_IN9	
	0x6	ADC12_IN6	ADC12_IN10	
	0x7	ADC12_IN7	ADC12_IN11	
	0x8	ADC12_IN8	Internal analog channel	
	0x9	ADC12_IN9	—	
	0xa	ADC12_IN10	—	
	0xb	ADC12_IN11	—	
	0xc	ADC1_IN12	—	
	0xd	ADC1_IN13	—	
	0xe	ADC1_IN14	—	
	0xf	ADC1_IN15	—	
Note: Please do not set it to an analog input that does not exist.				

Note:

- Set this register when ADC_STR. STRT is "0".

16.4.13 A/D Interrupt State Register (ADC_ISR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
—	—	—	—	—	—	EOCBF	EOCAF

Bit	Marking	Place name	Function	R/W
b7-b2	—	—	Read at 0, write at 0	R/W
b1	EOCBF	Sequence B conversion completion flag	Sequence B Scan all channels selected 1 When the register is set and needs to be cleared, please write "0" after reading "1".	R/W
b0	EOCAF	Sequence A conversion completion flag	Sequence A Selected Channels Scan All Post-1 When the register is set and needs to be cleared, please write "0" after reading "1".	R/W

16.4.14 A/D Interrupt License Register (ADC_ICR)

Reset value: 0x03

b7	b6	b5	b4	b3	b2	b1	b0
—	—	—	—	—	—	EOCBIEN	EOCAIEN

Bit	Marking	Place name	Function	R/W
b7-b2	—	—	Read at 0, write at 0	R/W
b1	EOCBIEN	Sequence B conversion complete interrupt enable	0: Sequence B conversion complete interrupt disable 1: Sequence B conversion complete interrupt enable	R/W
b0	EOCAIEN	Sequence A conversion complete interrupt enable	0: Sequence A conversion complete interrupt disable 1: Sequence A conversion complete interrupt enable	R/W

16.4.15 A/D Cooperative Mode Control Register ADC_SYNCRR

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	Marking	Place name	Function	R/W
b15-b8	SYNCDLY[7:0]	Synchronization delay time	In delayed trigger mode, the start-up delay time of the two ADCs is $t_{SYNCDLY}$. 0x1 means $t_{SYNCDLY} = 1 \times PCLK4$, 0xff means $t_{SYNCDLY} = 255 \times PCLK4$ Note: Set this register when SYNCEN is "0". Please do not write 0x00. According to the sampling time and conversion time of each ADC, set a reasonable delay time to avoid the error increase caused by multiple ADCs being in the sampling state at the same time, and avoid triggering again before the conversion of the ADC is completed, resulting in synchronization failure. The recommended settings are as follows: Single delayed trigger mode: $t_{SYNCDLY} > t_{SPL}$ Two ADC cycle delay trigger modes: $t_{SYNCDLY} > t_{SPL}$, 且 $t_{SYNCDLY} > t_{SCAN}/2$ Three ADC cycle delay trigger modes: $t_{SYNCDLY} > t_{SPL}$, 且 $t_{SYNCDLY} > t_{SCAN}/3$ Single parallel trigger mode: the setting of this register is invalid. Loop parallel trigger mode: $t_{SYNCDLY} > t_{SCAN}$	R/W
b7	—	—	Read at 0, write at 0	R/W
b6-b4	SYNCMD[2:0]	Synchronization mode selection	SYNCMD[2] 0: single trigger 1: Loop trigger SYNCMD[1] 0: delayed trigger mode 1: Parallel trigger mode SYNCMD[0] 0: ADC1 and ADC2 work synchronously, ADC3 works independently 1: ADC1, ADC2 and ADC3 work synchronously, if there is no ADC3, this bit must not be set to 1. Note: Set this register when SYNCEN is "0". When using a single trigger, please set the ADC to be synchronized to Sequence A single scan, or Sequence A continuous scan mode. When using loop trigger mode, set the ADC to Sequence A single-sweep mode.	R/W
b3-b1	—	—	Read at 0, write at 0	R/W
b0	SYNCEN	Synchronous Mode License	0: Synchronous mode is invalid 1: Synchronous mode is valid Note: Synchronous mode only supports sequence A. Before SYNCEN writes 1, please turn off sequence B of several ADCs involved in synchronization (ADC_CRO.MS[1]=0), and select the same number of channels for sequence A, and set the same channel sampling time ADC_SSTRx. In order to avoid the inconsistent scanning time t_{SCAN} of each ADC, causing subsequent synchronization failures. When the software writes 0 to ADC1_STR.START to force the conversion to stop, SYNCEN is automatically cleared to 0.	R/W

Note:

- This register is only carried in ADC1, and there is no such register in ADC2.

16.4.16 A/D Data Register (ADC_DR)

ADC_DRx (ADC1 x=0~16, ADC2 x=0~8) channel x data register

ADC_DRregister 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 of data - 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 of data - 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 of data - 8-bit resolution

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

Data Left Alignment - 12-bit Resolution

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
													0	0	0

Data Left Alignment - 10-bit Resolution

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
												0	0	0	0

Data 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

16.4.17 Analog Watchdog Control Register (ADC_AWDCR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
—	—	—	—	—	—	—	AWDIEN
b7	b6	b5	b4	b3	b2	b1	b0
AWDSS[1]	AWDSS[0]	—	AWDMD	—	—	—	AWDEN

Bit	Marking	Place name	Function	R/W
b15-b9	—	—	Read at 0, write at 0	R/W
b8	AWDIEN	Watchdog interrupt enable	0: Disable ADC_CHCMP, ADC_SEQCMP interrupt 1: Enable ADC_CHCMP, ADC_SEQCMP interrupt Note: This register does not affect ADC_CHCMP, ADC_SEQCMP event output	R/W
b7-b6	AWDSS[1:0]	Watchdog Sequence Selection	00: ADC_SEQCMP interrupt/event is output when sequence A and sequence B scans are completed 01: Output ADC_SEQCMP interrupt/event when sequence A scan is completed, sequence B does not output 10: Output ADC_SEQCMP interrupt/event when sequence B scan is completed, sequence A does not output 11: Same as 00 Note: The channel watchdog interrupt/event ADC_CHCMP is not controlled by this register, and is normally output according to the comparison result at the end of each channel conversion. The setting of the comparison status register of each channel of ADC_AWDSR is not controlled by this register.	R/W
b5	—	—	Read at 0, write at 0	R/W
b4	AWDMD	Watchdog Compare Mode	0: When AWDDR0 > conversion result, or conversion result > AWDDR1, the comparison condition is satisfied 1: When the conversion result of AWDDR0 to Q is to QA_WDDR1, the comparison condition is satisfied When the comparison condition is satisfied, the ADC_CHCMP event is output, and if the interrupt is enabled AWDIEN=1, an interrupt is output at the same time. When the sequence scan is completed, if one or more channels in this sequence meet the comparison conditions, and AWDSS[1:0] allows this sequence, the ADC_SEQCMP event will be output, and if the interrupt is enabled AWDIEN=1, an interrupt will be output at the same time.	R/W
b3-b1	—	—	Read at 0, write at 0	R/W
b0	AWDEN	Watchdog compare function enable	0: Watchdog comparison function is invalid 1: The watchdog comparison function is valid	R/W

16.4.18 Analog Watchdog Threshold Register ADC_AWDDR0, ADC_AWDDR1

Reset value: ADC_AWDDR0=0x0000, ADC_AWDDR1=0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8
AWDDR[15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
AWDDR[7:0]							
Bit	Marking	Place name	Function		R/W		
b15-b0	ADDR[15:0]	Comparative data	Comparative data		R/W		

AWDDR0 sets the low threshold, AWDDR1 sets the high threshold, and data can also be written into the register during A/D conversion to realize the dynamic threshold comparison function.

AWDDR0 and AWDDR1 vary according to the alignment (data right or left), and the resolution (12-bit, 10-bit or 8-bit).

- Right alignment of data - 12 bit resolution low 12 bits [11: 0] available
- Right alignment of data - 10 bit resolution low 10 bits [9: 0] available
- Right Alignment of Data - 8-bit Resolution Low 8-bit [7: 0] Available
- Data left alignment - 12 bit resolution high 12 bits [15: 4] available
- Data left alignment - 10 bit resolution high 10 bits [15: 6] available
- Data left alignment - 8 bit resolution high 8 bits available [15: 8]

When the multiple averaging function is valid, only the last average value is compared.

16.4.19 Analog Watchdog Compare Channel Select Register ADC_AWDCHSR0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AWDCH[15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
AWDCH[7:0]							

Bit	Marking	Place name	Function	R/W
b15-b0	AWDCH[15:0]	Watchdog compare function channel selection	0: The watchdog comparison function of this channel is not selected 1: The channel watchdog comparison function selection Each bit corresponds to a channel. The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. Note: It is valid only when the corresponding channel is selected in the sequence A or B scan, that is, the corresponding bit of the channel selection register ADC_CHSELRA or ADC_CHSELRB is "1".	R/W

Note:

- Please set this register when ADC_STR.STRT is "0" and ADC_AWDSCR.AWDEN is "0".

16.4.20 Analog Watchdog Compare Channel Select Register 1 ADC_AWDCHSR1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AWDCH[31:24]							
b7	b6	b5	b4	b3	b2	b1	b0
AWDCH[23:16]							

Bit	Marking	Place name	Function	R/W
b15-b0	AWDCH[31:16]	Watchdog compare function channel selection	0: The watchdog comparison function of this channel is not selected 1: The channel watchdog comparison function selection Each bit corresponds to a channel. The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. Note: It is valid only when the corresponding channel is selected in the sequence A or B scan, that is, the corresponding bit of the channel selection register ADC_CHSELRA or ADC_CHSELRB is "1".	R/W

Note:

- Please set this register when ADC_STR.STRT is "0" and ADC_AWDSCR.AWDEN is "0".

16.4.21 Analog Watchdog Status Register ADC_AWDSR0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AWDF[15:8]							
AWDF[7:0]							
Bit	Marking	Place name	Function				R/W
b15-b0	AWDF[15:0]	Watchdog compare status bit	0: The comparison condition is not established 1: The comparison condition is established Note: Each bit corresponds to a channel. The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. When the register is set and needs to be cleared, please write "0" after reading "1".				R/W

16.4.22 Analog Watchdog Status Register 1 ADC_AWDSR1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AWDF[31:24]							
AWDF[23:16]							
Bit	Marking	Place name	Function				R/W
b15-b0	AWDF[31:16]	Watchdog compare status bit	0: The comparison condition is not established 1: The comparison condition is established Note: Each bit corresponds to a channel. The corresponding bits of the non-existent channel are reserved bits, 0 at read and 0 at write. When the register is set and needs to be cleared, please write "0" after reading "1".				R/W

16.4.23 A/D Programmable Gain Amplifier Control Register ADC_PGACR

Reset value: 0x0000

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

Bit	Marking	Place name	Function	R/W
b15-b12	—	—	Read at 0, write at 0	R/W
b3~b0	PGACTL[3:0]	Amplifier control	0000: Amplifier invalid 1110: The amplifier is valid, and the signal is amplified according to the setting value of ADC_PGAGSR.GAIN[3:0] Note: Setting values other than the above is prohibited.	R/W

16.4.24 A/D Programmable Gain Multiplier Register ADC_PGAGSR

Reset value: 0x0000

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

Bit	Marking	Place name	Function	R/W
b15-b4	—	—	Read at 0, write at 0	R/W
b3-b0	GAIN[3:0]	Amplifier 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 to be set.	R/W

16.4.25 A/D Programmable Gain Amplifier Input Selection Register ADC_PGAINSR0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
—	—	—	—	—	—	—	PGAINSEL[8]
b7	b6	b5	b4	b3	b2	b1	b0
PGAINSEL[7:0]							

Bit	Marking	Place name	Function	R/W
b15-b9	—	—	Read at 0, write at 0	R/W
b8~b0	PGAINSEL[8:0]	Amplifier Analog Input Selection	0x000: input not selected 0x001: ADC1_IN0 0x002: ADC1_IN1 0x004: ADC1_IN2 0x008: ADC1_IN3 0x010: ADC1_IN4 0x020: ADC1_IN5 0x040: ADC1_IN6 0x080: ADC1_IN7 0x100: Internal analog channel (8bitDAC_1 output) Others: Prohibited settings	R/W

16.4.26 A/D Programmable Gain Amplifier Input Selection Register 1 ADC_PGAINSR1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
—	—	—	—	—	—	—	—
b7	b6	b5	b4	b3	b2	b1	b0
PGAVSSEN							

Bit	Marking	Place name	Function	R/W
b15-b1	—	—	Read at 0, write at 0	R/W
b0	PGAVSSEN	Amplifier Ground Cut Control Selection	0: Use external port PGAVSS as PGA negative input 1: Use the internal analog ground AVSS as the PGA negative input	R/W

16.5 Precautions for Use

16.5.1 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.

16.5.2 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.

16.5.3 Precautions for Module Stop and Low Power Consumption Setting

By setting the register PWC_FCG3, you can set the ADC module to stop and reduce power consumption. The initial state of ADC is stop state. When the A/D module needs to work, please set the corresponding bit of the PWC_FCG3 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 the conversion stop, that is, the ADC_STR.START bit is "0".

Before setting the system to enter the stop mode (STOP), please set the ADC to the module stop mode first.

For details, please refer to the Low Power Description section.

16.5.4 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.

16.5.5 Noise Control

In order to prevent abnormal voltages such as surges from damaging analog input pins, it is recommended to use the protection circuit shown in the electrical characteristics chapter of the Data Manual.

17 Temperature Sensor (OTS)

17.1 Introduction

The temperature sensor (On-chip Temperature Sensor, hereinafter referred to as OTS) can obtain the temperature inside the chip to support the reliable operation of the system. OTS provides a set of temperature-related digital quantities, and the temperature value can be obtained through calculation. When not in use, it can be turned off by the module stop function to reduce system power consumption.

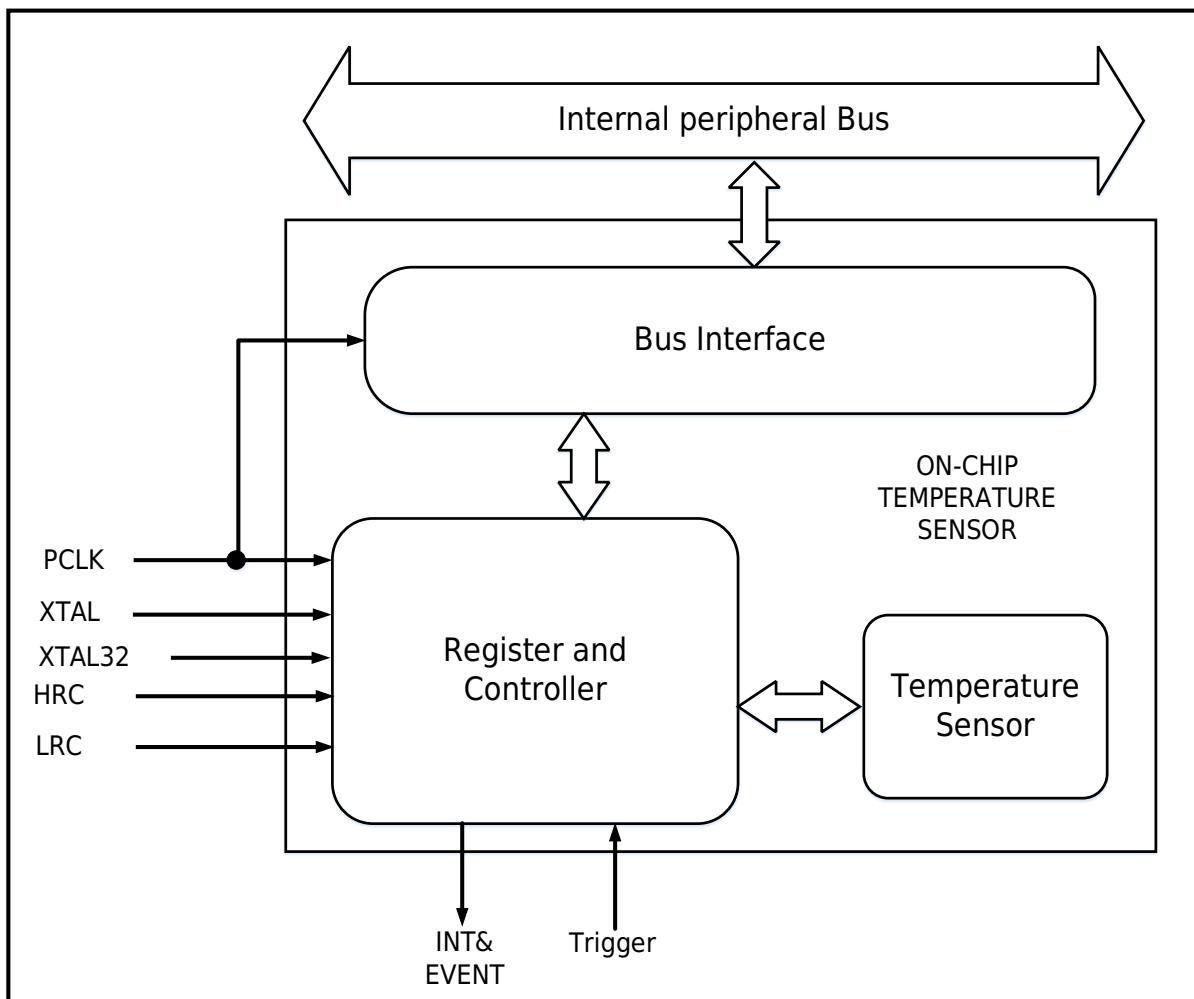


Figure 17-1 OTS functional block diagram

17.2 Usage Notes

Before using OTS to obtain the internal temperature of the chip, 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. Therefore, after confirming that OTSST is "0", read the temperature parameters in the registers OTS_DR1, 2 and OTS_ECR, and use the following calculation formula to obtain the temperature value.

$$T = K \times (1/D1 - 1/D2) \times Ehrc + M$$

[Parameter Description]

T: temperature ($^{\circ}$ C)

K: temperature slope (determined by calibration experiment)

D1: Temperature parameter 1 (read from register OTS_DR1)

D2: Temperature parameter 2 (read from register OTS_DR2)

Ehrc: HRC frequency error compensation amount (read from register OTS_ECR)

M: Temperature offset (determined by calibration experiment)

[Calibration experiment]

Calibration experiments were carried out at two defined temperatures to calculate K and M.

$$K = (T2 - T1) / (A2 - A1)$$

$$M = T1 - K \times A1 = T2 - K \times A2$$

T1: experimental temperature 1

T2: experimental temperature 2

$$A1: (1/D1_{T1} - 1/D2_{T1}) \times Ehrc_{T1}$$

D1_{T1}, D2_{T1}, Ehrc_{T1} are D1, D2, Ehrc measured at temperature T1 respectively;

$$A2: (1/D1_{T2} - 1/D2_{T2}) \times Ehrc_{T2}$$

D1_{T2}, D2_{T2}, Ehrc_{T2} are D1, D2, Ehrc measured at temperature T2 respectively;

The register OTS_CTL.OTSCK is used to select the temperature measurement clock. When the HRC action is selected, its frequency error may affect the accuracy of the final calculated temperature. In order to eliminate this error, please start XTAL32 before temperature measurement and use Ehrc when calculating temperature. Show Ehrc as constant 1 when selecting XTAL action clock.

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 is turned on after a temperature measurement is completed, so that the stabilization time of the analog temperature sensor from off to on will be automatically skipped in the next temperature measurement. To turn off the analog temperature sensor after each temperature measurement, please set TSSTP to "1".

The temperature measurement can be triggered by other peripheral events. When using it, please set the trigger target of the trigger source to OTS. When the temperature measurement is completed, an event that triggers the start of other peripherals will also be generated. When using it, please set the register OTS_TRG to select the trigger target. Please set the register OTS_CTL.OTSIE to "1" when using the temperature measurement completion interrupt.

17.3 Register Description

Reference address: 0x4004_A400

Table 17-1 List of OTS Registers

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

17.3.1 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
b3	TSSTP	Turn off the analog temperature sensor	Select whether to automatically turn off the analog temperature sensor after the temperature measurement is completed 0: Do not turn off the analog temperature sensor 1: Turn off the analog temperature sensor	R/W
b2	OTSIIE	Interrupt enable bit	0: Disable temperature measurement end interrupt request 1: Allow temperature measurement end interrupt request	R/W
b1	OTSCK	clock select bit	0: Select external high-speed clock (XTAL) action 1: Select Internal high-speed clock (HRC) action	R/W
b0	OTSST	Temperature measurement start bit	0: stop temperature measurement 1: start temperature measurement Set the "1" condition: (1) Software set "1" (2) Hardware trigger set to "1" Clear "0" condition: (1) Software "0" (2) Automatically clear "0" after temperature measurement	R/W

17.3.2 OTS Data Register 1 (OTS_DR1)

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

Reset value: 0x0000

Bit	Marking	Place name	Function	Read and write
b15~b0	TSDC[15:0]	Temperature data 1	Temperature data D1 It will be automatically updated after the temperature measurement is completed. Please confirm that OTS_CTL.OTSST is "0" before reading.	R

17.3.3 OTS Data Register 2 (OTS_DR2)

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

Reset value: 0x0000

Bit	Marking	Place name	Function	Read and write
b15~b0	TSDC[15:0]	Temperature data 2	Temperature data D2 It will be automatically updated after the temperature measurement is completed. Please confirm that OTS_CTL.OTSST is "0" before reading.	R

17.3.4 OTS Error Compensation Register (OTS_ECR)

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

Reset value: 0x0000

Bit	Marking	Place name	Function	Read and write
b15~b0	TSEC[15:0]	Error factor	Error coefficient Ehrc It will be automatically updated after the temperature measurement is completed. Please confirm that OTS_CTL.OTSST is "0" before reading.	R

18 Advanced Control Timer (Timer6)

18.1 Introduction

Advanced Control Timer 6 (Timer6) is a high-performance timer with a 16-bit count width, which can be used to count and generate different forms of clock waveforms, and output them for external use. The timer supports two waveform modes of triangle wave and sawtooth wave, and can generate various PWM waveforms; software synchronous counting and hardware synchronous counting can be realized between units; each reference value register supports cache function; supports 2-phase quadrature encoding and 3-phase quadrature Cross coding; support EMB control. This series of products is equipped with 3 units of Timer6.

18.2 Basic Block Diagram

The basic functions and characteristics of Timer6 are shown in Table Table 18-1.

Table 18-1 Basic functions and features of Timer6

Waveform mode	Sawtooth wave (counting up and down), triangle wave (counting up and down)
Basic functions	• Capture input
	• Software synchronization
	• Hardware synchronization
	• Cache function
	• Pulse width measurement
	• Periodic measurement
	• Orthogonal coding count
	• General PWM output
	• EMB control
Interrupt output	Count comparison match interrupt
	Count cycle match interrupt
	Dead time error interrupt
Event output	Count compare match events
	Count period match event

The basic block diagram of Timer6 is shown in Figure 18-1, "<t>" represents the unit number, that is, "<t>" is 1~3. This section refers to "<t>" when referring to "<t>" later.

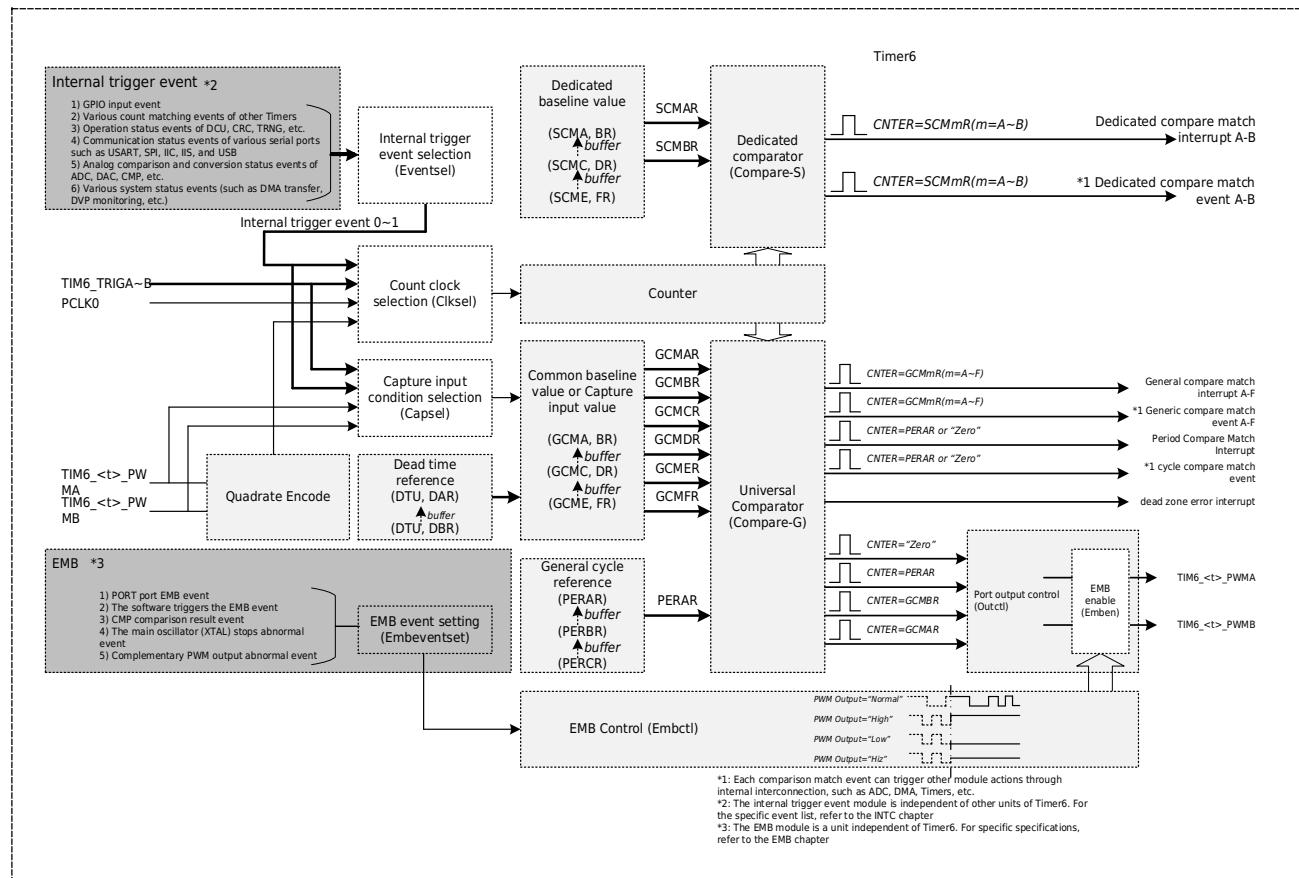


Figure 18-1 Timer6 basic block diagram

The list of input and output ports of Timer6 is shown in Table 18-2.

Table 18-2 Timer6 port list

Port name	Direction	Function
TIM6_<t>_PWMA	in or out	1) Quadrature encoding count clock input port or capture input port or comparison output port 2) Hardware start, stop, clear condition input port
TIM6_<t>_PWMB		
TIM6_TRIGA	in	1) Hardware count clock input port or capture input port 2) Hardware start, stop, clear condition input port
TIM6_TRIGB		

18.3 Functional Description

18.3.1 Basic Action

18.3.1.1 Waveform Mode

Timer6 has two basic counting wave modes: Serrated wave mode and triangular wave mode. The triangular waveform mode is divided into triangular wave A mode and triangular wave B mode due to different internal counting actions. The basic waveforms of sawtooth and triangle waves are as Figure 18-2 Figure 18-3 shown.

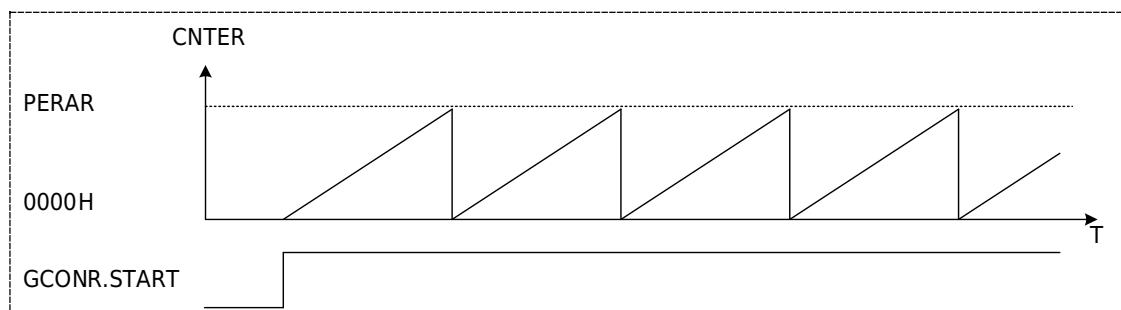


Figure 18-2 Sawtooth Waveform (Counting Up)

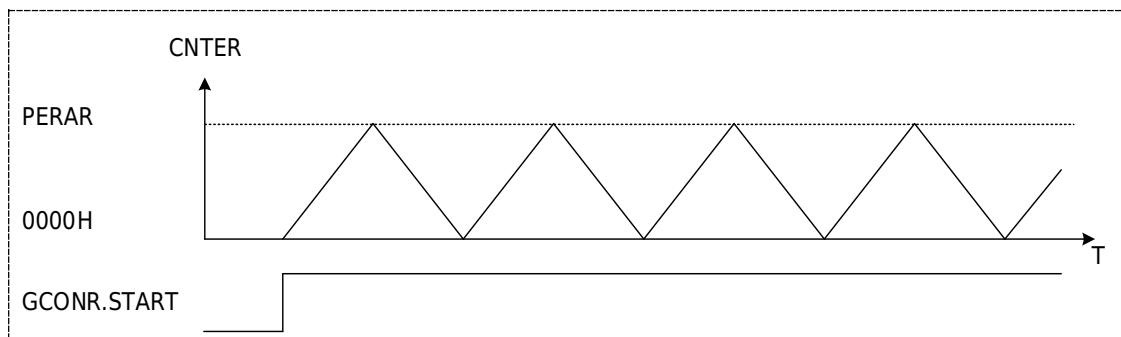


Figure 18-3 Triangle Waveform

18.3.1.2 Comparison Output

Timer6 of each unit has 2 comparison output ports (TIM6_<t>_PWMA, TIM6_<t>_PWMB), which can output a specified level when the count value matches the comparison reference value. The GCMAR and GCMBR registers correspond to the count comparison reference values of TIM6_<t>_PWMA and TIM6_<t>_PWMB respectively. When the count value of the timer is equal to GCMAR, the TIM6_<t>_PWMA port outputs the specified level; when the count value of the timer is equal to GCMBR, the TIM6_<t>_PWMB port outputs the specified level.

TIM6_<t>_PWMA, TIM6_<t>_PWMB port's counting start level, counting stop level, counting comparison matching level, counting cycle matching level, etc., can be controlled by the port control register (PCONR) PCONR.STACA, PCONR.STPCA, PCONR.STATPSA, PCONR.CMPCA[1:0], PCONR.PERCA[1:0] bit settings. Figure 18-4 This is an example of the operation of the comparison output.

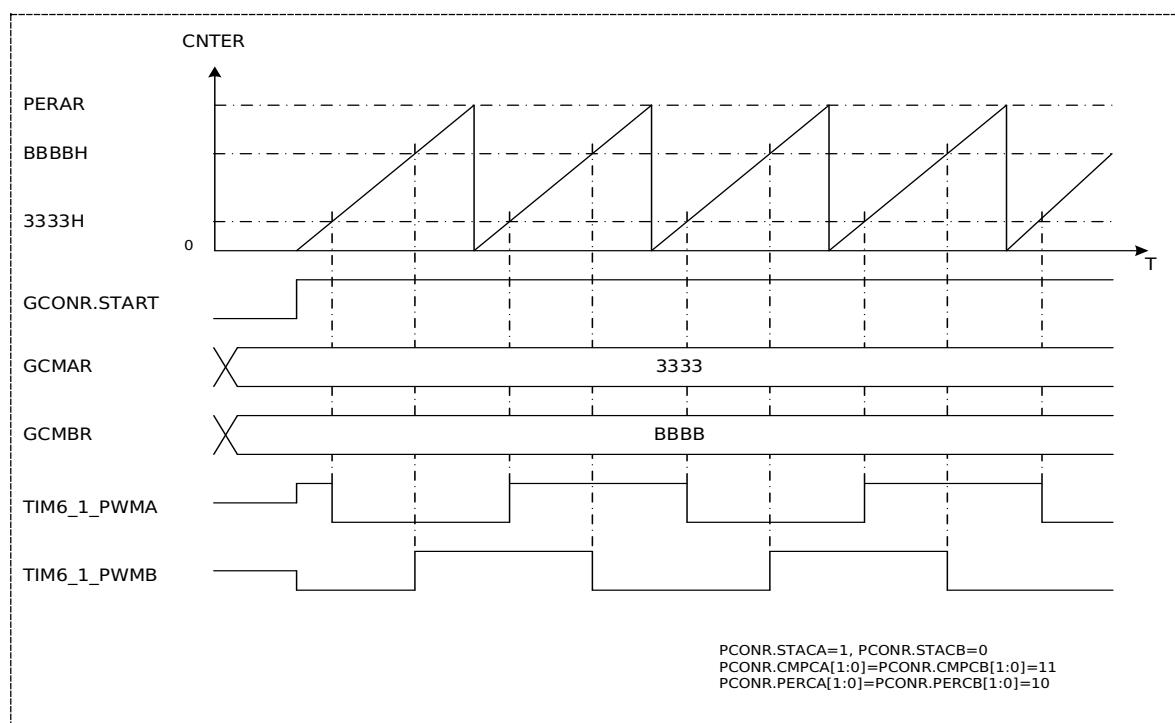


Figure 18-4 Compare Output Action

18.3.1.3 Capture Input

Each unit has a capture input function, with 2 sets of capture input registers (GCMAR, GCMBR), used to save the captured count value. Set the PCONR.CAPMDA and PCONR.CAPMDB bits of the port control register (PCONR) to 1, and the capture input function becomes valid. When the corresponding capture input condition is set and the condition is valid, the current count value is saved to the corresponding capture register (GCMAR, GCMBR).

The condition of each capture input of each unit can be internal trigger event input, TIM6_TRIGA or TIM6_TRIGB port input, TIM6_<t>_PWMA or TIM6_<t>_PWMB port input, etc. The specific conditions can be selected through the hardware capture event selection register (HCPAR, HCPBR) to set.

Figure 18-5 An example of an action for capturing input.

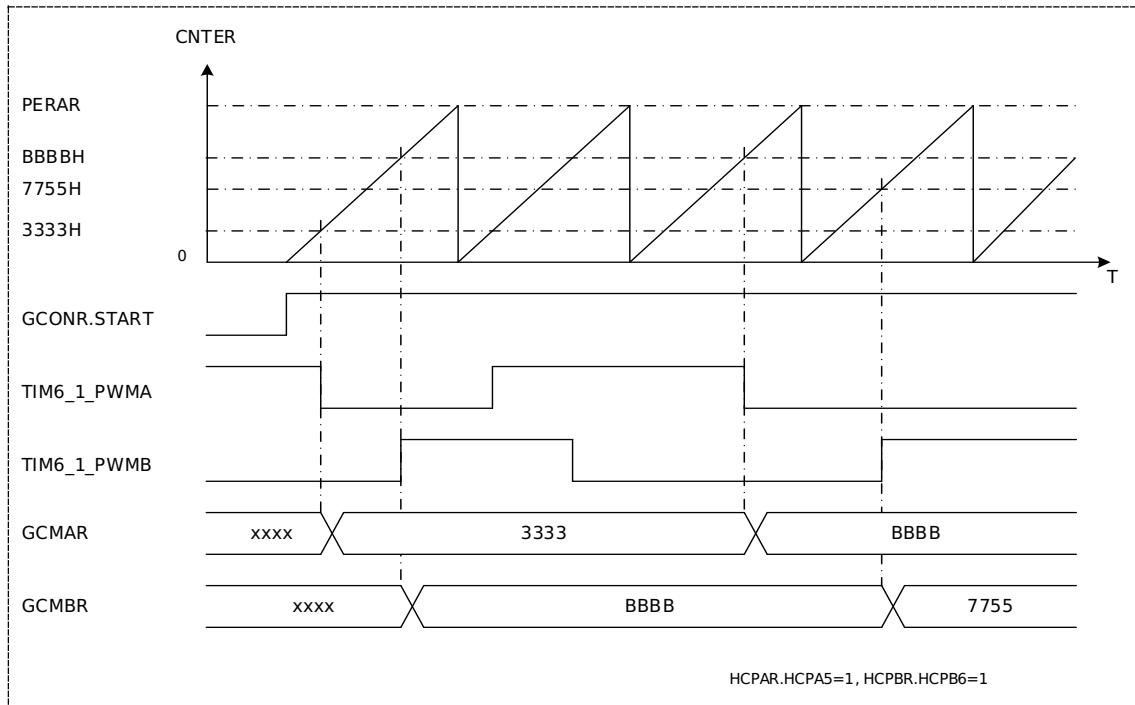


Figure 18-5 Capturing Input Actions

18.3.2 Timer Selection

The counting clock of Timer6 can have the following options:

- a) PCLK0 and 2, 4, 8, 16, 64, 256, 1024 frequency division of PCLK0 (GCONR.CKDIV[2:0] setting)
- b) Internal trigger event trigger input (HCUPR[17:16] or HCDOR[17:16] setting)
- c) Port input of TIM6_TRIGA-B (HCUPR[11:8] or HCDOR[11:8] setting)
- d) Quadrature code input for TIM6_<t>_PWMA and TIM6_<t>_PWMB (HCUPR[7:0] or HCDOR[7:0] setting)

The counting timer is a software counting mode, and the counting timer is b, c, and d hardware counting mode. As can be refer ton from the above description, the b, c and d clocks are independent of each other, can be set to be valid or invalid respectively, and a clock is automatically invalid when b, c and d clocks are selected.

18.3.3 Counting Direction

The timer counting direction of Timer6 can be changed by software. The method of changing the counting direction is slightly different in different waveform modes.

18.3.3.1 Sawtooth Counting Direction

In sawtooth wave mode, the counting direction can be set while the timer is counting or stopped.

When counting up, set GCONR.DIR=0 (count down), then the timer will change to down count mode after counting to overflow; when counting down, set GCONR.DIR=1 (count up), the timer will change to up counting mode after counting to underflow.

The GCONR.DIR bit is set when counting is stopped, and the setting of GCONR.DIR is reflected in the counting until it overflows or underflows after the counting starts.

18.3.3.2 Triangular Wave Counting Direction

In the triangular wave mode, the set counting direction is invalid, and the counting direction will be changed automatically when counting to the counting peak point or counting valley point.

18.3.4 Digital Filtering

The TIM6_<t>_PWMA, TIM6_<t>_PWMB, TIM6_TRIGA~B port inputs of Timer6 all have digital filter function. The filter function of the corresponding port can be enabled by setting the relevant enable bit of the filter control register (FCONR). The filter reference clock when the filter is valid can also be set by the filter control register (FCONR).

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. Its action is shown in Figure 18-6 the example.

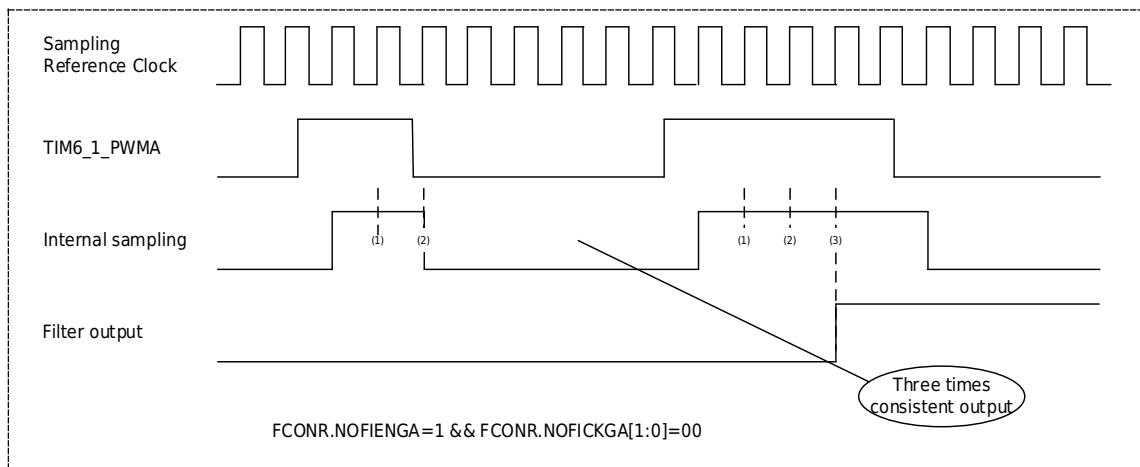


Figure 18-6 Filtering Function of Capture Input Port

The TIM6_TRIGA~B port is a port shared by a group of units. The digital filter function of this group of ports is set by the FCONR of unit 1, and the FCONR of other units is invalid for the digital filter function setting of this group of ports. When any unit uses the digital filtering function, it is necessary to clear the TIMER6_1 bit in the function controller (PWC_FCG2).

18.3.5 Software Synchronization

18.3.5.1 Software Synchronization Start

Each unit can realize the synchronous start of the target unit by setting the relevant bits of the software synchronous start control register (SSTAR).

18.3.5.2 Software Co-stop

Each unit can realize the synchronous stop of the target unit by setting the relevant bits of the software synchronous stop control register (SSTPR).

18.3.5.3 Software Synchronous Clear

Each unit can realize the synchronous clearing of the target unit by setting the relevant bits of the software synchronous clearing control register (SCLRR).

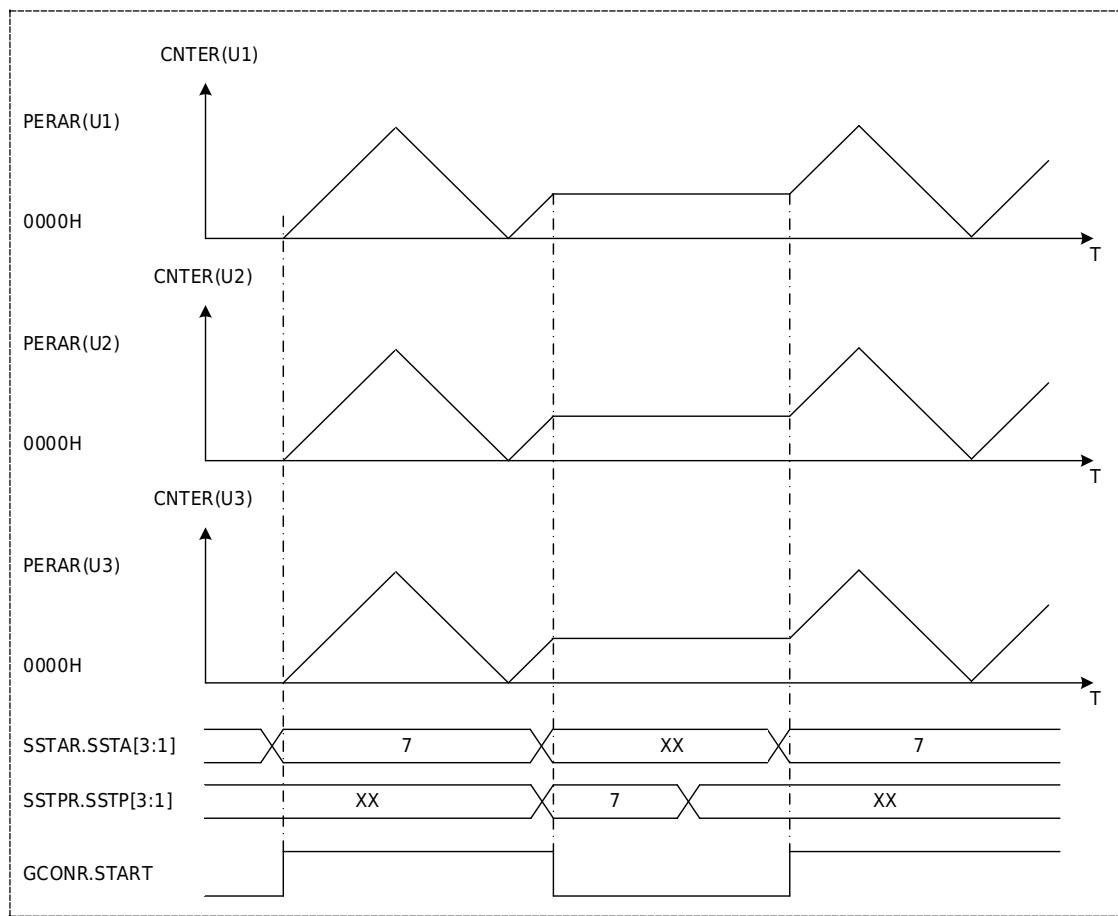


Figure 18-7 Software Synchronization Action

As Figure 18-7 shown, if $SSTAR.SSTA1=SSTAR.SSTA2=SSTAR.SSTA3=1$ is set, the software of units 1~3 can be started synchronously.

Software synchronous action related registers (SSTAR, SSTPR, SCLRR) are a group of registers that are independent of the unit and shared between each unit. Each bit of this group of registers is only

valid when writing 1, and writing 0 is invalid. When reading the SSTAR register, the timer status of each unit will be read, and when reading SSTPR or SCLRR, 0 will be read.

18.3.6 Hardware Synchronization

In addition to independently having 2 general-purpose input ports (TIM6_<t>_PWMA, TIM6_<t>_PWMB), each unit also has 2 external general-purpose input ports (TIM6_TRIGA, TIM6_TRIGB) and 2 internal trigger event input conditions. Realize hardware synchronization between units.

The event source of the internal hardware trigger event can be selected by setting the corresponding number in the hardware trigger event selection register (HTSSR0~1). For the specific event correspondence, please refer to [Interrupt Controller (INTC)] the chapter. When using the internal hardware trigger function, the function clock control register (PWC_FCG0) peripheral circuit trigger function is required to enable position 1.

18.3.6.1 Hardware Sync 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 selection register (HSTAR).

18.3.6.2 Hardware Co-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 selection register (HSTPR).

18.3.6.3 Hardware Synchronous Clear

Each unit can choose to clear the timer by hardware, and select the units with the same hardware clearing condition to realize synchronous clearing when the clearing condition is valid. The specific hardware clear condition is determined by the setting of the hardware clear event select register (HCLR).

18.3.6.4 Hardware Synchronous Capture Input

Each unit can choose to use hardware to realize the capture input function, and select the unit with the same capture input function condition to realize synchronous capture input when the capture input function condition is valid. The specific hardware capture input function conditions are determined by the settings of the hardware capture event selection registers (HCPAR, HCPBR).

18.3.6.5 Hardware Sync-count

Each unit can choose to use the hardware input as CLOCK to count, and the units with the same hardware counting conditions can realize synchronous counting when the hardware counting clock is valid. The specific hardware counting conditions are determined by the settings of the hardware

increment event selection register (HCUPR) and the hardware decrement event selection register (HCDOR).

Figure 18-8 Shown is an example of hardware synchronous operation of units 1 to 3.

When the hardware synchronous counting function is selected, only the external input clock source is selected, which does not affect the start, stop, and reset actions of the timer. The start, stop, and reset of the timer also need to be set separately.

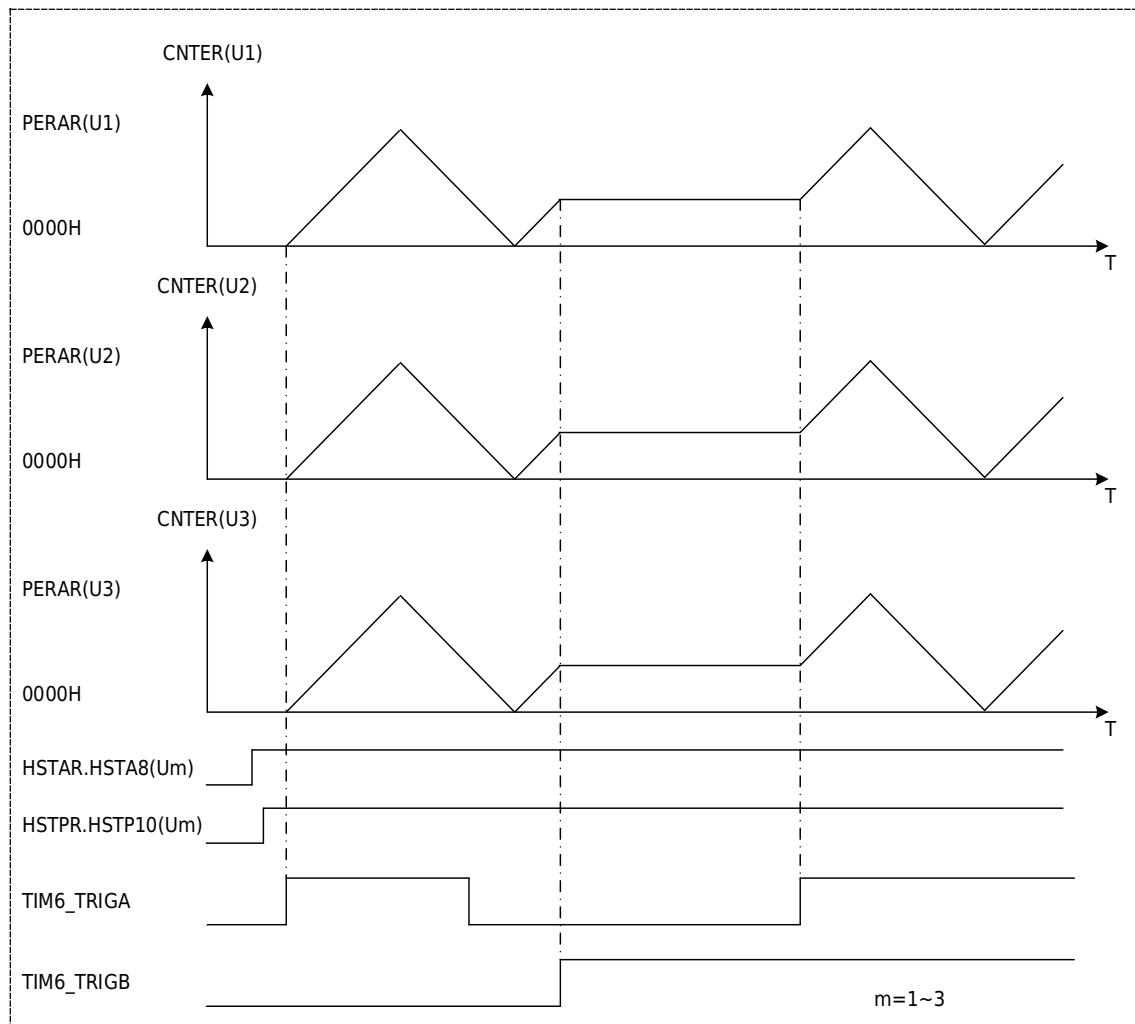


Figure 18-8 Hardware Synchronous Action

18.3.7 Pulse Width Measurement

When using the hardware trigger related functions of the TIM6_<t>_TRIGA~B port (refer to the hardware synchronization chapter), each unit can realize 2 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, and the hardware clear condition, stop condition, and capture input condition of the GCMAR register are all set to the falling edge of TIM6_<t>_TRIGA, continuous Pulse width measurement.

18.3.8 Periodic Measurement

When using the hardware trigger-related functions of the TIM6_<t>_TRIGA~B port (refer to the hardware synchronization chapter), each unit can implement 2-way independent period measurement functions.

For example, setting the hardware start condition of the counter, the hardware clear condition, and the capture input condition of the GCMR register to the rising edge of TIM6_<t>_TRIGB can realize continuous period measurement.

18.3.9 Cache Function

The Counting Period Value, General Comparison Reference Value, Special Comparison Reference Value, and DEAD ZONE TIME SETTING VALUE of Timer6 All Have A Cache Function, Which Can Realize Cycle Change, Duty Cycle Change, Dead Zone Change, ETC. During The Counting Period. The counting cycle value, general comparison reference value and special comparison reference value have single-buffer and double-buffer functions, and the dead zone time setting value has single-buffer function.

18.3.9.1 Single Cache Action

Single buffer action means that by setting the buffer control register (BCONR) and the dead zone control register (DCONR), at the time point of buffer transmission, the following events are selected:

- a) The value of the general period reference buffer register (PERBR) is automatically transferred to the general period reference register (PERAR)
- b) The value of the general comparison reference value buffer register (GCMCR, GCMDR) is automatically transferred to the general comparison reference value register (GCMAR, GCMBR) (when comparing output)
- c) The value of the general comparison reference value register (GCMAR, GCMBR) is automatically transferred to the general comparison reference value buffer register (GCMCR, GCMDR) (when capturing input)
- d) The value of the dedicated comparison reference value buffer register (SCMCR, SCMDR) is

automatically transferred to the dedicated comparison reference value register (SCMAR, SCMBR)

- e) The value of the dead time reference buffer register (DTUBR, DTDBR) is automatically transferred to the dead time reference register (DTUAR, DTDAR)

As shown in Figure 18-9, it is a timing chart of the single-buffer method of the general-purpose comparison reference value register during the comparison output operation of Unit 1. It can be seen from the figure that changing the value of the general comparison reference value register (GCMAR) during counting can adjust the output duty cycle, and changing the value of the general period reference value register (PERAR) can adjust the output cycle.

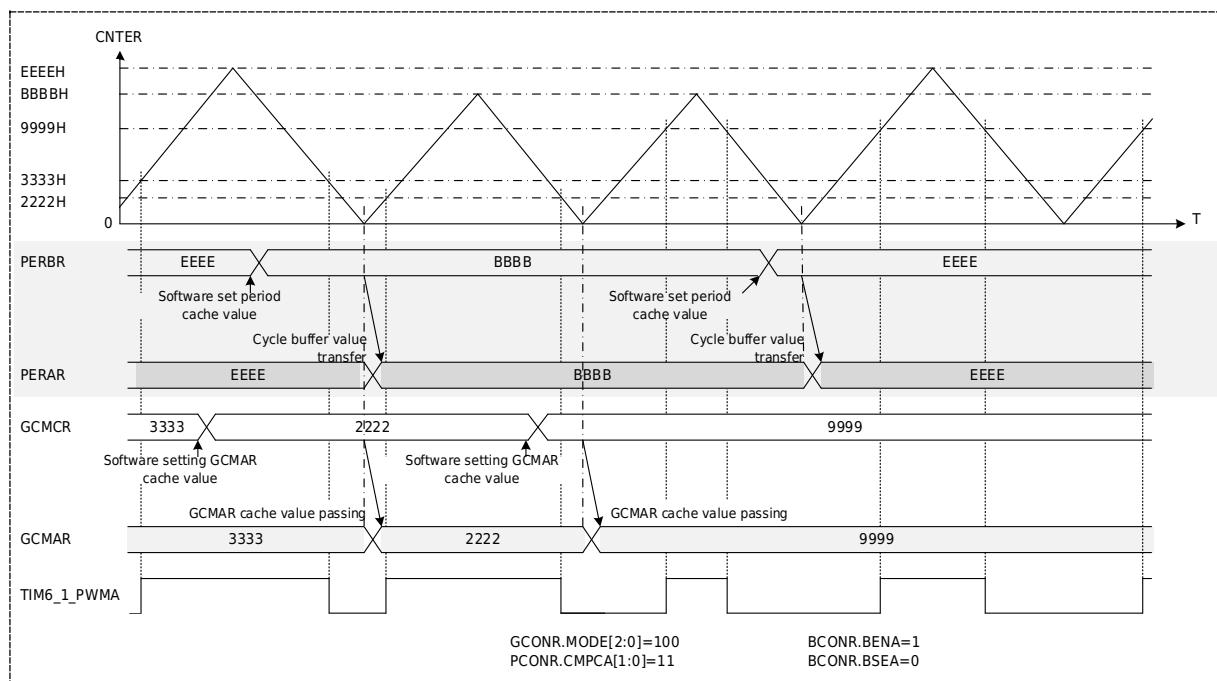


Figure 18-9 Single-buffer mode comparison output timing

18.3.9.2 Double Buffering Action

The double buffering action refers to selecting the following events to occur at the buffer transfer time point by setting the buffer control register (BCONR):

- a) The value of the general period reference value buffer register (PERBR) is automatically transferred to the general period reference value register (PERAR), and the value of the general period reference value double buffer register (PERCR) is automatically transferred to the general period reference value buffer register (PERBR)
- b) The value of the general comparison reference value buffer register (GCMCR, GCMDR) is automatically transferred to the general comparison reference value register (GCMAR, GCMBR), and the value of the general comparison reference value double buffer register (GCMER, GCMFR) is automatically transferred to the general comparison reference value buffer register (GCMCR, GCMDR) (when comparing output)
- c) The value of the general comparison reference value buffer register (GCMCR, GCMDR) is

automatically transferred to the general comparison reference value double buffer register (GCMER, GCMFR), and the value of the general comparison reference value register (GCMAR, GCMBR) is automatically transferred to the general comparison reference value buffer In registers (GCMCR, GCMDR) (when capturing input)

- d) The value of the dedicated comparison reference value buffer register (SCMCR, SCMDR) is automatically transferred to the dedicated comparison reference value register (SCMAR, SCMBR), and the value of the dedicated comparison reference value double buffer register (SCMER, SCMFR) is automatically transferred to the dedicated comparison reference value buffer Registers (SCMCR, SCMDR) in

Figure 18-10 Shown is the timing diagram of the double buffer mode when the internal trigger event 0 triggers the capture input.

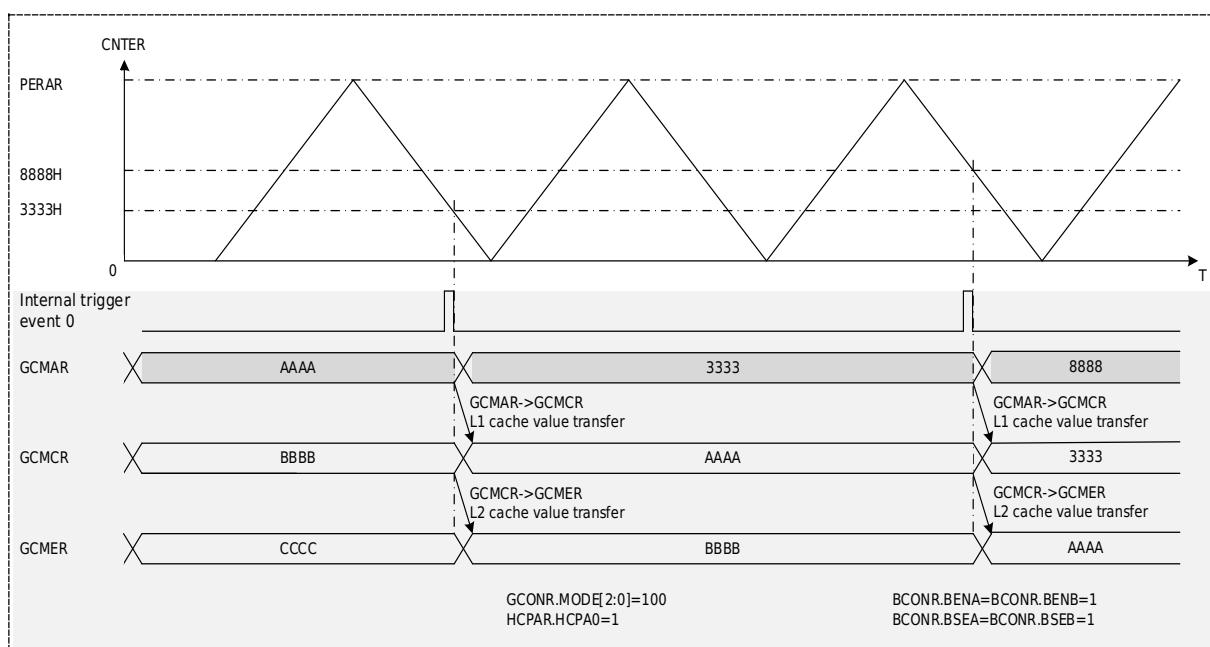


Figure 18-10 Double buffer capture input timing

18.3.9.3 Cache Delivery Time Point

General Cycle Reference Value Cache Transmission Time Point

The cycle reference value can choose single buffer function or double buffer function (BCONR.BSEP). The buffer transmission time point is the counting overflow point or the counting down point underflowing point during the sawtooth wave, and the counting valley point during the triangular wave.

Universal Baseline Value Cache Delivery Time Point

In sawtooth wave mode, set BCONR.BENA=1 or BCONR.BNEB=1, and the cache operation is valid. The caching action can be selected as a single caching function or a double caching function. Buffer transfers occur at overflow or underflow points, as Figure 18-11 indicated.

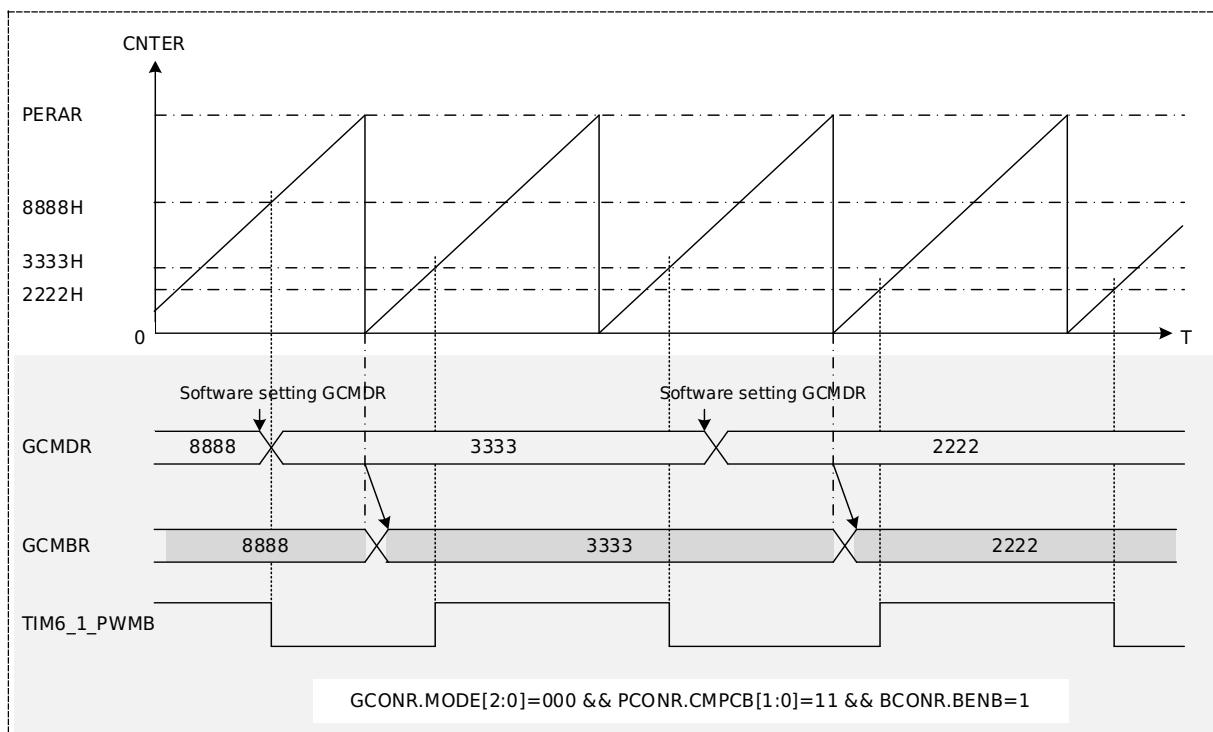


Figure 18-11 Counting Buffer Action in Sawtooth Wave Mode

In triangular wave A mode, set BCONR.BENA=1 or BCONR.BNEB=1, the cache operation is valid. The caching action can be selected as a single caching function or a double caching function. Buffer transfers occur at count valley points, as Figure 18-12 shown.

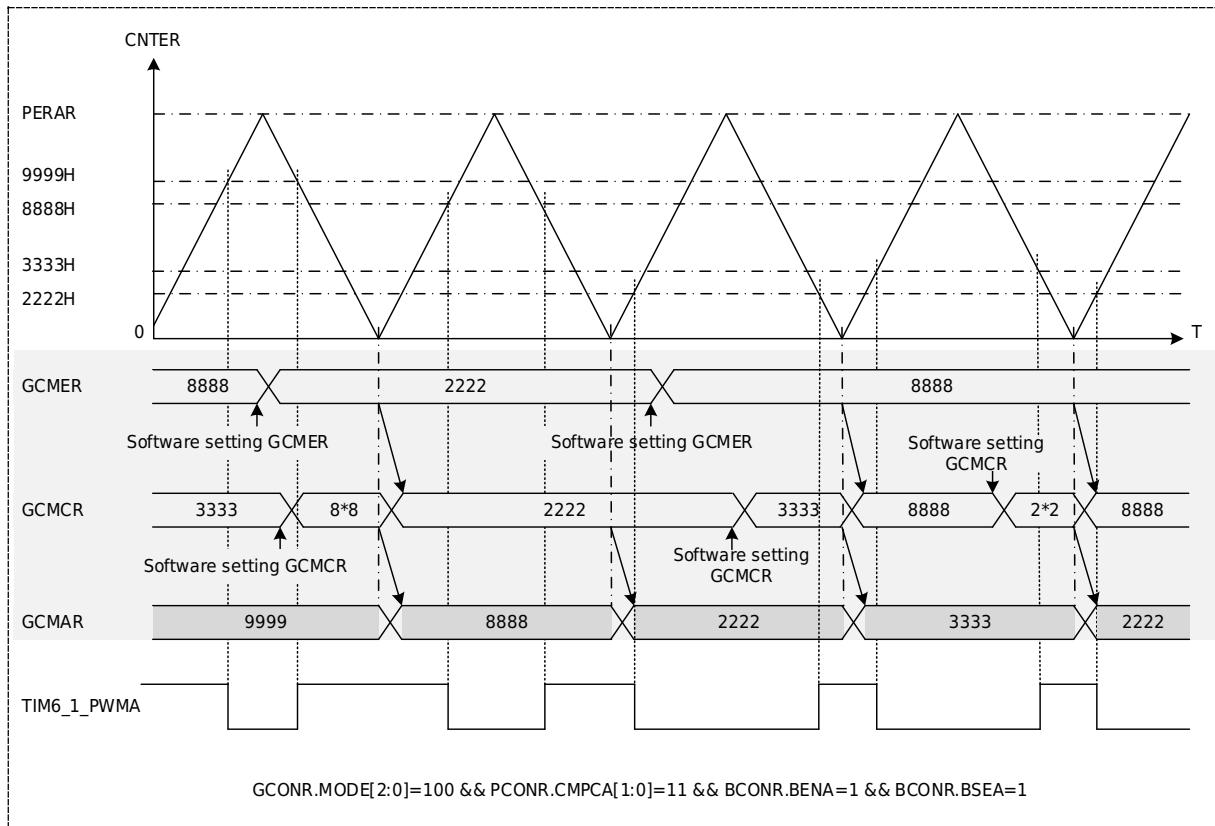


Figure 18-12 Counting Buffer Action in Triangular Wave A Mode

In triangular wave B mode, set BCONR.BENA=1 or BCONR.BNEB=1, the cache operation is valid. The caching action can be selected as a single caching function or a double caching function. Buffer transfers occur at count troughs or count peaks, as Figure 18-13 shown.

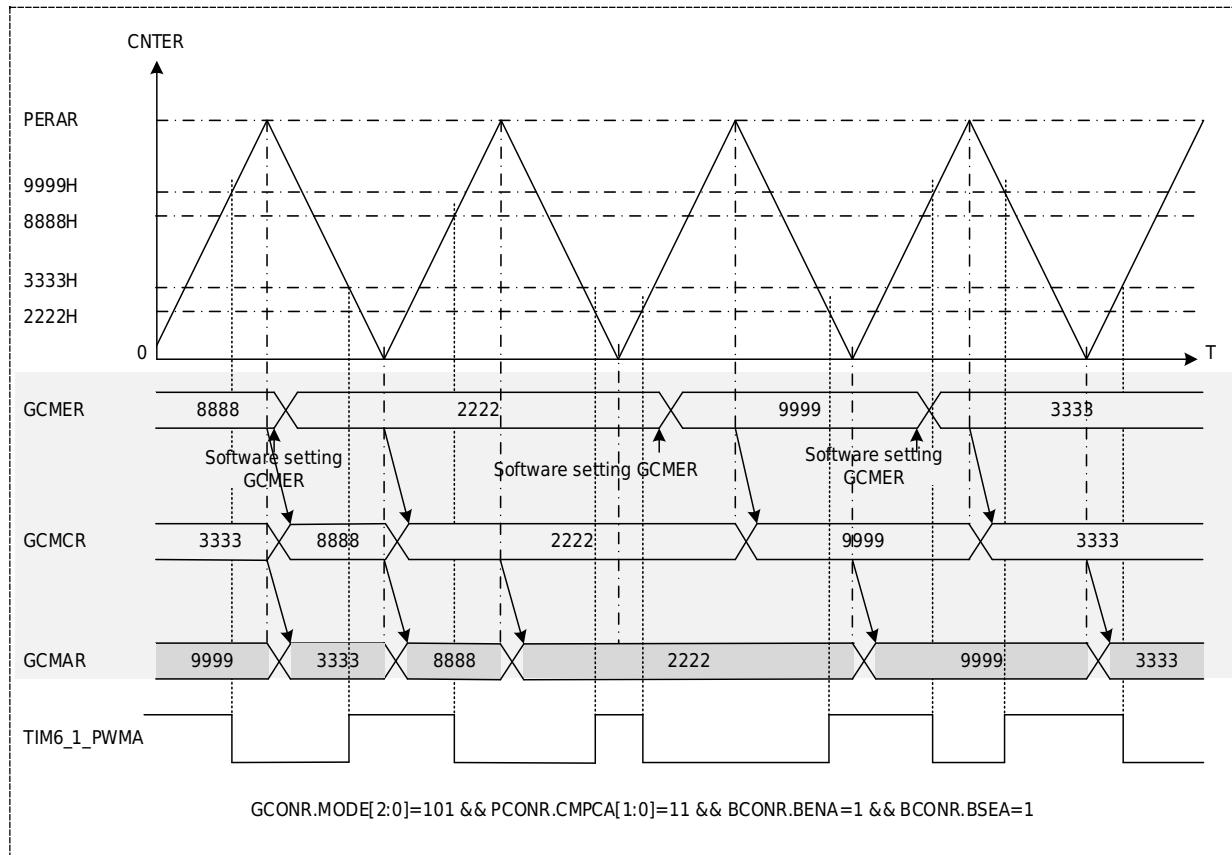


Figure 18-13 Counting buffer action in triangular wave B mode

Single-buffer transfer or double-buffer transfer is determined by BCONR.BENA, BCONR.BENB, BCONR.BSEA, and BCONR.BSEB.

Capture Input Value Buffer Transfer Time Point

Capture input action can choose single buffer function or double buffer function (BCONR.BSEA or BCONR.BSEB). The buffer transfer time point is when the input action is captured.

Dedicated Baseline Value Cache Delivery Time Point

Dedicated comparison reference value can choose single buffer function or double buffer function (BCONR.BSESPA or BCONR.BSESPB). The buffer transfer time point is set by BCONR.BTRSPA and BCONR.BTRSPB of the buffer control register BCONR.

Dead Time Reference Buffer Transfer Time Point

The dead-time reference has a single-buffer function. The buffer transmission time point is the counting overflow point or the counting down point underflowing point during the sawtooth wave, and the counting valley point during the triangular wave.

Buffer Transfer During Clear Action

In Sawtooth Wave Counting Mode or Hardware Counting Mode, if there is a clean action during the comparison output operation, The General Cycle Reference Value, General C Omparison Reference Value, Special Comparison Reference Value, DEAD TIME Reference Value Register, ETC. Will Act According to The corresponding buffer A buffer transfer occurs for the set state (single buffer, double buffer, etc.).

18.3.10 General PWM Output

18.3.10.1 Independent PWM Output

The two ports TIM6_<t>_PWMA and TIM6_<t>_PWMB of each unit can independently output PWM waves. For example Figure 18-14, TIM6_<t>_PWMA port outputs PWM wave.

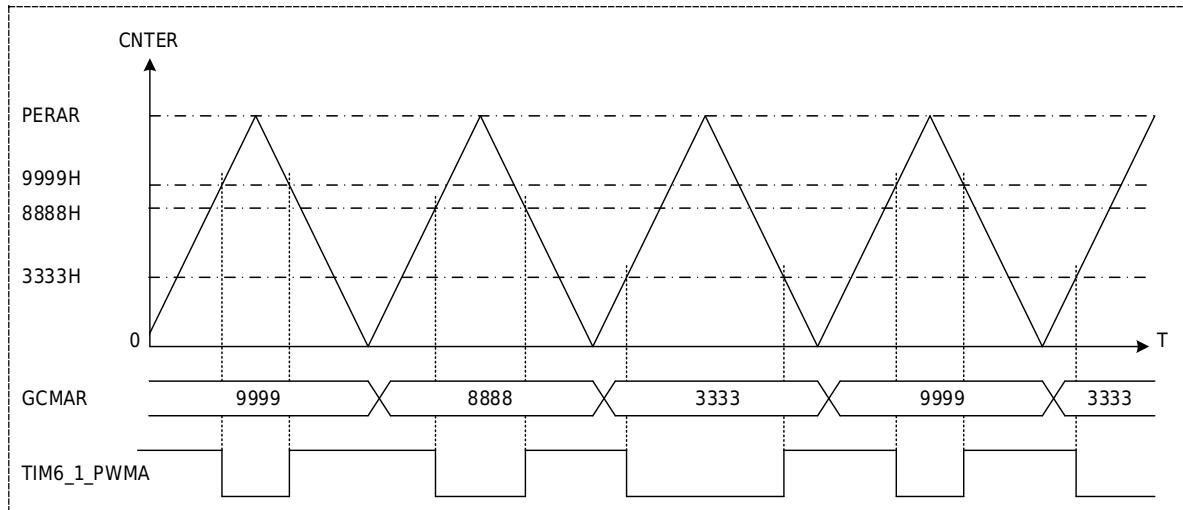


Figure 18-14 TIM6_<t>_PWMA output PWM wave

18.3.10.2 Complementary PWM Output

The TIM6_<t>_PWMA port and the TIM6_<t>_PWMB port can be combined to output complementary PWM waveforms in different modes.

Software Setting GCMBR Complementary PWM Output

Software setting GCMBR complementary PWM output means that in sawtooth wave mode and triangular wave mode, the general comparison reference value register (GCMBR) used for TIM6_<t>_PWMB port waveform output is directly written by the CPU, etc., and has no direct relationship with the value of GCMAR. relation.

Figure 18-15 shows an example of software setting GCMBR complementary PWM wave output

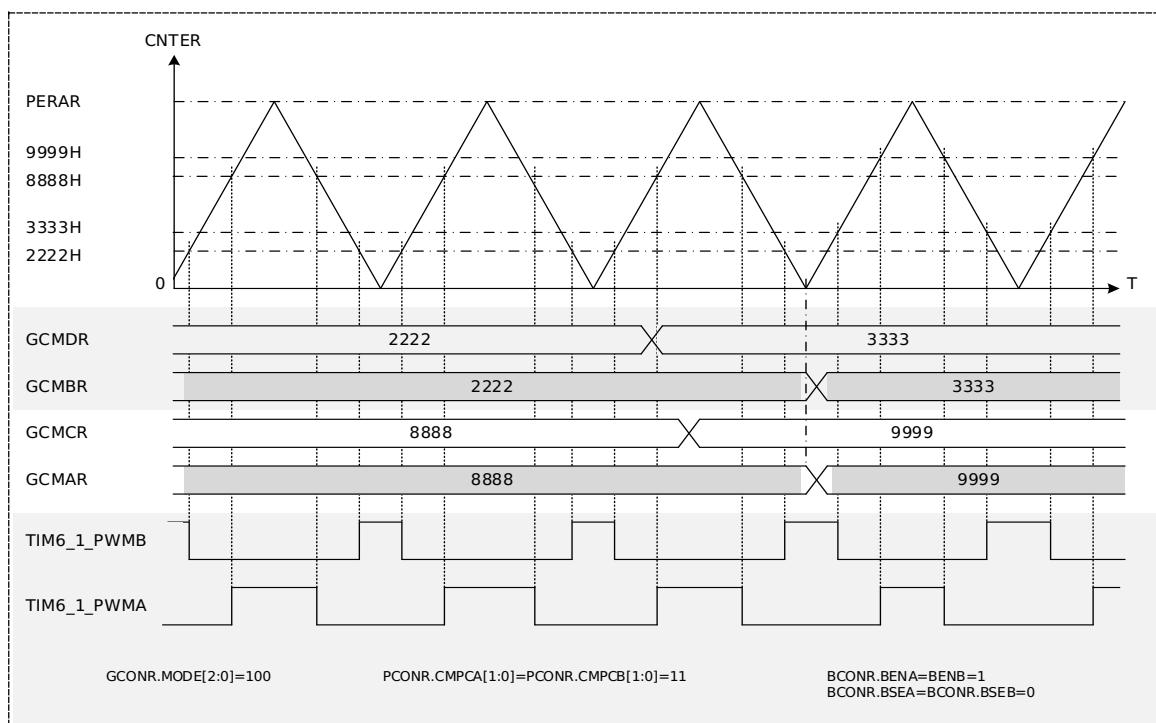


Figure 18-15 Triangle wave A mode software setting GCMBR complementary PWM wave output

Hardware Setting GCMBR Complementary PWM Output

Hardware setting GCMBR complementary PWM output means that in the triangle wave mode, the value of the general comparison reference value register (GCMBR) used for TIM6_<t>_PWMB port waveform output is determined by the general comparison reference value register (GCMAR) and the dead time reference value The value operation of the register (DTU<D>AR) is determined.

The dead time setting also has a cache function. When the cache function is valid (DCONR.DTBENU/DTBEND=1), at the time point of buffer transmission (counting valley point during triangular wave), the value of DTUBR is transmitted to DTUAR, and the value of DTDBR is transmitted to DTDAR.

In Figure 18-16, Example of setting GCMBR complementary PWM wave output for hardware.

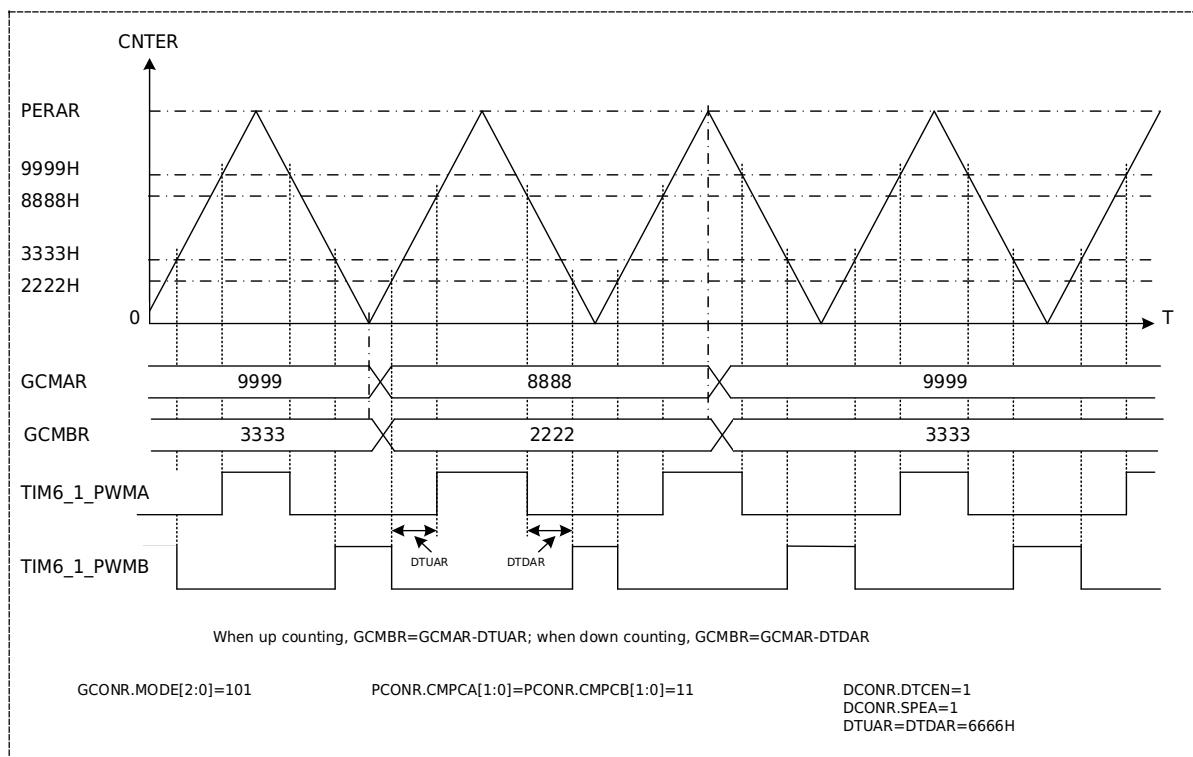


Figure 18-16 Triangle wave B mode hardware setting GCMBR complementary PWM wave output (symmetrical dead zone)

18.3.10.3 Multi-phase PWM output

The TIM6_<t>_PWMA and TIM6_<t>_PWMB ports of each unit can output 2 phases of independent PWM waves or a set of complementary PWM waves, multiple units can be combined, and combined with software and hardware synchronous actions to achieve multi-phase PWM wave output. As shown in Figure 18-17, the combination of unit 1, unit 2 and unit 3 outputs 6-phase PWM waves; As shown in Figure 18-18, the combination of unit 1, unit 2 and unit 3 outputs 3 complementary PWM waves.

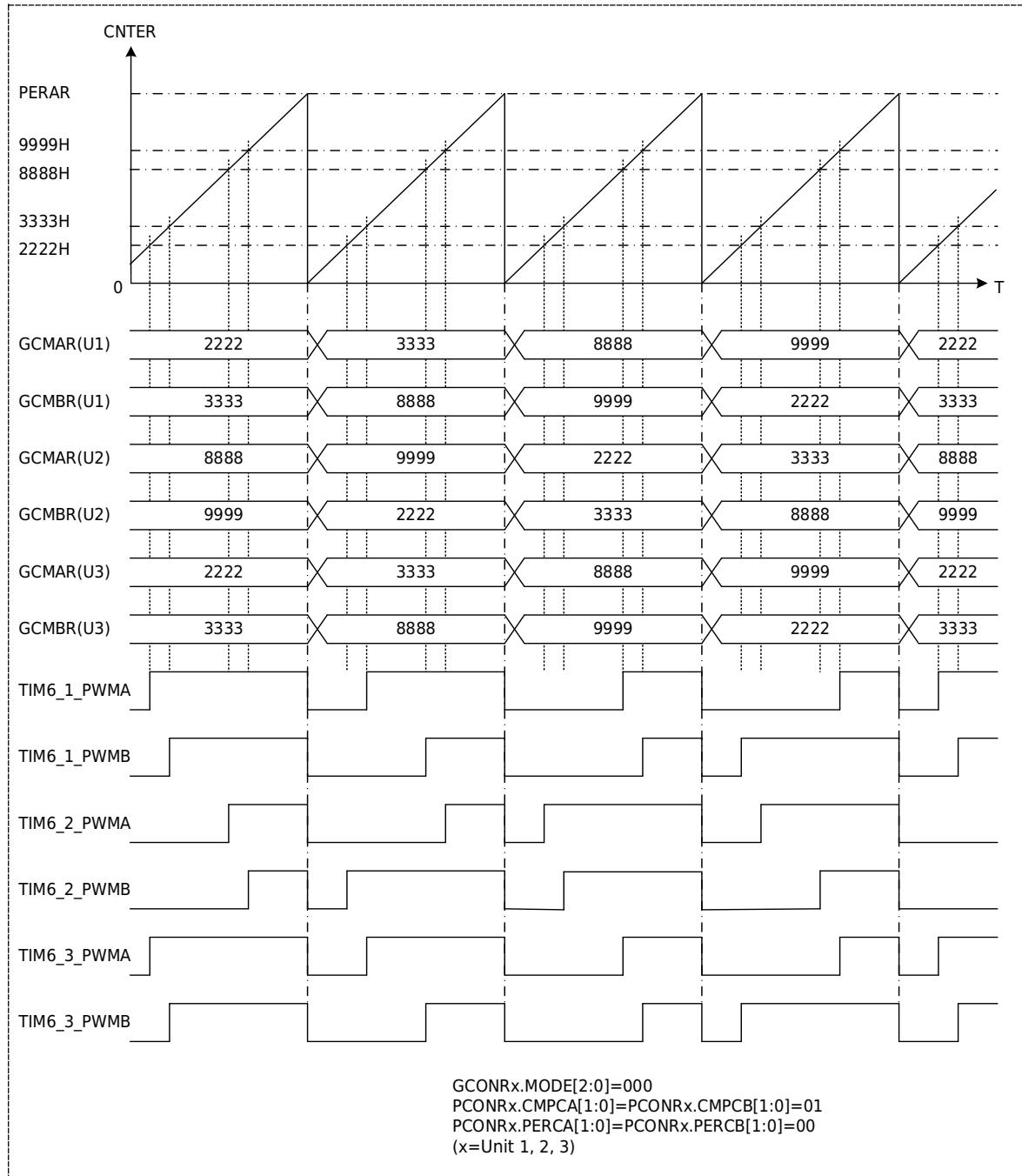


Figure 18-17 6-phase PWM wave

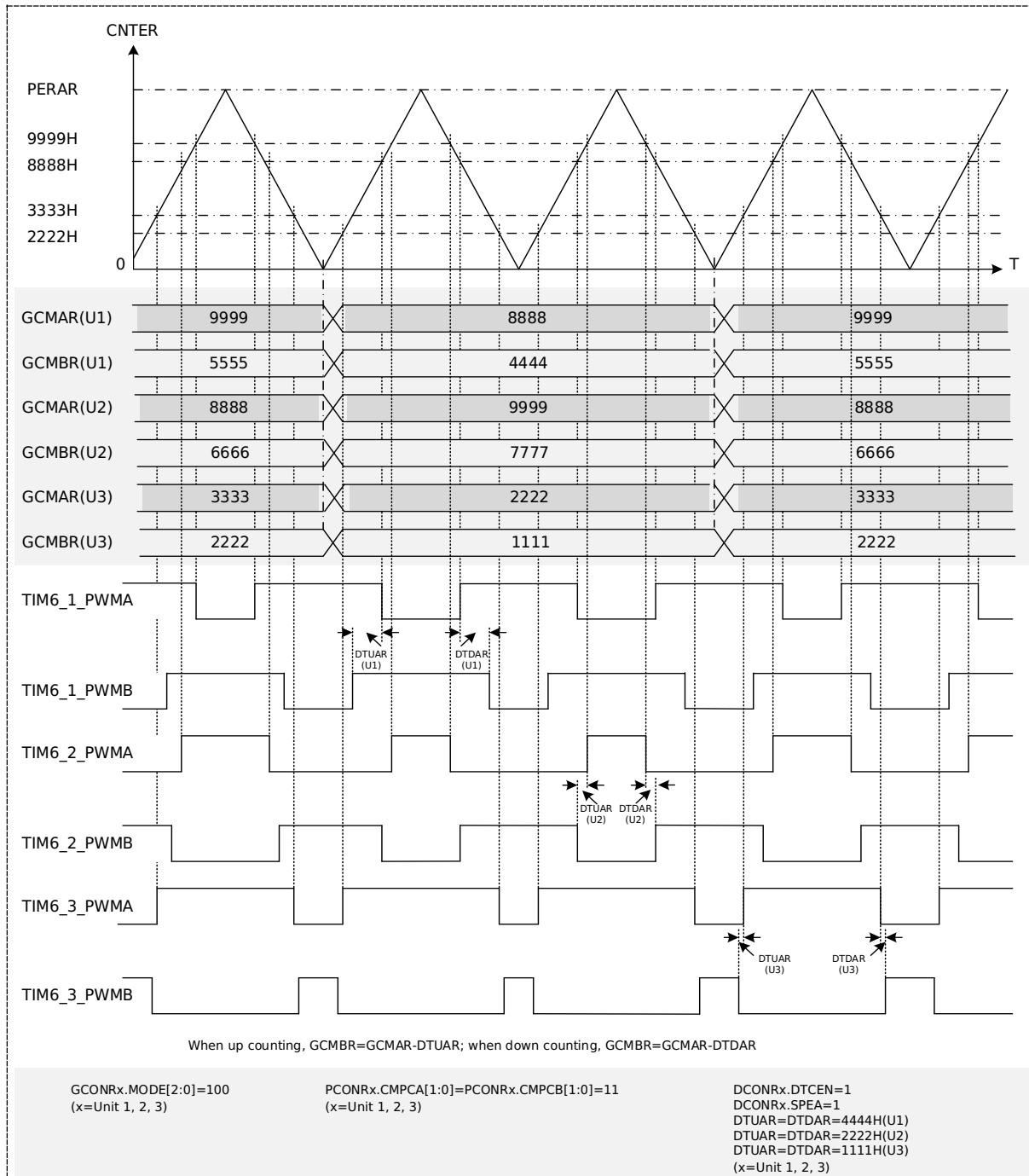


Figure 18-18 Three-phase complementary PWM wave output with dead time in triangular wave A mode

18.3.11 Orthogonal Coding Count

Treating TIM6_<t>_PWMA input as AInput, TIM6_<t>_PWMB input as BInput, and any input in TIM6_TRIGA-B as ZInput, Timer6 can realize the quadrature encoding count of 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, one unit is used for position counting, and the other is used for revolution counting.

In the revolution counting mode, units 1 and 2 are combined, unit 1 is used as a position counting unit, and unit 2 is used as a revolution counting unit to realize position counting and revolution counting respectively. Unit 3 is not used in revolution mode.

The counting conditions of AIN and BIN are realized by setting the orthogonal relationship between TIM6_<t>_PWMA and TIM6_<t>_PWMB in the hardware increment event selection register (HCUPR) and hardware decrement event selection register (HCDOR); the input action of ZIN Clear the position timer of the position counting unit by setting the hardware clearing event selection register (HCLRR) of the position unit, and realize the revolution timer counting of the revolution counting unit by setting the hardware increment event selection register (HCUPR) of the revolution counting unit .

18.3.11.1 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 the AIN or BIN port, as shown in Figure 18-19 below.

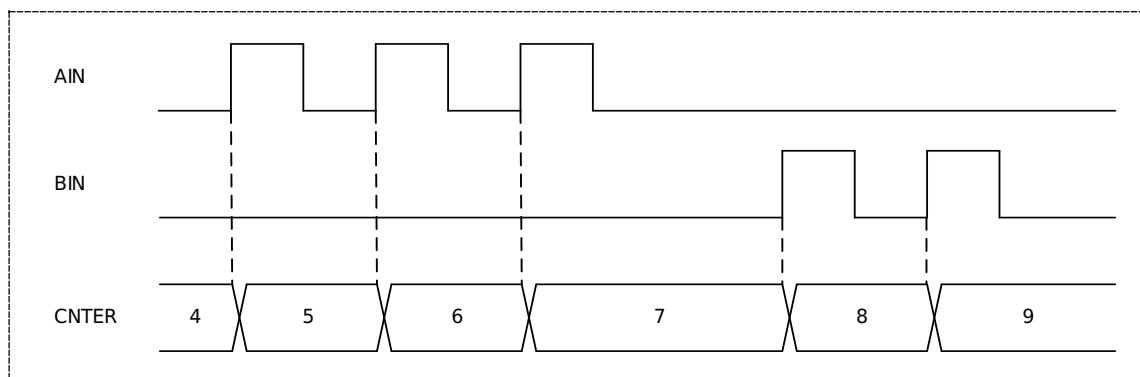


Figure 18-19 Position Mode - Basic Counting

Phase Difference Count

Phase difference counts are counted according to the phase relationship between AIN and BIN. According to different settings, 1-fold counting, 2-fold counting, 4-fold counting, etc. can be realized, as shown in the following Figure 18-20~Figure 18-22.

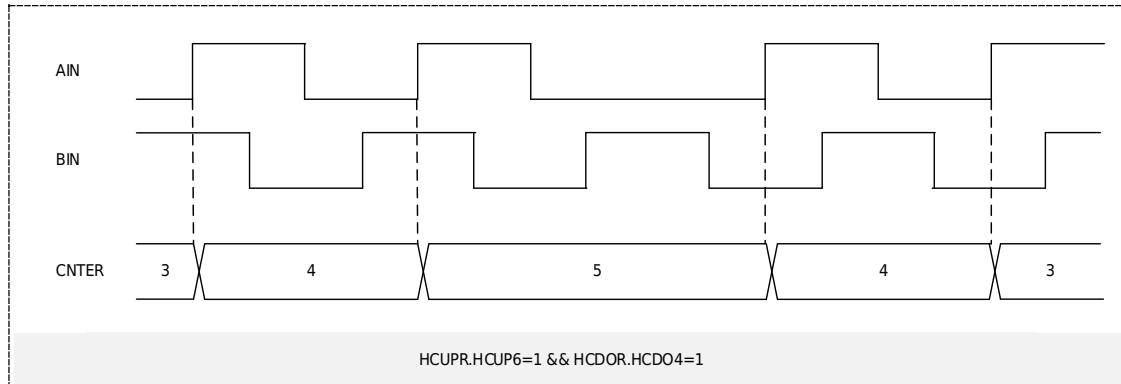


Figure 18-20 Position Counting Mode - Phase Difference Counting (1 times counting)

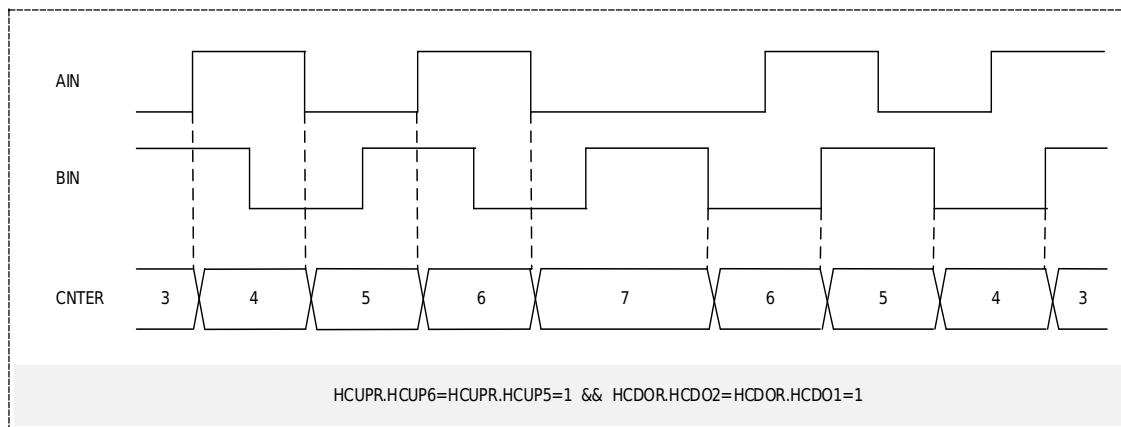


Figure 18-21 Position Counting Mode - Phase Difference Counting (2 times counting)

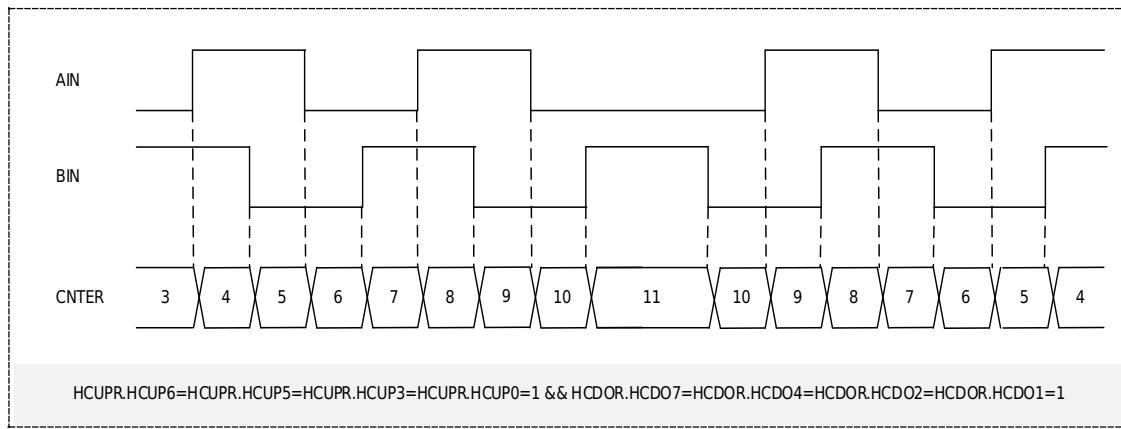


Figure 18-22 Position Counting Mode - Phase Difference Counting (4 times counting)

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 the Figure 18-23 below.

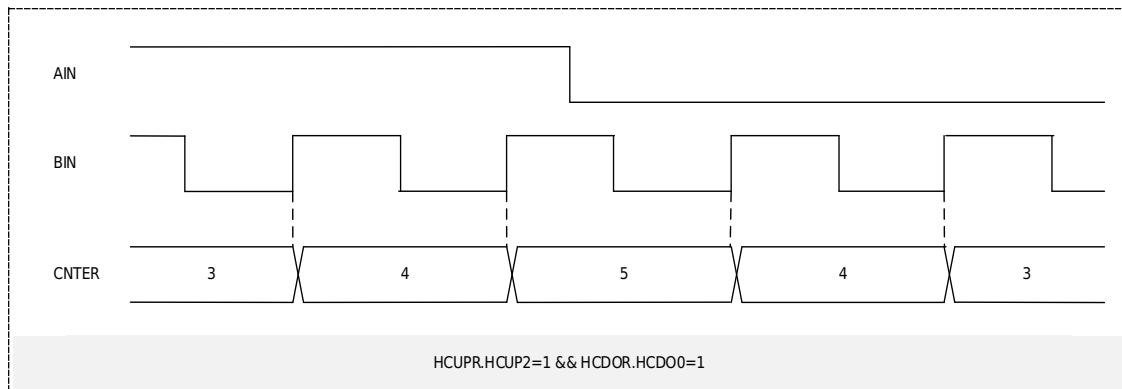


Figure 18-23 Position Counting Mode - Direction Counting

18.3.11.2 Revolution Counting Mode

Orthogonal encoding revolution counting mode refers to the addition of ZInput events on the basis of AIN and BIN counting to realize the judgment of the number of revolutions, etc. In the revolution counting mode, according to the counting method of the revolution timer, the Z-phase counting function, the position overflow counting function and the mixed counting function can be realized.

Z Phase Count

Z-phase counting refers to the counting action that the revolution counting unit counts according to the input of ZIN, and the position counting unit is cleared at the same time. As shown in Figure 18-24 below.

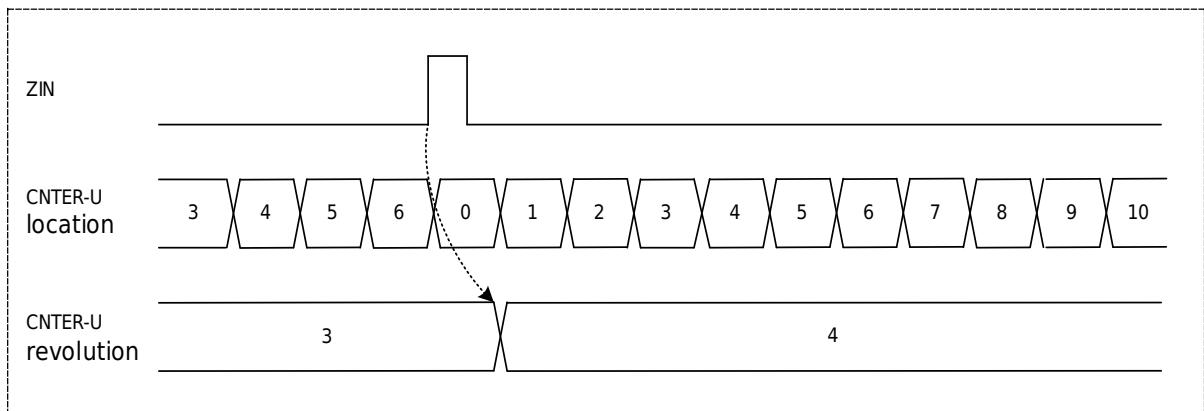


Figure 18-24 Revolution Counting Mode - Z-Phase Counting

Position Overflow Count

Position overflow counting means that when the counting of the position counting unit overflows or underflows, an overflow event is generated, thereby triggering 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 event of the position counting unit is gated through the internal trigger event interface to realize the counting of the revolution counting unit, and the position overflow counting can be realized. The increment (decrement) event selection register (HCUPR or HCDOR) of the hardware increment (decrement) event selection register (HCUPR or HCDOR) of the revolution counting unit selects 1 bit in Bit16~Bit7, and at the same time triggers the corresponding event in the event selection register (HTSSR0~1). The number is set as an overflow or underflow event of the position counting unit. For specific event numbers, refer to Interrupt Controller (INTC) chapters. As shown in Figure 18-25 below.

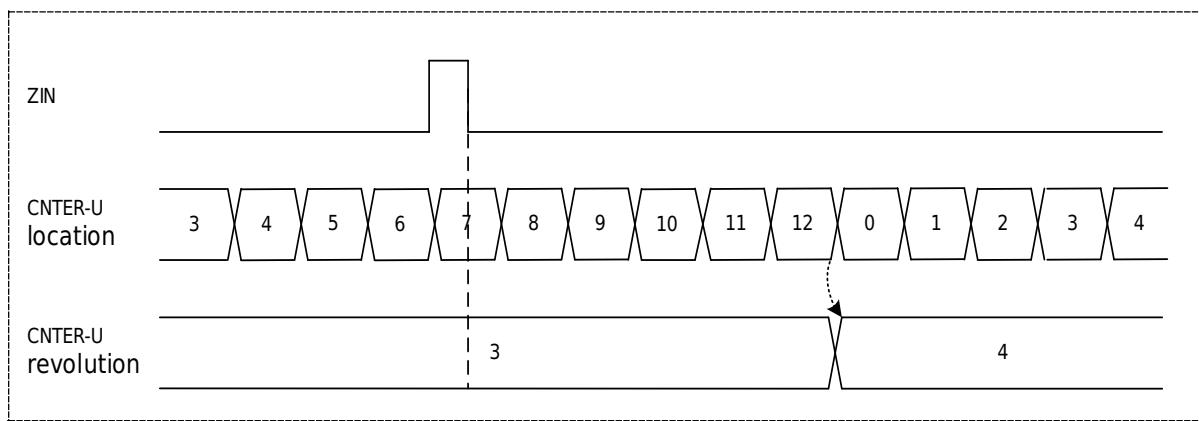


Figure 18-25 Revolution Counting Mode - Position Overflow Counting

Mixed Count

Mixed counting refers to the counting action that combines the above two counting methods of Z-phase counting and position overflow counting, and its realization method is also a combination of the above two counting methods. As shown in Figure 18-26 below.

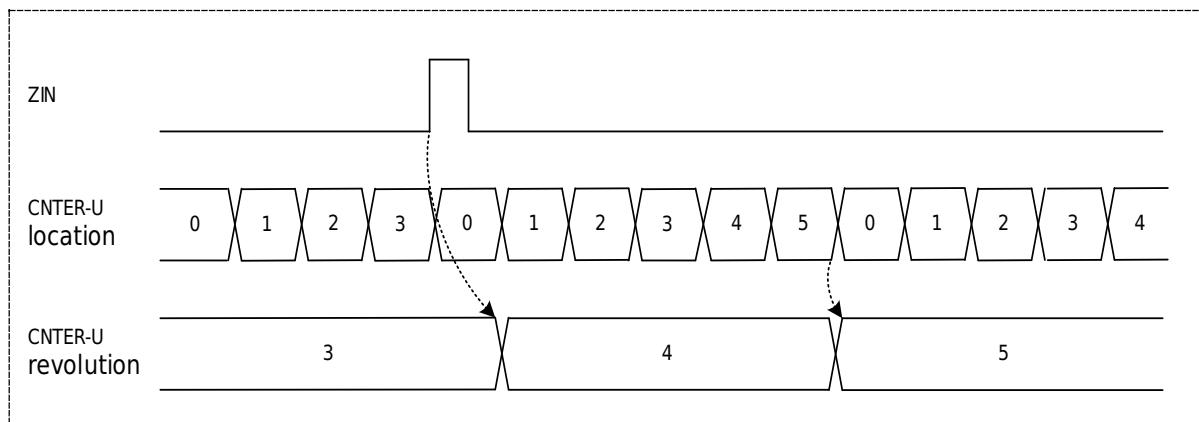


Figure 18-26 Revolution Counting Mode - Mixed Counting

18.3.11.3 Z Phase Action Shielding

In the Z-phase counting function or mixed counting function of the revolution counting mode, it can be set within a few cycles after the overflow point or underflow point of the position timer (GCONR.ZMSKVAL[0:1] setting), the effective input of ZIN is masked, and the counting of the revolution counting unit and the clearing of the position counting unit are not performed. When this function is used, it can only be realized by combining units 1 and 2, unit 1 is used as a position counting unit, and unit 2 is used as a revolution counting unit.

When the GCONR.ZMSKPOS of the general control register (GCONR) of the position counting unit is 1, the Z-phase shielding function of the position counting unit is enabled, and the number of cycles of Z-phase shielding is set by GCONR.ZMSKVAL; the general control register of the revolution counting unit (When GCONR.ZMSKREV of GCONR) is 1, the Z-phase shielding function of the revolution counting unit is enabled. Figure 18-27 shows the mixed counting of the revolution counting mode, when there is a ZIN phase input within 4 counting cycles after the position counting unit overflows, the action of the ZIN phase input is invalid, that is, the revolution counting unit does not count, and the position counting unit does not clear; after that Then the ZIN phase input works normally.

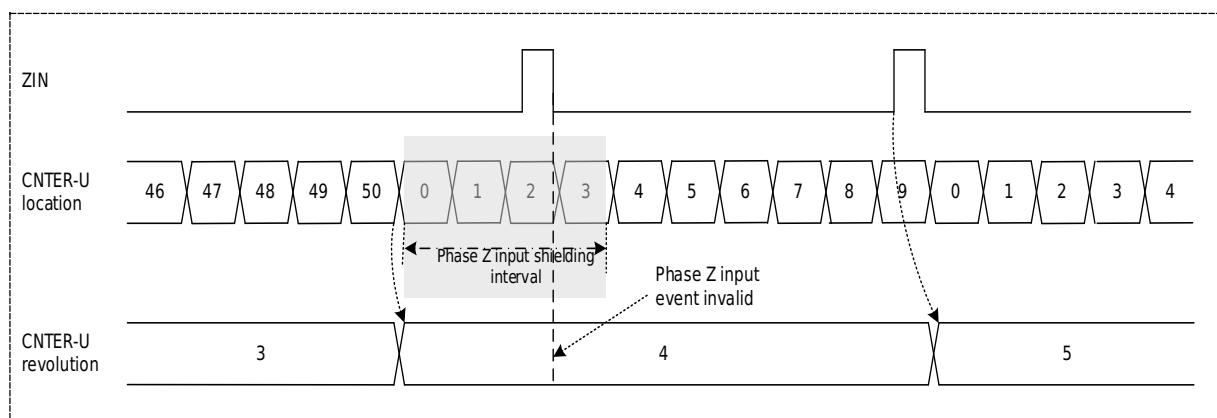


Figure 18-27 Revolution Counting Mode - Mixed Counting Z-Phase Shielding Action Example 1

Figure 18-28 shows the mixed counting mode in the revolution counting mode, the counting diary changes in the third cycle after the posenting unit overflows, and the mask ING Cycle of 4 Cycles Set at this time Becomes Invalid (The Actual Zin Phase Masking Function Maintautas 3 Cycles), Start counting down. After the counting underflow occurs in the position counting unit, the ZIN phase shielding function is re-opened and becomes invalid after 4 cycles. During the masking period of the ZIN phase, the input function of the ZIN phase is invalid, that is, the revolution counting unit does not count, and the position counting unit does not clear; the subsequent ZIN phase input operates normally.

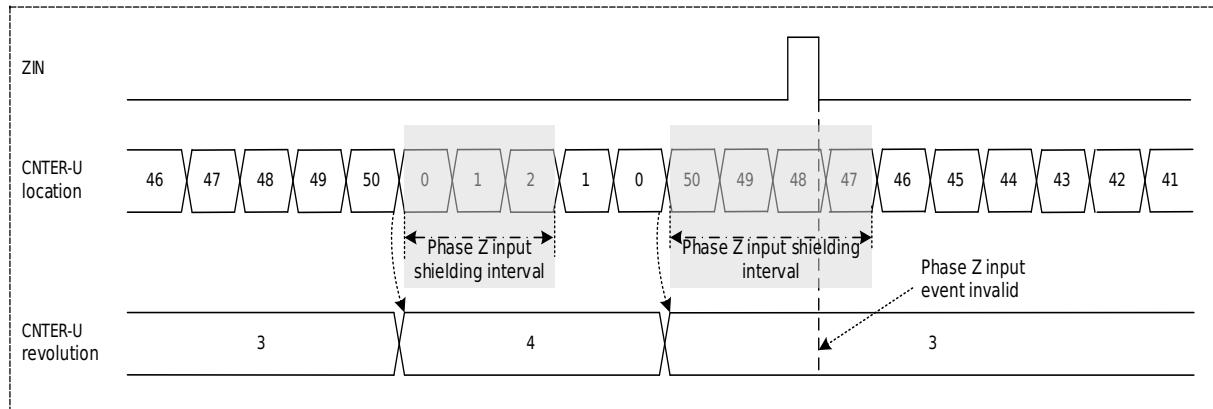


Figure 18-28 Revolution Counting Mode - Mixed Counting Z-Phase Shielding Action Example 2

18.3.12 Periodic interval response

The two dedicated comparison reference value registers (SCMAR, SCMBR) of Timer6 can respectively output the special comparison match interrupt A signal and the special comparison match interrupt B signal to INTC to generate the corresponding interrupt when the counting comparison matches; same time, they can output the special comparison match The event A signal and the dedicated comparison match event B signal are used to associate actions with other modules, and are mostly used to start the ADC, etc.

The interrupt and event request signal can generate an effective request signal every several cycles, that is, realize periodic interval response. This function is enabled by setting the VPERR.PCNTE[1:0] bits and VPERR.SPPERIA/B bits of the valid period register (VPERR). Set the VPERR.PCNTS[2:0] bit to specify how many cycles the request signal is valid for once. In other cycles, even if the count value is equal to the value of the dedicated comparison reference value register SCMAR or SCMBR, a valid request signal will not be output. .

If you stop and restart the timer when using the periodic interval response function, please configure VPERRR. There is a deviation.

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 dedicated compare match interrupt and event output (the period of STFLR.VPERNUM=0 in the figure below). Figure 18-29 shows an example of the operation of the periodic interval valid request signal.

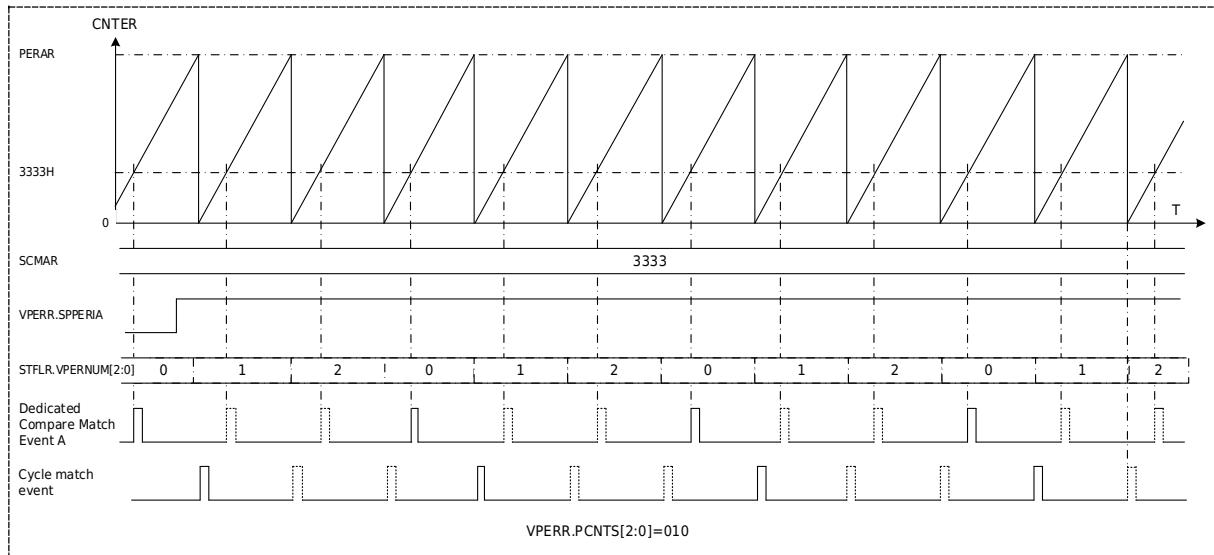


Figure 18-29 Cycle Interval Valid Request Signal Action

18.3.13 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 a common port output control interface, which is connected to a group of EMB events output by the EMB module. At the same time, the abnormal condition events gated on the interface can be set from the EMB side (see the EMB chapter). When abnormal conditions are detected on these interfaces, the control of the general PWM output can be realized.

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 an EMB abnormal event occurs on the general-purpose PWM output port, the port state can change to output high-impedance state, output low level or output high level (determined by the settings of PCONR.EMBVALA and PCONR.EMBVALB). For example, if PCONR.EMBVALA=01 is set, then during the normal output period of the TIM6_<t>_PWMA port, if an EMB event occurs, the output of the TIM6_<t>_PWMA port will become a high-impedance state.

After the abnormal event selected by the EMB module disappears and the EMB module resets the corresponding event status bit, Timer6 will automatically release the protection state at the next cycle point (the counting trough of the triangle wave, the overflow point or the underflow point of the sawtooth wave), It becomes a normal PWM output, thereby realizing Cycle By Cycle control of the PWM port.

18.3.14 Typical Application

The following describes the basic settings of Timer6 related registers in several typical application situations for user reference.

18.3.14.1 Basic Counting and Interrupt Actions

- a) Setting the universal period reference value (PERAR)
- b) Set the required comparison reference value, including general comparison reference value (GCMAR~GCMFR), special comparison reference value (SCMAR~SCMBR), etc.
- c) 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.
- d) Set internal count clock frequency division (GCONR.CKDIV[2:0])
- e) Set the waveform mode (GCONR.MODE[2:0])
- f) Set the counting direction (need to be set only when sawtooth wave mode GCONR.MODE[2:0]=000)
- g) Start counter (GCONR.START=1)

18.3.14.2 Comparison Output and Interrupt Action

- a) Setting the Universal Period Reference Value (PERAR)
- b) Set the comparison reference value of each channel, including general comparison reference value A (GCMAR), general comparison reference value B (GCMBR)
- c) Set the required interrupt enable bit, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), count match interrupt (ICONR.INTENA~B), etc.
- d) Set the port output state of each channel in different counting states (refer to the related control of bit7~bit1 and bit23~bit17 of PCONR)
- e) Set internal count clock frequency division (GCONR.CKDIV[2:0])
- f) Set the waveform mode (GCONR.MODE[2:0])
- g) Set the counting direction (need to be set only when sawtooth wave mode GCONR.MODE[2:0]=000)
- h) Set the comparison output mode of each channel (PCONR.CAPMDA=0, PCONR.CAPMDB=0)
- i) Set the output enable of each channel (PCONR.OUTENA=1, PCONR.OUTENB=1)
- j) Start counter (GCONR.START=1)

18.3.14.3 Capture Input and Interrupt Actions

- a) Setting the Universal Period Reference Value (PERAR)
- b) Set the required interrupt enable bit, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), capture input interrupt (ICONR.INTENA~B), etc.
- c) Set the capture input external conditions of each channel (refer to all valid control bits of HCPAR or HCPBR. The effective control bits are independent of each other, and multiple

can be selected as the capture input conditions of a certain channel at the same time)

- d) Set internal count clock frequency division (GCONR.CKDIV[2:0])
- e) Set the waveform mode (GCONR.MODE[2:0])
- f) Set the counting direction (need to be set only when sawtooth wave mode GCONR.MODE[2:0]=000)
- g) Set capture input mode (PCONR.CAPMDA=1, PCONR.CAPMDB=1)
- h) Start counter (GCONR.START=1)
- i) Wait for the capture input condition to be generated, read the capture input value of the corresponding channel (GCMAR or GCMBR) or wait for the corresponding interrupt to be generated

18.3.14.4 Buffer Transfer Action (Period Reference Value)

- a) Set desired common period reference values (PERAR, PERBR, PERCR)
- b) Set single and double buffer transfer mode (BCONR.BSEP)
- c) Set internal count clock frequency division (GCONR.CKDIV[2:0])
- d) Set the waveform mode (GCONR.MODE[2:0]) (buffer transmission time points are different in different waveform modes)
- e) Set the counting direction (need to be set only when sawtooth wave mode GCONR.MODE[2:0]=000)
- f) Set the cache function to be valid (BCONR.BENP=1)
- g) Start counter (GCONR.START=1)
- h) Wait for the corresponding cache transmission time point, and the cache action occurs (PERBR->PERAR (when BCONR.BSEP=0), PERCR->PERBR->PERAR (when BCONR.BSEP=1))

18.3.14.5 Cache Transfer Action (Common Benchmark Value)

- a) Set the desired common comparison base value (GCMAR, GCMCR, GCMER, GCMBR, GCMDR, GCMFR)
- b) Set the single and double buffer transmission mode of each channel (BCONR.BSEA, BCONR.BSEB)
- c) Set internal count clock frequency division (GCONR.CKDIV[2:0])
- d) Set the waveform mode (GCONR.MODE[2:0]) (buffer transmission time points are different in different waveform modes)
- e) Set the counting direction (need to be set only when sawtooth wave mode GCONR.MODE[2:0]=000)
- f) Set the buffer function of each channel to be valid (BCONR.BENA=1, BCONR.BENB=1)
- g) Start counter (GCONR.START=1)
- h) Wait for the corresponding buffer transmission time point set by each channel, and the 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)

18.3.14.6 Cache Transfer Action (Dedicated Benchmark Value)

- a) Set the required specific comparison reference value (SCMAR, SCMCR, SCMER, SCMBR, SCMDR, SCMFR)
- b) Set the single and double buffer transmission mode of each channel (BCONR.BSESPA, BCONR.BSESPB)
- c) Set the buffer transmission time point of each channel (BCONR.BTRSPA[1:0], BCONR.BTRSPB[1:0])
- d) Set internal count clock frequency division (GCONR.CKDIV[2:0])
- e) Set the waveform mode (GCONR.MODE[2:0])
- f) Set the counting direction (need to be set only when sawtooth wave mode GCONR.MODE[2:0]=000)
- g) Set the buffer function of each channel to be valid (BCONR.BENSPA=1, BCONR.BENSPB=1)
- h) Start counter (GCONR.START=1)
- i) Wait for the corresponding buffer transmission time point set by each channel, and the 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))

18.3.14.7 Buffer Transfer Action (Deadband Reference Value)

- a) Set the desired dead time reference (DTUAR, DTUBR, DTDAR, DTDBR)
- b) Set internal count clock frequency division (GCONR.CKDIV[2:0])
- c) Set the waveform mode (GCONR.MODE[2:0])
- d) Set the counting direction (need to be set only when sawtooth wave mode GCONR.MODE[2:0]=000)
- e) Set the cache function to be valid (DCONR.DTBENU=1, DCONR.DTBEND=1)
- f) Set the hardware dead zone function to be valid (DCONR.DTCEN=1)
- g) Start counter (GCONR.START=1)
- h) Wait for the corresponding cache transmission time point, and the cache action occurs (DTUBR->DTUAR, DTDBR->DTDAR)

18.3.14.8 Synchronous Start Action (Software Method)

- a) Refer to steps a~f in the [Basic Counting and Interrupt Action] chapter to set the units that need to be started synchronously
- b) Start the counter synchronously (set the corresponding bit of the SSTAR register to 1, and each unit corresponds to a register bit)

18.3.14.9 Synchronous Start Action (Hardware Method)

- a) Setting the Universal Period Reference Value (PERAR)
- b) Set the required comparison reference value, including general comparison reference value (GCMAR~GCMFR), special comparison reference value (SCMAR~SCMBR), etc.
- c) 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.
- d) Set hardware start conditions (selected by HSTAR.HSTAx, x=0~1, 8~11)
- e) Set hardware start enable (HSTAR.STARTS=1)
- f) Repeat steps a~e above to make settings for each unit that needs to be started synchronously (in each unit that needs to be started synchronously, the setting of step d must be consistent)
- g) Wait for the set trigger event to occur, and confirm that the counters of each unit start synchronously

18.3.14.10 Quadrature Encoder Counting Action (2 Phases)

- a) Setting the Universal Period Reference Value (PERAR)
- b) Set the required comparison reference value, including general comparison reference value (GCMAR~GCMFR), special comparison reference value (SCMAR~SCMBR), etc.
- c) 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.
- d) Set the required hardware count-up condition (selected by HCUPR.HCUPx, x=0~7)
- e) Set the required hardware down counting conditions (selected by HCDOR.HCDOx, x=0~7)
- f) Start counter (GCONR.START=1)
- g) Wait for the set quadrature encoding counting event to be generated, and confirm that the counter counts normally

18.3.14.11 Quadrature Encoder Counting Action (3 Phases)

- a) Refer to steps a~e in the chapter [Quadrature Encoder Counting Action (2 Phases)] to set the position counting unit
- b) Set the hardware clearing condition of the position counting unit (selected by HCLRR.HCLRx, x=8~11)
- c) Set position counter unit hardware clear enable (HCLRR.CLEARS=1)
- d) Set the general period reference value (PERAR) of the revolution counter unit
- e) Set the comparison reference value of revolution counting unit, including general

comparison reference value (GCMAR~GCMFR), special comparison reference value (SCMAR~SCMBR), etc.

- f) 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.
- g) Set the hardware up counting condition 1 of the revolution counting unit (ZIN phase input) (selected by HCUPR.HCUPx, x=8~11, the set event here should be consistent with the event set by the position counting unit in step b)
- h) Set the hardware up counting 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.HCUP16)
- i) Set the hardware down counting condition of the revolution counting unit (the underflow event input of the position counting unit) (select internal hardware trigger event 1 through HCDOR.HCDO17)
- j) Set the trigger source number in HTSSR0 to the count overflow event of the position counting unit (the overflow event number refers to the INTC chapter)
- k) Set the trigger source number in HTSSR1 to the counting underflow event of the position counting unit (for the underflow event number, refer to the INTC chapter)
- l) Start revolution counting unit counter (GCONR.START=1)
- m) Start position counting unit counter (GCONR.START=1)
- n) Wait for the set AIN, BIN, ZIN phase count event to occur, and confirm that the counter counts normally

18.3.14.12 Single PWM Output

- a) Refer to the setting of steps a~j in chapter [Comparison Output and Interrupt Action] (the output states of the two PWM channels TIM6_<t>_PWMA and TIM6_<t>_PWMB inside each unit can be set independently to form two mutual unrelated single PWM output)

18.3.14.13 Complementary PWM Output (Software Dead Time)

- a) Setting the Universal Period Reference Value (PERAR)
- b) Set general comparison reference value A (GCMAR) and general comparison reference value B (GCMBR)
- c) Set the required interrupt enable bit, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), count match interrupt (ICONR.INTENA~B), etc.
- d) Set the port output state in different counting states (refer to the related control of bit7~bit1, bit23~bit16 of PCONR, combined with the setting values of GCMAR and GCMBR, it is necessary to ensure that a complementary dead zone is formed between the two PWM

- outputs)
- e) Set internal count clock frequency division (GCONR.CKDIV[2:0])
 - f) Set the waveform mode to triangle wave mode (GCONR.MODE=100 or 101)
 - g) Set comparison output mode (PCONR.CAPMDA=0, PCONR.CAPMDB=0)
 - h) Set output enable (PCONR.OUTENA=1, PCONR.OUTENB=1)
 - i) Start counter (GCONR.START=1)

18.3.14.14 Complementary PWM Output (Hardware Dead Zone)

- a) Setting the Universal Period Reference Value (PERAR)
- b) Set general comparison reference value A (GCMAR), dead time reference value (DTUAR, DTDAR)
- c) Set the required interrupt enable bits, including count overflow interrupt (ICONR.INTNOVF), count underflow interrupt (ICONR.INTNUDF), count match interrupt (ICONR.INTNA~B), dead zone error interrupt (ICONR.INTENDTE) wait
- d) Set the port output state in different counting states (refer to the related control of bit7~bit1, bit23~bit16 of PCONR, combined with the setting values of GCMAR, DTUAR and DTDAR, it is necessary to ensure that a complementary dead zone is formed between the two PWM outputs)
- e) Set internal count clock frequency division (GCONR.CKDIV[2:0])
- f) Set the waveform mode to triangle wave mode (GCONR.MODE[2:0]=100 or 101)
- g) Set the comparison output mode of each channel (PCONR.CAPMDA=0, PCONR.CAPMDB=0)
- h) Set the output enable of each channel (PCONR.OUTENA=1, PCONR.OUTENB=1)
- i) Set the hardware dead zone function to be valid (DCONR.DTCEN=1)
- j) Start counter (GCONR.START=1)

18.3.14.15 EMB Monitoring and Interrupt Action

- a) Refer to steps a~h in chapter [Complementary PWM Output (Software Dead Time)] or steps a~i in chapter [Complementary PWM Output (Hardware Dead Zone)] to set the complementary PWM output action
- b) Set the state of the PWM port (PCONR.EMBVALA, PCONR.EMBVAB) when the EMB event occurs (select the corresponding protection state according to the system application)
- c) Set the relevant registers of the EMB module (including EMB interrupt permission register (EMB_INTENO), EMB control register 0 (EMB_CTL0), etc.)
- d) Start the counter (GCONR.START=1), the EMB module monitors the system status in real time

18.3.15 Function Summary Table

In the sawtooth mode and triangular wave A and B modes of Timer6, the summary table of the main functions is shown in the table below.

Table 18-3 Comparison of functions in different modes

PWM output function			Sawtooth wave	Triangle wave		Corresponding register setting (X=A,B)	Remark
				Triangle wave A	Triangle wave B		
Independent PWM Output	Port Status Control	Port output setting at startup	Support	Support	Support	PCONR.STACK PCONR.STASTPSX	
		Port output setting at stop	Support	Support	Support	PCONR.STOPCX PCONR.STASTPSX	
		Port output setting for compare match	Support	Support	Support	PCONR.CMPCX	
		Port output setting at cycle match	Support	Support	Support	PCONR.PERCX	
	Cache Send	Cycle Reference value	Single cache	Support	Support	Control bits: BCONR.BENP Benchmark: PERAR, PERBR	
			Double buffer	Support	Support	Control bits: BCONR.BSEP Reference value PERAR,PERBR,PERCR	
		Compare Reference value	Single cache	Support	Support	Control bits: BCONR.BENX Reference value GCMAR,GCMCR GCMBR,GCMDR	The buffer transmission points of the triangular wave A mode and the triangular wave B mode are different
			Double buffer	Support	Support	Control bits: BCONR.BSEX Reference value GCMAR,GCMCR,GCMER GCMBR,GCMDR,GCMFR	
	Emergency brake			Support	Support	PCONR.EMVALX	
Complementary PWM Output	Port Status Control	Port output setting at startup	Support	Support	Support	PCONR.STACK PCONR.STASTPSX	
		Port output setting at stop	Support	Support	Support	PCONR.STACK PCONR.STASTPSX	
		Port output setting for compare match	Support	Support	Support	PCONR.CMPCX	
		Port output setting at cycle match	Support	Support	Support	PCONR.PERCX	
	Cache Send	Cycle Reference value	Single cache	Support	Support	Control bits: BCONR.BENP Benchmark: PERAR, PERBR	
			Double buffer	Support	Support	Control bits: BCONR.BSEP Reference value: PERAR,PERBR,PERC	
		Compare Reference value	Single cache	Support	Support	Control bits: BCONR.BENX Reference value: GCMAR,GCMCR GCMBR,GCMDR	The buffer transmission points of the triangular wave A mode and the triangular wave B mode are different
			Double buffer	Support	Support	Control bits: BCONR.BSEX Reference value: GCMAR,GCMCR,GCMER GCMBR,GCMDR,GCMFR	
		Dead zone Reference value	Single cache	Support	Support	Control bits: DCONR.DTBENU DCONR.DTBEND Reference value:	

PWM output function			Sawtooth wave	Triangle wave		Corresponding register setting (X=A,B)	Remark
				Triangle wave A	Triangle wave B		
	PWM output without dead zone					DTUAR,DTDAR DTUBR,DTDBR	
		Support	Support	Support	GCMAR=GCMBR		
	PWM output with dead zone	Software Way	Support	Support	Support	GCMAR≠GCMBR	
		Hardware Way	not support	Support	Support	Control bits: DCONR.DTCEN Reference value GCMAR,DTUAR,DTDAR	
	Emergency brake		Support	Support	Support	PCONR.EMBVALX	

18.4 Interrupt and Event Description

18.4.1 Interrupt Output

Timer6 contains 6 general-purpose counting compare match interrupts (including 2 capture input interrupts), 2 dedicated counting compare match interrupts, 2 count cycle match interrupts, and 1 dead-time error interrupt.

18.4.1.1 Count Comparison Match Interrupt

There are six general-purpose comparison reference registers (GCMAR-GCMFR), which can be compared with the count value to generate a comparison match. When the count compare matches, the STFLR.CMAF~STFLR.CMFF bits in the status flag register (STFLR) will be set to 1 respectively. At this time, if the corresponding bit in the INTENA~INTENF of the interrupt control register (ICONR) is set to 1 to enable the interrupt, the corresponding interrupt request (TMR6_U<t>_GCMA~F) will also be triggered.

The capture input action occurs when the capture input valid condition selected by the hardware capture event selection 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_U<t>_GCMA~B) will be triggered.

Two dedicated comparison reference registers (SCMAR-SCMBR) can also be compared with the count value to generate a comparison match. When the count compare matches, the STFLR.CMSPAF~CMSPBF bits in the status flag register (STFLR) will be set to 1 respectively. At this time, if the corresponding bit in the INTENSAU<D> or INTENSBU<D> of the interrupt control register (ICONR) is set to 1 to enable the interrupt, the corresponding interrupt request (TMR6_U<t>_SCMA~B) will also be triggered.

18.4.1.2 Count Cycle Match Interrupt

When the sawtooth wave counts up to the overflow point, the sawtooth wave counts down to the underflow point, the triangle wave counts to the valley point or the triangle wave counts to the peak point, the STFLR.1. At this time, if the ICONR.INTENOVF or ICONR.INTENUDF bit of the interrupt control register (ICONR) is set to enable the interrupt, the counting cycle match interrupt (TMR6_U<t>_GOVF and TMR6_U<t>_GUDF) can be triggered at the corresponding time point.

18.4.1.3 Dead Time Error Interrupt

When the value of the dead time reference register (DTU<D>AR) is loaded into the general comparison reference register (GCMBR), if it exceeds the cycle limit, a dead time error will be generated, and the STFLR of the status flag register (STFLR). The DTEF bit will be set to 1. At this time, if the INTENDTE bit of the interrupt control register (ICONR) is set to enable the interrupt, the dead time error interrupt (TMR6_U<t>_GDTE) will be triggered at this moment.

18.4.2 Event Output

During the clock counting process, if a period matching event (overflow and underflow points of the sawtooth wave, counting peak or valley point of the triangle wave), a general counting comparison match event, and a special counting comparison match event are generated, the corresponding event will be generated. The output signal is used to select to trigger other modules.

The figure below is the action example of general compare match interrupt AF&event AF, special compare match interrupt AB&event AB, cycle match interrupt&event of unit 1.

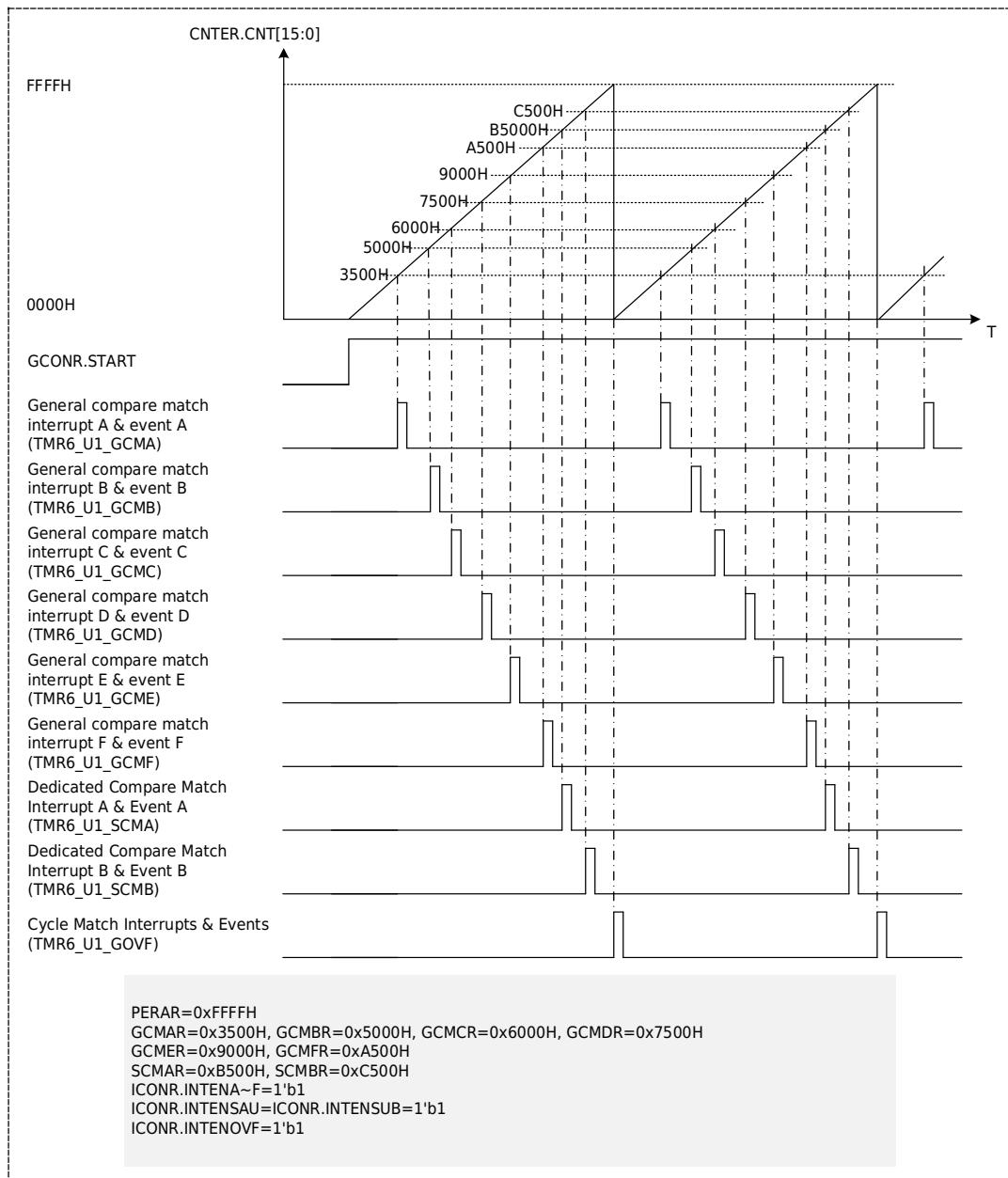


Figure 18-30 Example Of Interrupt & Event Output In Sawtooth Wave Mode

18.5 Register Description

Table 18-4 shows the register list of the Timer6 module.

BASE ADDR: 0x4001_8000 (U1), 0x4001_8400 (U2), 0x4001_8800 (U3)

Table 18-4 Register List

Register name	Symbol	Offset	Bit width	Reset value
General count register	TMR6_CNTER	0x0000	32	0x0000_0000
General purpose period reference register A	TMR6_PERAR	0x0004	32	0x0000_FFFF
General purpose period reference register B	TMR6_PERBR	0x0008	32	0x0000_FFFF
Universal Period Reference Register C	TMR6_PERCR	0x000C	32	0x0000_FFFF
General purpose comparison reference value register A	TMR6_GCMAR	0x0010	32	0x0000_FFFF
General purpose comparison reference register B	TMR6_GCMBR	0x0014	32	0x0000_FFFF
General purpose comparison reference value register C	TMR6_GCMCR	0x0018	32	0x0000_FFFF
General comparison reference value register D	TMR6_GCMDR	0x001C	32	0x0000_FFFF
General comparison reference value register E	TMR6_GCMER	0x0020	32	0x0000_FFFF
General comparison reference value register F	TMR6_GCMFR	0x0024	32	0x0000_FFFF
Dedicated comparison reference value register A	TMR6_SCMAR	0x0028	32	0x0000_FFFF
Dedicated comparison reference register B	TMR6_SCMBR	0x002C	32	0x0000_FFFF
Dedicated comparison reference value register C	TMR6_SCMCR	0x0030	32	0x0000_FFFF
Dedicated comparison reference value register D	TMR6_SCMDR	0x0034	32	0x0000_FFFF
Dedicated comparison reference value register E	TMR6_SCMER	0x0038	32	0x0000_FFFF
Dedicated comparison reference value register F	TMR6_SCMFR	0x003C	32	0x0000_FFFF
Dead Time Reference Register UA	TMR6_DTUAR	0x0040	32	0x0000_FFFF
Dead Time Reference Register DA	TMR6_DTDAR	0x0044	32	0x0000_FFFF
Dead Time Reference Register UB	TMR6_DTUBR	0x0048	32	0x0000_FFFF
DEAD TIME REFERENCE REGISTER DB	TMR6_DTDDBR	0x004C	32	0x0000_FFFF
General control register	TMR6_GCONR	0x0050	32	0x0000_0100
Interrupt control register	TMR6_ICONR	0x0054	32	0x0000_0000
Port control register	TMR6_PCONR	0x0058	32	0x0000_0000
Cache control register	TMR6_BCONR	0x005C	32	0x0000_0000
Dead Band Control Register	TMR6_DCONR	0x0060	32	0x0000_0000
Filter control register	TMR6_FCONR	0x0068	32	0x0000_0000
Valid period register	TMR6_VPERR	0x006C	32	0x0000_0000
Status flag register	TMR6_STFLR	0x0070	32	0x8000_0000
Hardware Boot Event Select Register	TMR6_HSTAR	0x0074	32	0x0000_0000
Hardware Stop Event Select Register	TMR6_HSTPR	0x0078	32	0x0000_0000
Hardware clear event select register	TMR6_HCLRR	0x007C	32	0x0000_0000
Hardware Capture Event Select Register A	TMR6_HCPAR	0x0080	32	0x0000_0000

Register name	Symbol	Offset	Bit width	Reset value
Hardware Capture Event Select Register B	TMR6_HCPBR	0x0084	32	0x0000_0000
Hardware Adder Event Selection Register	TMR6_HCUPR	0x0088	32	0x0000_0000
Hardware Decrease Event Selection Register	TMR6_HCDOR	0x008C	32	0x0000_0000
Software Synchronous Start Control Register	TMR6_SSTAR	(0x4001_83F4)	32	0x0000_0000
Software Synchro Stop Control Register	TMR6_SSTPR	(0x4001_83F8)	32	0x0000_0000
Software Synchronous Clear Control Register	TMR6_SCLRR	(0x4001_83FC)	32	0x0000_0000

Note:

- The software synchronization registers (TMR6_SSTAR, TMR6_SSTPR, TMR6_SCLRR) are 3 registers independent of the unit, shared by the 3 units Timer6.

18.5.1 General Counter Value Register (TMR6_CNTER)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0"	R											
b15~b0	CNT[15:0]	Count value	Current timer count	R/W											

18.5.2 Universal Period Reference Register (TMR6_PERA R-PERCR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PERA-C[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0"	R											
b15~b0	PERA-C[15:0]	Count cycle value	Set the counting cycle value and corresponding cache value of each round of counting	R/W											

18.5.3 General Compare Reference Registers (TMR6_GCMAR-GCMFR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
GCMA-F[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0"	R											
b15~b0	GCMA-F[15:0]	Count baseline	The comparison reference value is set, and the matching signal is valid when it is equal to the count value	R/W											

18.5.4 Dedicated Comparison Reference Registers (TMR6_SCMAR-SCMFR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SCMA-F[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0"	R											
b15~b0	SCMA-F[15:0]	Dedicated baseline value	Set comparison reference value and cache value	R/W											

18.5.5 Dead Time Reference Register (TMR6_DTU<D>AR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DTUA-B[15:0]/DTDA-B[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0"	R											
b15~b0	DTU/DA-B[15:0]	Dead time value	Dead time setting value and cache value	R/W											

18.5.6 General Control Register (TMR6_GCONR)

Reset value: 0x0000_0100

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	ZMSK VAL[1:0]	ZMSK POS	ZMSK REV	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	DIR	-	CKDIV[2:0]	CKDIV[2:0]	MODE[2:0]	MODE[2:0]	START	START	

Bit	Marking	Place name	Function	Read and write
b31~b20	Reserved	-	Read as "0", write as "0"	R/W
b19~b18	ZMSKVAL[1:0]	Phase Z input muting cycle number	Quadrature encoding phase Z input masked count period value 00: Phase Z input shielding function is invalid 01: The Z-phase input is masked within 4 counting cycles after the position count overflows or underflows 10: The Z-phase input is masked within 8 counting cycles after the position count overflows or underflows 11: The Z-phase input is masked within 16 count cycles after the position count overflows or underflows	R/W
b17	ZMSKPOS	Z phase input position timer selection	0: When the Z phase is input, the unit is used as a position timer, and the position timer clear function works normally during the masking period 1: When the Z phase is input, the unit is used as a position timer, and the position timer clearing function is masked during the masking period	R/W
b16	ZMSKREV	Z phase input revolution timer selection	0: When the Z phase is input, the unit acts as a revolution timer, and the revolution timer counting function works normally during the shielding period 1: When the Z phase is input, the unit is used as a revolution timer, and the counting function of the revolution timer is shielded during the shielding period	R/W
b15~b9	Reserved	-	Read as "0", write as "0"	R/W
b8	DIR	Counting direction	0: count down 1: count up	R/W
b7	Reserved	-	Read as "0", write as "0"	R/W
b6~b4	CKDIV[2:0]	Counting clock selection	000: PCLK0 001: PCLK0/2 010: PCLK0/4 011: PCLK0/8 100: PCLK0/16 101: PCLK0/64 110: PCLK0/256 111: PCLK0/1024	R/W
b3~b1	MODE[2:0]	Counting mode	000: Sawtooth mode 100: triangle wave A mode 101: Triangular wave B mode Please do not set other values	R/W
b0	START	Timer start	0: Timer off 1: Timer Startup Note: This bit will automatically become 0 when the software stop condition or hardware stop condition is valid	R/W

18.5.7 Interrupt Control Register (TMR6_ICONR)

Reset value: 0x0000_0000

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
-	-	-	-	-	-	-	INTEN DTE	INTEN UDF	INTEN OVF	INTEN F	INTEN E	INTEN D	INTEN C	INTEN B	INTEN A
Bit	Marking	Place name	Function												Read and write
b31~b20	Reserved	-	Read as "0", write as "0"												R/W
b19	INTENSBD	Dedicated down count interrupt enable B	0: During down counting, when the SCMBR register is equal to the count value, the interrupt is invalid 1: During down counting, when the SCMBR register is equal to the count value, the interrupt is enabled Note: When used as a dedicated compare match event output, it is also enabled by this bit												R/W
b18	INTENSBU	Dedicated up-counting interrupt enable B	0: During up-counting, when the SCMBR register is equal to the count value, the interrupt is invalid 1: During up-counting, when the SCMBR register is equal to the count value, the interrupt is enabled Note: When used as a dedicated compare match event output, it is also enabled by this bit												R/W
b17	INTENSAD	Dedicated down count interrupt enable A	0: During down counting, when the SCMAR register is equal to the count value, the interrupt is invalid 1: During down counting, when the SCMAR register is equal to the count value, the interrupt is enabled Note: When used as a dedicated compare match event output, it is also enabled by this bit												R/W
b16	INTENSAU	Dedicated up-counting interrupt enable A	0: During up-counting, when the SCMAR register is equal to the count value, the interrupt is invalid 1: During up-counting, when the SCMAR register is equal to the count value, the interrupt is enabled Note: When used as a dedicated compare match event output, it is also enabled by this bit												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 is wrong, the interrupt is invalid 1: When the dead time is wrong, the interrupt is enabled												R/W
b7	INTENUDF	Underflow interrupt enable	0: When an overflow occurs during a sawtooth wave or counts to the valley point during a triangular wave, the interrupt is invalid 1: When an underflow occurs during a sawtooth waveform or counts to a valley point during a triangular waveform, the interrupt is enabled												R/W
b6	INTENOVF	Overflow interrupt enable	0: When an overflow occurs during a sawtooth wave or counts to the peak point during a triangular wave, the interrupt is invalid 1: When an overflow occurs in a sawtooth wave or counts to the peak point in a triangular wave, 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 invalid 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 invalid 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 invalid 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 invalid 1: When the GCMCR register is equal to the count value, the interrupt is enabled												R/W
b1	INTENB	Count match interrupt enable B	0: The interrupt is invalid when the CMPARRegister is equal to the count value or when a capture input event occurs 1: This interrupt is enabled when the CMPARRegister is equal to the count value, or when a capture input event occurs												R/W

b0	INTENA	Count Match interrupt enable A	0: The interrupt is invalid when the CMPARregister is equal to the count value or when a capture input event occurs 1: This interrupt is enabled when the CMPARregister is equal to the count value, or when a capture input event occurs	R/W
----	--------	--------------------------------	--	-----

18.5.8 Port Control Register (TMR6_PCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	EMBVALB[1:0]	-	-	OUTENB	PERCB[1:0]	CMPCB[1:0]	STASTP SB	STPCB	STACB	CAPMDB			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	EMBVALA[1:0]	-	-	OUTENA	PERCA[1:0]	CMPCA[1:0]	STASTPSA	STPCASA	STACASA	CAPMDA			

Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	Read as "0", write as "0"	R/W
b28~b27	EMBVALB[1:0]	EMB State Control B	00: When the EMB condition is met, the TIM6_<t>_PWMB port outputs normally 01: When the EMB condition is met, the TIM6_<t>_PWMB port outputs a high-impedance state 10: When the EMB condition is met, the TIM6_<t>_PWMB port outputs a low level 11: When the EMB condition is met, the TIM6_<t>_PWMB port outputs a high level	R/W
b26~b25	Reserved	-	Read as "0", write as "0"	R/W
b24	OUTENB	Output enable B	0: TIM6_<t>_PWMB port output is invalid when Timer6 is functioning 1: TIM6_<t>_PWMB port output is valid when Timer6 is functioning	R/W
b23~b22	PERCB[1:0]	Port status setting when period values matchB	00: When the timer count value is equal to the period value, the TIM6_<t>_PWMB port output is set to low level 01: When the timer count value is equal to the period value, the TIM6_<t>_PWMB port output is set to high level 10: When the timer count value is equal to the period value, the TIM6_<t>_PWMB port output maintains the previous state 11: When the timer count value is equal to the period value, the TIM6_<t>_PWMB port output is set to the inversion level	R/W
b21~b20	CMPCB[1:0]	Port status setting when comparison values matchB	00: When the timer count value is equal to GCMBR, the TIM6_<t>_PWMB port output is set to low level 01: When the timer count value is equal to GCMBR, the TIM6_<t>_PWMB port output is set to high level 10: When the timer count value is equal to GCMBR, the TIM6_<t>_PWMB port output maintains the previous state 11: When the timer count value is equal to GCMBR, the TIM6_<t>_PWMB port output is set to the inverted level	R/W
b19	STASTPSB	Count start stop port state selection B	0: When counting starts or stops, the output of TIM6_<t>_PWMB port is determined by STACB and STPCB 1: When counting starts or stops, the TIM6_<t>_PWMB port output maintains the previous state Note: The counting start here refers to the initial counting start or stop and restart; the counting stop refers to the initial stop or counting start and then stop	R/W
b18	STPCB	Count stop port status setting B	0: When counting stops, TIM6_<t>_PWMB port output is set to low level 1: When counting stops, TIM6_<t>_PWMB port output is set to high level	R/W
b17	STACB	Count start port status setting B	0: When counting starts, TIM6_<t>_PWMB port output is set to low level 1: When counting starts, TIM6_<t>_PWMB port output is set to high level	R/W
b16	CAPMDB	Function mode selection B	0: Comparison output function 1: Capture input function	R/W
b15~b13	Reserved	-	Read as "0", write as "0"	R/W
b12~b11	EMBVALA[1:0]	EMB State Control A	00: When the EMB condition is met, the TIM6_<t>_PWMA port outputs normally 01: When the EMB condition is met, the TIM6_<t>_PWMA port outputs a high-impedance state 10: When the EMB condition is met, the TIM6_<t>_PWMA port outputs a low level 11: When the EMB condition is met, the TIM6_<t>_PWMA port outputs a high level	R/W
b10~b9	Reserved	-	Read as "0", write as "0"	R/W

b8	OUTENA	Output enable A	0: TIM6_<t>_PWMA port output is invalid when Timer6 is functioning 1: TIM6_<t>_PWMA port output is valid when Timer6 is functioning	R/W
b7~b6	PERCA[1:0]	Port status setting when period values matchA	00: When the timer count value is equal to the period value, the TIM6_<t>_PWMA port output is set to low level 01: When the timer count value is equal to the period value, the TIM6_<t>_PWMA port output is set to high level 10: When the timer count value is equal to the period value, the TIM6_<t>_PWMA port output maintains the previous state 11: When the timer count value is equal to the period value, the output of the TIM6_<t>_PWMA port is set to the inversion level	R/W
b5~b4	CMPCA[1:0]	Port State Setting A When Comparing Values Match	00: When the timer count value is equal to GCMAR, the TIM6_<t>_PWMA port output is set to low level 01: When the timer count value is equal to GCMAR, the TIM6_<t>_PWMA port output is set to high level 10: When the timer count value is equal to GCMAR, the TIM6_<t>_PWMA port output maintains the previous state 11: When the timer count value is equal to GCMAR, the TIM6_<t>_PWMA port output is set to the inversion level	R/W
b3	STASTPSA	Count start stop port state selection A	0: When counting starts or stops, the output of TIM6_<t>_PWMA port is determined by STACA and STPCA 1: When counting starts or stops, the TIM6_<t>_PWMA port output maintains the previous state Note: The counting start here refers to the initial counting start or stop and restart; the counting stop refers to the initial stop or counting start and then stop	R/W
b2	STPCA	Count stop port state setting A	0: When count stops, TIM6_<t>_PWMA port output is set to low 1: When count stops, TIM6_<t>_PWMA port output is set to high	R/W
b1	STACA	Count start port status setting A	0: When counting starts, the output of TIM6_<t>_PWMA port is set to low level 1: When counting starts, TIM6_<t>_PWMA port output is set to high level	R/W
b0	CAPMDA	Functional mode selection A	0: Comparison output function 1: Capture input function	R/W

18.5.9 Cache Control Register (TMR6_BCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	BTRSPB[1:0]	-	-	BSE SPB	BEN SPB	-	-	BTRSPA[1:0]	-	-	BSE SPA	BEN SPA		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	BSE P	BEN P	-	-	-	-	BSE B	BEN B	BSE A	BEN A

Bit	Marking	Place name	Function	Read and write
b31~b30	Reserved	-	Read as "0", write as "0"	R/W
b29~28	BTRSPB[1:0]	Dedicated benchmark value buffer transmission time point B	Sawtooth time 00: no transmission Other than 00: transmit when overflow or underflow Triangle wave time 00: no transmission 01: Buffer transmission when the count reaches the peak point 10: Buffer transmission when the count reaches the valley point 11: Buffer transmission when counting to peak or valley point	R/W
b27-b26	Reserved	-	Read as "0", write as "0"	R/W
b25	BSESPB	Dedicated Baseline Value Cache Transfer Option B	0: Single buffer transfer (SCMDR->SCMBR) 1: Double buffer transfer (SCMFR->SCMDR->SCMBR)	R/W
b24	BENSPB	Dedicated baseline cache transfer B	0: Buffer transmission is invalid 1: Buffer transfer enable	R/W
b23~b22	Reserved	-	Read as "0", write as "0"	R/W
b21~b20	BTRSPA[1:0]	Dedicated benchmark value cache transmission time point A	Sawtooth time 00: no transmission Other than 00: transmit when overflow or underflow Triangle wave time 00: no transmission 01: Buffer transmission when the count reaches the peak point 10: Buffer transmission when the count reaches the valley point 11: Buffer transmission when counting to peak or valley point	R/W
b19~b18	Reserved	-	Read as "0", write as "0"	R/W
b17	BSESPA	Dedicated baseline value buffer transfer option A	0: single buffer transfer (SCMCR->SCMAR) 1: Double buffer transfer (SCMER->SCMCR->SCMAR)	R/W
b16	BENSPA	Dedicated baseline cache transfer A	0: Buffer transmission is invalid 1: Buffer transfer enable	R/W
b15~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	BSEP	Period value buffer transfer selection	0: single buffer transfer (PERBR->PERAR) 1: Double buffer transfer (PERCR->PERBR->PERAR) Note: The transmission time point has nothing to do with the counting mode, only at the overflow point, underflow point of the sawtooth wave or the trough of the triangular wave	R/W
b8	BENP	Period value buffer transfer	0: Buffer transmission is invalid 1: Buffer transfer enable	R/W
b7~b4	Reserved	-	Read as "0", write as "0"	R/W
b3	BSEB	General comparison value buffer transfer option B	When comparing output functions: 0: single buffer transfer (GCMDR->GCMBR) 1: Double buffer transfer (GCMFR->GCMDR->GCMBR) When capturing the input function: 0: single buffer transfer (GCMBR->GCMDR) 1: Double buffer transfer (GCMBR->GCMDR->GCMFR)	R/W
b2	BENB	General comparison value buffer transfer B	0: Buffer transmission is invalid 1: Buffer transfer enable	R/W
b1	BSEA	General comparison value buffer	When comparing output functions: 0: single buffer transfer (GCMCR->GCMAR) 1: Double buffer transfer (GCMER->GCMCR->GCMAR)	R/W

		transfer option A	When capturing the input function: 0: single buffer transfer (GCMAR->GCMCR) 1: Double buffer transfer (GCMAR->GCMCR->GCMER)	
b0	BENA	General comparison value buffer transfer A	0: Buffer transmission is invalid 1: Buffer transfer enable	R/W

18.5.10 Dead Band Control Register (TMR6_DCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								SEPA	-	-	DTB END	DTB ENU	-	-	DTC EN
Bit	Marking		Place name		Function										Read and write
b31~b9	Reserved		-		Read as "0", write as "0"										R/W
b8	SEPA		Separate settings		0: DTUAR and DTDAR are set separately 1: The value of DTDAR is automatically equal to the value of DTUAR										R/W
b7~b6	Reserved		-		Read as "0", write as "0"										R/W
b5	DTBEND		Dead Time Value Cache Transfer D		0: Buffer transmission is invalid 1: Buffer transfer enable (DTDBR->DTDAR)										R/W
b4	DTBENU		Dead time value cache transfer U		0: Buffer transmission is invalid 1: Buffer transfer enable (DTUBR->DTUAR)										R/W
b3~b1	Reserved		-		Read as "0", write as "0"										R/W
b0	DTCEN		Dead zone function		0: Dead zone function is invalid 1: The dead zone function is valid										R/W

18.5.11 Filter Control Register (TMR6_FCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								NOFI CKTB[1:0]	NOFI ENTB	-	NOFI CKTA[1:0]	NOFI ENTA			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								NOFI CKGB[1:0]	NOFI ENGB	-	NOFI CKGA[1:0]	NOFI ENGA			
Bit	Marking		Place name		Function										Read and write
b31~b23	Reserved		-	Read as "0", write as "0"										R/W	
b22~b21	NOFICKTB[1:0]		Filter sampling reference clock selection TB	00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64										R/W	
b20	NOFIENTB		Capture input port filter TB	0: TIM6_TRIGB port input filter function is invalid 1: TIM6_TRIGB port input filter function is enabled										R/W	
b19	Reserved		-	Read as "0", write as "0"										R/W	
b18~b17	NOFICKTA[1:0]		Filter sampling reference clock selection TA	00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64										R/W	
b16	NOFIENTA		Capture input port filter TA	0: TIM6_TRIGA port input filter function is invalid 1: TIM6_TRIGA port input filter function is enabled										R/W	
b15~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		Capture input port filter GB	0: The filter function of the TIM6_<t>_PWMB input port of this unit is invalid 1: The unit TIM6_<t>_PWMB input port filter function 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		Capture input port filter GA	0: The filter function of the TIM6_<t>_PWMA input port of this unit is invalid 1: The unit TIM6_<t>_PWMA input port filtering function is enabled										R/W	

18.5.12 Valid Period Register (TMR6_VPERR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved										PCNTS[2:0]	PCNTE[1:0]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved					SP PERIB	SP PERIA	Reserved									
<hr/>																
Bit	Marking	Place name	Function										Read and write			
b31~b21	Reserved	-	Read as "0", write as "0"										R/W			
b20~b18	PCNTS[2:0]	Valid period selection	000: Valid cycle selection function is invalid 001: Valid every 1 cycle 010: Valid every 2 cycles 011: Valid every 3 cycle 100: Valid every 4 cycle 101: Valid every 5 cycle 110: Valid every 6 cycle 111: Valid every 7 cycle										R/W			
b17~b16	PCNTE[1:0]	Active cycle count condition selection	00: Valid period selection function is invalid 01: The sawtooth wave counts the upper and lower overflow points or the triangular wave trough as the counting condition 10: The sawtooth wave counts the upper and lower overflow points or the triangular wave peak as the counting condition 11: The sawtooth wave counts the upper and lower overflow points or the triangular wave trough and peak as the counting condition										R/W			
b15~b10	Reserved	-	Read as "0", write as "0"										R/W			
b9	SPPERIB	Dedicated signal effective period selection B	0: Valid cycle selection function is invalid 1: Valid cycle selection function is enabled										R/W			
b8	SPPERIA	Special signal effective period selection A	0: Valid cycle selection function is invalid 1: Valid cycle selection function is enabled										R/W			
b7~b0	Reserved	-	Read as "0", write as "0"										R/W			

18.5.13 Status Flag Register (TMR6_STFLR)

Reset value: 0x8000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DIRF	Reserved						VPERNUM[2:0]			Reserved					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	CMS BDF	CMS BUF	CMS ADF	CMS AUF	DTE F	UDF F	OVF F	CMF F	CME F	CMD F	CMC F	CMB F	CMA F

Bit	Marking	Place name	Function	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 cycles	When the effective cycle selection function is enabled, the number of cycles after counting	R
b20~b13	Reserved	-	Read as "0", write as "0"	R
b12	CMSBDF	Count down special reference value match 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	Count up special reference value match 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	Count down special base value match 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	Up-Count-Specific Baseline Match 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: A dead time error has occurred	R
b7	UDFF	Underflow match	0: Sawtooth underflow does not occur or triangle wave counts to the valley point 1: A sawtooth wave underflow occurs or a triangular wave counts to a valley point	R/W
b6	OVFF	Overflow match	0: No sawtooth overflow or triangular wave counting to peak 1: Sawtooth overflow occurs or triangular wave counts to peak	R/W
b5	CMFF	Count 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	count 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	Count 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	Count 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	Count match B	0: The value of the GCMBR register is not equal to the count value, and the TIM6_<t>_PWMB capture completion action has not occurred 1: The value of the GCMBR register is equal to the count value, or a TIM6_<t>_PWMB capture completion action occurs	R/W
b0	CMAF	Count Match A	0: The value of the GCMAR register is not equal to the count value, and the TIM6_<t>_PWMA capture completion action has not occurred 1: The value of the GCMAR register is equal to the count value, or a TIM6_<t>_PWMA capture completion action occurs	R/W

18.5.14 Hardware Start Event Select Register (TMR6_HSTAR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
STA RTS	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HSTA 11	HSTA 10	HSTA 9	HSTA 8	HSTA 7	HSTA 6	HSTA 5	HSTA 4	-	-	HSTA 1	HSTA 0

Bit	Marking	Place name	Function	Read and write
b31	STARTS	Hardware enable	0: hardware startup is invalid 1: Hardware startup is valid Note: When the hardware boot is valid, the setting of SSTAR is invalid	R/W
b30~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HSTA11	Hardware startup condition 11	Condition: TIM6_TRIGB port is sampled to the falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W
b10	HSTA10	Hardware startup condition 10	Condition: TIM6_TRIGB port is sampled to the rising edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W
b9	HSTA9	Hardware startup condition 9	Condition: TIM6_TRIGA port is sampled to the falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W
b8	HSTA8	Hardware startup condition 8	Condition: TIM6_TRIGA port is sampled to the rising edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W
b7	HSTA7	Hardware startup condition 7	Condition: TIM6_<t>_PWMB port is sampled to the falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W
b6	HSTA6	Hardware startup condition 6	Condition: TIM6_<t>_PWMB port is sampled to the rising edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W
b5	HSTA5	Hardware startup condition 5	Condition: TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W
b4	HSTA4	Hardware startup condition 4	Condition: TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	HSTA1	Hardware startup condition 1	Condition: Internal Hardware Trigger Event 1Valid 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W
b0	HSTA0	Hardware startup condition 0	Condition: Internal Hardware Trigger Event 0Valid 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W

18.5.15 Hardware Stop Event Select Register (TMR6_HSTPR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
STOPS	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HSTP 11	HSTP 10	HSTP 9	HSTP 8	HSTP 7	HSTP 6	HSTP 5	HSTP 4	-	-	HSTP 1	HSTP 0

Bit	Marking	Place name	Function	Read and write
b31	STOPS	Hardware disable	0: hardware stop is invalid 1: Hardware stop is valid Note: When the hardware stop is valid, the setting of SSTPR is invalid	R/W
b30~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HSTP11	Hardware stop condition 11	Condition: TIM6_TRIGB port is sampled to the falling edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b10	HSTP10	Hardware stop condition 10	Condition: TIM6_TRIGB port is sampled to the rising edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b9	HSTP9	Hardware stop condition 9	Condition: TIM6_TRIGA port is sampled to the falling edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b8	HSTP8	Hardware stop condition 8	Condition: TIM6_TRIGA port is sampled to the rising edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b7	HSTP7	Hardware stop condition 7	Condition: TIM6_<t>_PWMB port is sampled to the falling edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b6	HSTP6	Hardware stop condition 6	Condition: TIM6_<t>_PWMB port is sampled to the rising edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b5	HSTP5	Hardware stop condition 5	Condition: TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b4	HSTP4	Hardware stop condition 4	Condition: TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	HSTP1	Hardware stop condition 1	Condition: Internal Hardware Trigger Event 1Valid 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b0	HSTP0	Hardware stop condition 0	Condition: Internal Hardware Trigger Event 0Valid 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W

18.5.16 Hardware Clear Event Select Register (TMR6_HCLRR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CLEAR	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HCLE 11	HCLE 10	HCLE 9	HCLE 8	HCLE 7	HCLE 6	HCLE 5	HCLE 4	-	-	HCLE 1	HCLE 0

Bit	Marking	Place name	Function	Read and write
b31	CLEAR	Hardware clear enable	0: Hardware reset is invalid 1: Hardware clear is valid Note: When the hardware clear is valid, the setting of SCLRR is invalid	R/W
b30~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HCLE11	Hardware zeroing condition 11	Condition: TIM6_TRIGB port is sampled to the falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b10	HCLE10	Hardware zeroing condition 10	Condition: TIM6_TRIGB port is sampled to the rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b9	HCLE9	Hardware zeroing condition 9	Condition: TIM6_TRIGA port is sampled to the falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b8	HCLE8	Hardware zeroing condition 8	Condition: TIM6_TRIGA port is sampled to the rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b7	HCLE7	Hardware zeroing condition 7	Condition: TIM6_<t>_PWMB port is sampled to the falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b6	HCLE6	Hardware zeroing condition 6	Condition: TIM6_<t>_PWMB port is sampled to the rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b5	HCLE5	Hardware zeroing condition 5	Condition: TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b4	HCLE4	Hardware zeroing condition 4	Condition: TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	HCLE1	Hardware zeroing condition 1	Condition: Internal Hardware Trigger Event 1Valid 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b0	HCLE0	Hardware zeroing condition 0	Condition: Internal Hardware Trigger Event 0Valid 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W

18.5.17 Hardware Capture Event Select Register (TMR6_HCPAR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HCP A11	HCP A10	HCP A9	HCP A8	HCP A7	HCP A6	HCP A5	HCP A4	-	-	HCP A1	HCP A0
Bit	Marking		Place name		Function										Read and write
b31~b12	Reserved		-		Read as "0", write as "0"										R/W
b11	HCPA11		Hardware Capture A Condition 11		Condition: TIM6_TRIGB port is sampled to the falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W
b10	HCPA10		Hardware Capture A Condition 10		Condition: TIM6_TRIGB port is sampled to the rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W
b9	HCPA9		Hardware Capture A Condition 9		Condition: TIM6_TRIGA port is sampled to the falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W
b8	HCPA8		Hardware Capture A Condition 8		Condition: TIM6_TRIGA port is sampled to the rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W
b7	HCPA7		Hardware Capture A Condition 7		Condition: TIM6_ t _PWMB port is sampled to the falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W
b6	HCPA6		Hardware Capture A Condition 6		Condition: TIM6_ t _PWMB port is sampled to the rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W
b5	HCPA5		Hardware Capture A Condition 5		Condition: TIM6_ t _PWMA port is sampled to the falling edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W
b4	HCPA4		Hardware Capture A Condition 4		Condition: TIM6_ t _PWMA port is sampled to the rising edge 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W
b3~b2	Reserved		-		Read as "0", write as "0"										R/W
b1	HCPA1		Hardware Capture A Condition 1		Condition: Internal Hardware Trigger Event 1 Valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W
b0	HCPA0		Hardware capture A condition 0		Condition: Internal Hardware Trigger Event 0 Valid 0: When the condition is matched, the hardware capture A is invalid 1: When the condition is matched, the hardware capture A is valid										R/W

18.5.18 Hardware Capture Event Select Register (TMR6_HCPBR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HCPB11	HCPB10	HCPB9	HCPB8	HCPB7	HCPB6	HCPB5	HCPB4	-	-	HCPB1	HCPB0
Bit	Marking	Place name	Function												Read and write
b31~b12	Reserved	-	Read as "0", write as "0"												R/W
b11	HCPB11	Hardware Capture B Condition 11	Condition: TIM6_TRIGB port is sampled to the falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W
b10	HCPB10	Hardware Capture B Condition 10	Condition: TIM6_TRIGB port is sampled to the rising edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W
b9	HCPB9	Hardware Capture B Condition 9	Condition: TIM6_TRIGA port is sampled to the falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W
b8	HCPB8	Hardware Capture B Condition 8	Condition: TIM6_TRIGA port is sampled to the rising edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W
b7	HCPB7	Hardware Capture B Condition 7	Condition: TIM6_ t _PWMB port is sampled to the falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W
b6	HCPB6	Hardware Capture B Condition 6	Condition: TIM6_ t _PWMB port is sampled to the rising edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W
b5	HCPB5	Hardware Capture B Condition 5	Condition: TIM6_ t _PWMA port is sampled to the falling edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W
b4	HCPB4	Hardware Capture B Condition 4	Condition: TIM6_ t _PWMA port is sampled to the rising edge 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W
b3~b2	Reserved	-	Read as "0", write as "0"												R/W
b1	HCPB1	Hardware Capture B Condition 1	Condition: Internal Hardware Trigger Event 1Valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W
b0	HCPB0	Hardware capture B condition 0	Condition: Internal Hardware Trigger Event0Valid 0: When the condition is matched, the hardware capture B is invalid 1: When the condition is matched, the hardware capture B is valid												R/W

18.5.19 Hardware Increment Event Select Register (TMR6_HCUPR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved															HC UP17	HC UP16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
-	-	-	-	HC UP11	HC UP10	HC UP9	HC UP8	HC UP7	HC UP6	HC UP5	HC UP4	HC UP3	HC UP2	HC UP1	HC UP0	
Bit	Marking		Place name			Function										Read and write
b31~b18	Reserved		-			Read as "0", write as "0"										R/W
b17	HCUP17		Hardware Added Condition 17			Condition: Internal Hardware Trigger Event 1Valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b16	HCUP16		Hardware Added Condition 16			Condition: Internal Hardware Trigger Event 0Valid 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b15~b12	Reserved		-			Read as "0", write as "0"										R/W
b11	HCUP11		Hardware Added Conditions11			Condition: TIM6_TRIGB port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b10	HCUP10		Hardware Added Condition 10			Condition: TIM6_TRIGB port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b9	HCUP9		Hardware Added Condition 9			Condition: TIM6_TRIGA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b8	HCUP8		Hardware Added Condition 8			Condition: TIM6_TRIGA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b7	HCUP7		Hardware Added Condition 7			Condition: When the TIM6_<t>_PWMB port is high level, the TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b6	HCUP6		Hardware Added Condition 6			Condition: When the TIM6_<t>_PWMB port is high, the TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b5	HCUP5		Hardware Added Condition 5			Condition: When the TIM6_<t>_PWMB port is low level, the TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b4	HCUP4		Hardware Added Condition 4			Condition: When the TIM6_<t>_PWMB port is low, the TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b3	HCUP3		Hardware Added Condition 3			Condition: When the TIM6_<t>_PWMB port is high, the TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b2	HCUP2		Hardware Added Condition 2			Condition: When the TIM6_<t>_PWMB port is low, the TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b1	HCUP1		Hardware Added Condition 1			Condition: When the TIM6_<t>_PWMB port is high, the TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W
b0	HCUP0		Hardware Added Condition 0			Condition: When the TIM6_<t>_PWMB port is low level, the TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches										R/W

18.5.20 Hardware Decrement Event Select Register (TMR6_HCDOR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved														HC DO17	HC DO16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
				HC DO11	HC DO10	HC DO9	HC DO8	HC DO7	HC DO6	HC DO5	HC DO4	HC DO3	HC DO2	HC DO1	HC DO0

Bit	Marking	Place name	Function	Read and write
b31~b18	Reserved	-	Read as "0", write as "0"	R/W
b17	HCDO17	Hardware decrement condition 17	Condition: Internal Hardware Trigger Event 1Valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b16	HCDO16	Hardware decrement condition 16	Condition: Internal Hardware Trigger Event0Valid 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HCDO11	Hardware decrement condition11	Condition: TIM6_TRIGB port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b10	HCDO10	Hardware decrement condition 10	Condition: TIM6_TRIGB port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b9	HCDO9	Hardware decrement condition 9	Condition: TIM6_TRIGA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b8	HCDO8	Hardware decrement condition 8	Condition: TIM6_TRIGA port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b7	HCDO7	Hardware decrement condition 7	Condition: When the TIM6_<t>_PWMB port is high level, the TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b6	HCDO6	Hardware decrement condition 6	Condition: When the TIM6_<t>_PWMB port is high, the TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b5	HCDO5	Hardware decrement condition 5	Condition: When the TIM6_<t>_PWMB port is low level, the TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b4	HCDO4	Hardware decrement condition 4	Condition: When the TIM6_<t>_PWMB port is low, the TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b3	HCDO3	Hardware decrement condition 3	Condition: When the TIM6_<t>_PWMB port is high, the TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b2	HCDO2	Hardware decrement condition 2	Condition: When the TIM6_<t>_PWMB port is high level, the TIM6_<t>_PWMA port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W
b1	HCDO1	Hardware decrement condition 1	Condition: When the TIM6_<t>_PWMB port is low level, the TIM6_<t>_PWMA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match	R/W

match			
b0	HCDO0	Hardware decrement condition 0	Condition: When the TIM6 <math><t>_PWMA</math> port is low level, the TIM6 <math><t>_PWMB</math> port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match

18.5.21 Software Synchronous Start Control Register (TMR6_SSTAR)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	SSTA3	Unit 3 Software Startup	0: Software startup is invalid 1: Software startup enable	R/W
b1	SSTA2	Unit 2 Software Startup	0: Software startup is invalid 1: Software startup enable	R/W
b0	SSTA1	Unit 1 software start	0: Software startup is invalid 1: Software startup enable	R/W

18.5.22 Software Synchronous Stop Control Register (TMR6_SSTPR)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	SSTP3	Unit 3 Software Stop	0: Software stop is invalid 1: Software stop enable	R/W
b1	SSTP2	Unit 2 Software Stop	0: Software stop is invalid 1: Software stop enable	R/W
b0	SSTP1	Unit 1 software stop	0: Software stop is invalid 1: Software stop enable	R/W

18.5.23 Software Synchronous Clear Control Register (TMR6_SCLRR)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	SCLE3	Unit 3 software reset	0: Software reset is invalid 1: Software clear enable	R/W
b1	SCLE2	Unit 2 software reset	0: Software reset is invalid 1: Software clear enable	R/W
b0	SCLE1	Unit 1 software clear	0: Software reset is invalid 1: Software clear enable	R/W

19 General Control Timer (Timer4)

19.1 Introduction

General control timer 4 (Timer4) is a timer module for three-phase motor control. It provides three-phase motor control schemes for various applications. The timer supports triangular wave and sawtooth wave modes and generates various PWM waveforms. Supporting cache function; Support EMB control. 3 unit of Timer 4 is carried in this product family.

19.2 Basic Block Diagram

The basic functions and characteristics of Timer4 are shown in Table 19-1.

Table 19-1 Basic Functions And Features Of Timer4

Waveform mode	Sawtooth wave, triangular wave
Basic functions	• Direction of increments and decrements
	• Cache function
	• General PWM output
	• Dedicated event output
	• EMB control
Interrupt type	Count comparison match interrupt
	Count cycle match interrupt
	Heavy load count matching interrupt

As shown in Figure 19-1, the basic structure of the general control timer Timer4 is described. As shown in the block diagram, " $< t >$ " represents the unit number, that is, " $< t >$ " is 1~3. This section refers to " $< t >$ " when referring to " $< t >$ " later.

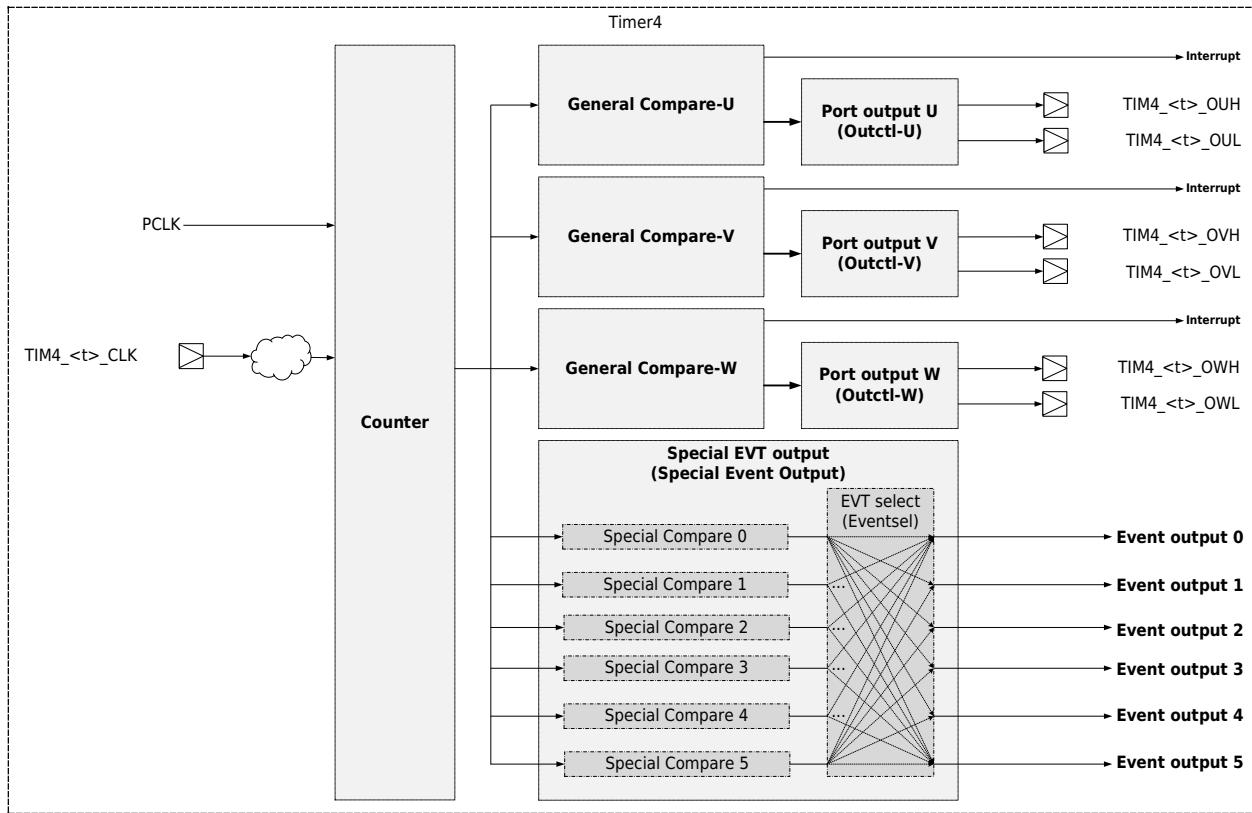


Figure 19-1 Timer4 Basic Block Diagram

Table 19-2 shows the list of input and output ports of Timer4.

Table 19-2 Timer4 Port List

Port name	Direction	Function
TIM4_<t>_CLK	in	Counting clock input port
TIM4_<t>_OUH		
TIM4_<t>_OUL		
TIM4_<t>_OVH		
TIM4_<t>_OVL		
TIM4_<t>_OWH		
TIM4_<t>_OWL		

19.3 Functional Description

19.3.1 Basic Action

19.3.1.1 Waveform Mode

Timer4 has two basic counting wave modes: Serrated wave mode and triangular wave mode. According to CCSR.MODE, two waveform modes can be implemented. Figure 19-2 and Figure 19-3 shows the waveform diagrams of sawtooth wave and triangular wave respectively.

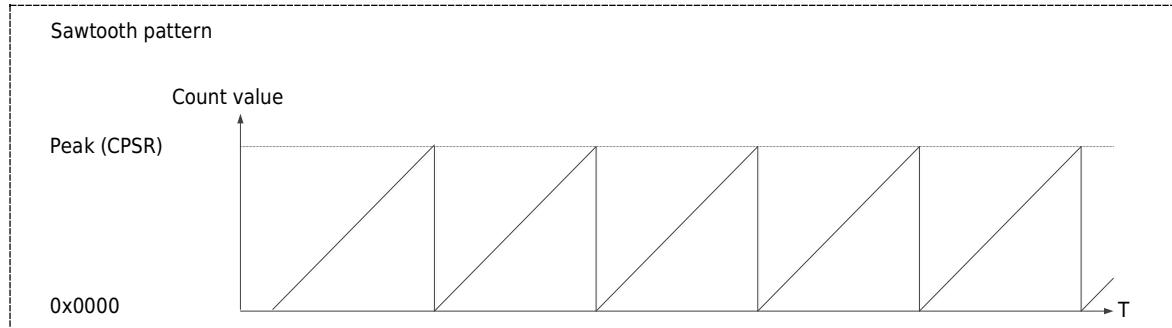


Figure 19-2 Timer4 Sawtooth Waveform

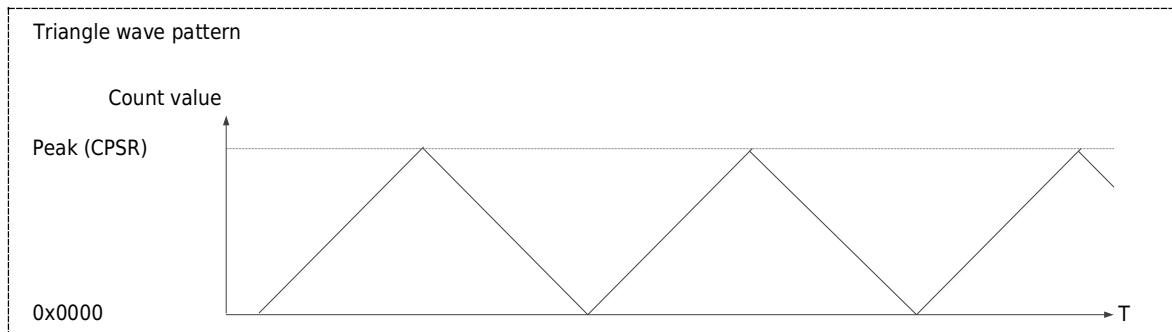


Figure 19-3 Timer4 Triangle Wave Waveform

19.3.1.2 Counting Action

- 1) The sawtooth wave counting operation and control flow are as Figure 19-4 shown.

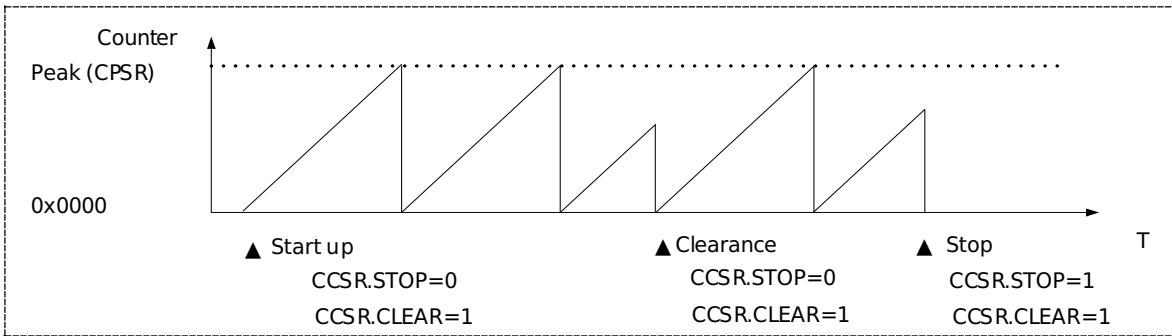


Figure 19-4 Timer4 sawtooth wave mode counting action

- a) Set mode CCSR.MODE = 0.
- b) Sets the count peak CPSRRegister.
- c) Write CCSR.STOP = 0 and CCSR.CLEAR = 1, and the counter count value (CNTR) is

initialized to 0x0000 and the counting operation is initiated. Counter values are counted incrementally from 0x0000. When the peak value (CPSR) is reached and the count value is returned to 0x0000, the operation is repeated in turn.

- d) Counting cycle = $(CPSR + 1) \times$ Counting clock cycle
 - e) During counting, write CCSR.STOP = 0 and CCSR.CLEAR = 1 to initialize the count value to 0x0000 and continue counting operations.
 - f) During counting, write CCSR.STOP = 1 and CCSR.CLEAR = 1 to initialize the count value to 0x0000 and stop the counting operation.
- 2) The triangular wave counting operation and control flow are as Figure 19-5 shown.

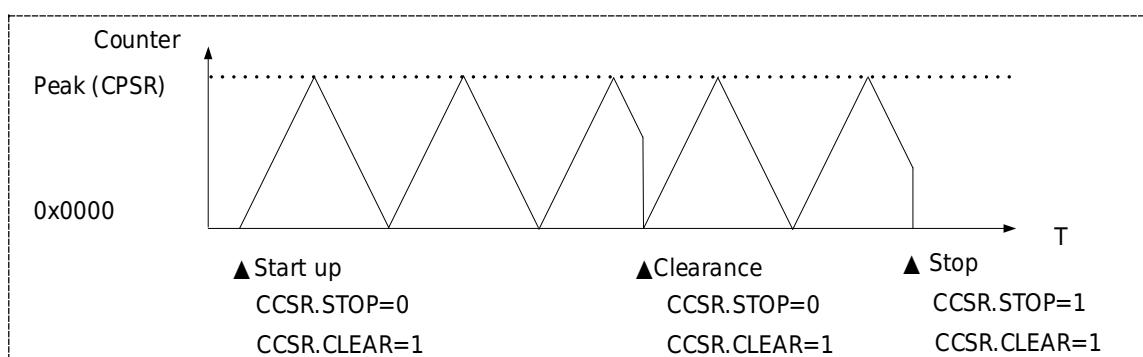


Figure 19-5 Timer4 Triangular Wave Mode Counting Action

- a) Set mode CCSR.MODE = 1.
- b) Sets the count peak CPSRRegister.
- c) Write CCSR.STOP = 0 and CCSR.CLEAR = 1, and the counter count value (CNTR) is initialized to 0x0000 and the counting operation is initiated. Counter values are counted incrementally from 0x0000, before reaching the peak. When the peak value (CPSR) is reached, the counter begins to count down until the count value is returned to 0x0000; The incremental counting operation is then repeated in turn.
- d) Counting cycle = $(CPSR) \times 2 \times$ Clock cycle
- e) During the counting process, write CCSR.STOP=0 and CCSR.CLEAR=1 to initialize the counting value to 0x0000 and restart the counting up operation, and then repeat the above operation.
- f) During counting, write CCSR.STOP = 1 and CCSR.CLEAR = 1 to initialize the count value to 0x0000 and stop the counting operation.

19.3.1.3 Comparison Output

- The following Figure 19-6 is an example of the waveform output of the comparison output module in sawtooth wave count mode.

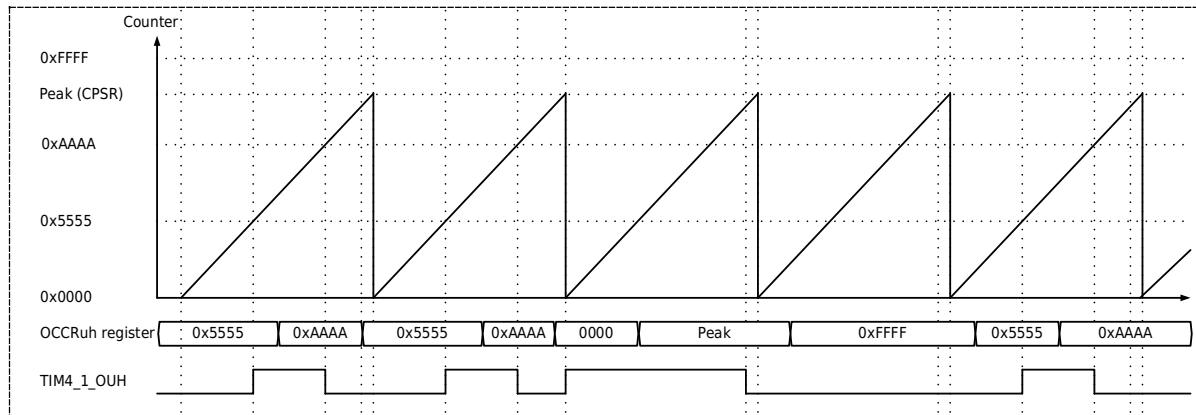


Figure 19-6 Waveform Output Example in Sawtooth Mode

- The following Figure 19-7 is an example of the waveform output of the comparison output module in the triangle wave count mode.

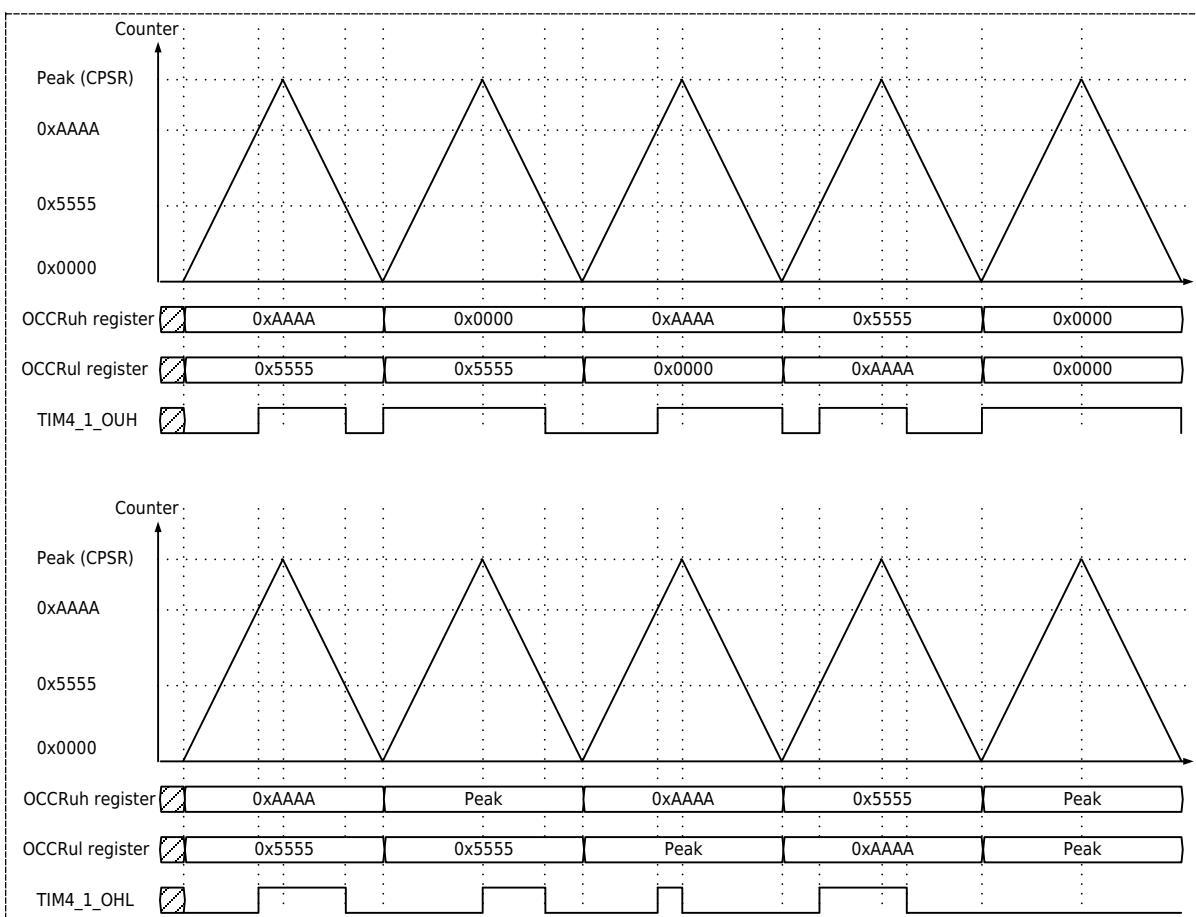


Figure 19-7 Triangle Wave Mode Waveform Output Example

19.3.2 Cache Function

Timer4's periodic reference register (CPSR), generic reference register (OCCR), generic mode control register (OCMR), dedicated reference register (SCCR) and dedicated mode control register (SCMR) all have caching capabilities.

19.3.2.1 Periodic Reference Register Cache Function

The CPSR has a cache function register, and the written count peak data is first stored in the buffer register. Data is transferred from the buffer register to the CPSRregister under the following conditions.

- When buffer is disabled (CCSR.BUFEN = 0), write data is immediately transferred from buffer register to CPSRregister.
- Data is transferred from the buffer register to the CPSRregister when the buffer function enables (CCSR.BUFEN = 1), when the counter stops (CCSR.STOP = 1), or when the counter count value is "0x0000".

Note:

- When reading data from a 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.

As shown in Figure 19-8, the operation of modifying the count peak CPSR when sawtooth mode and the buffer function is disabled.

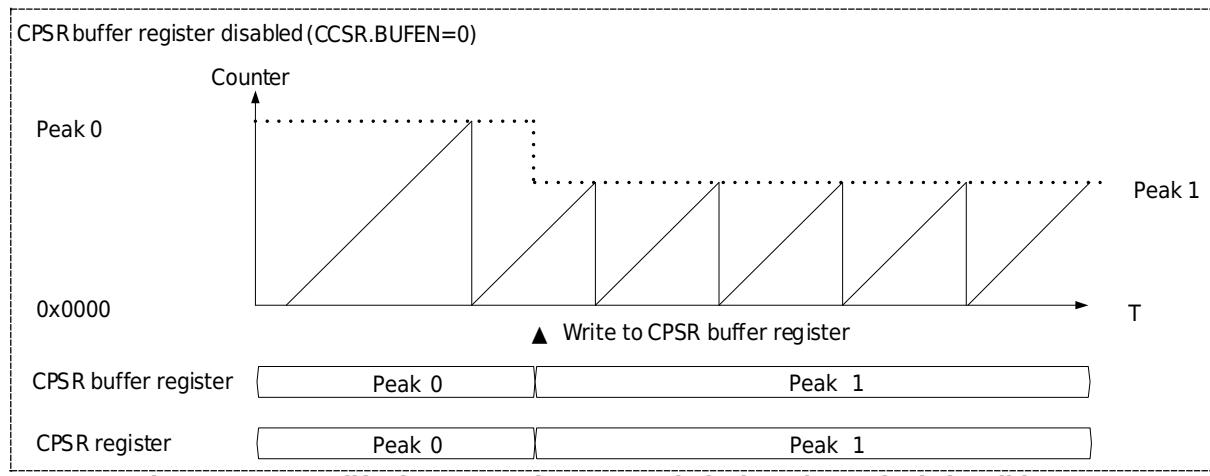


Figure 19-8 Modify the sawtooth count period when the cache is invalid

Note:

- When the buffer function is disabled, write data is immediately transferred from the buffer register to the CPSRcounter, and the count period changes immediately after the write operation is completed. In this case, if the value written is less than the current counter count, the counter continues to count incrementally until it reaches "0xFFFF", which requires special attention.

As shown in Figure 19-9, the operation of modifying the count peak value CPSR when the sawtooth mode is enabled and the buffer function is enabled.

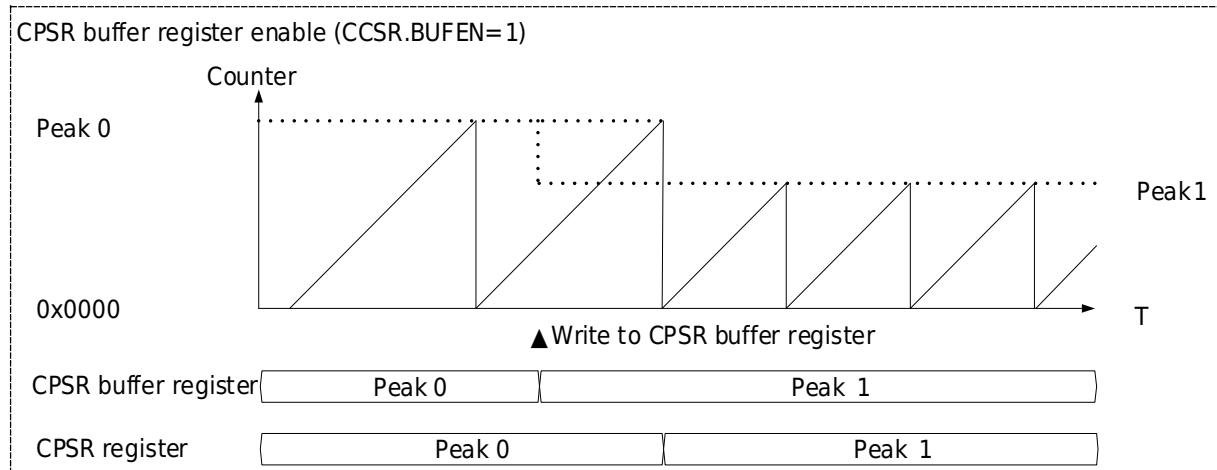


Figure 19-9 Modify the sawtooth count period when the cache is enabled

As shown in the figure, when the buffer function is enabled, the data written is transferred from the buffer register to the CPSRregister when the counter stops or when the counter count value is "0x0000".

As shown in Figure 19-10, the operation of modifying the count peak CPSR when the triangle wave mode and the buffer function is enabled.

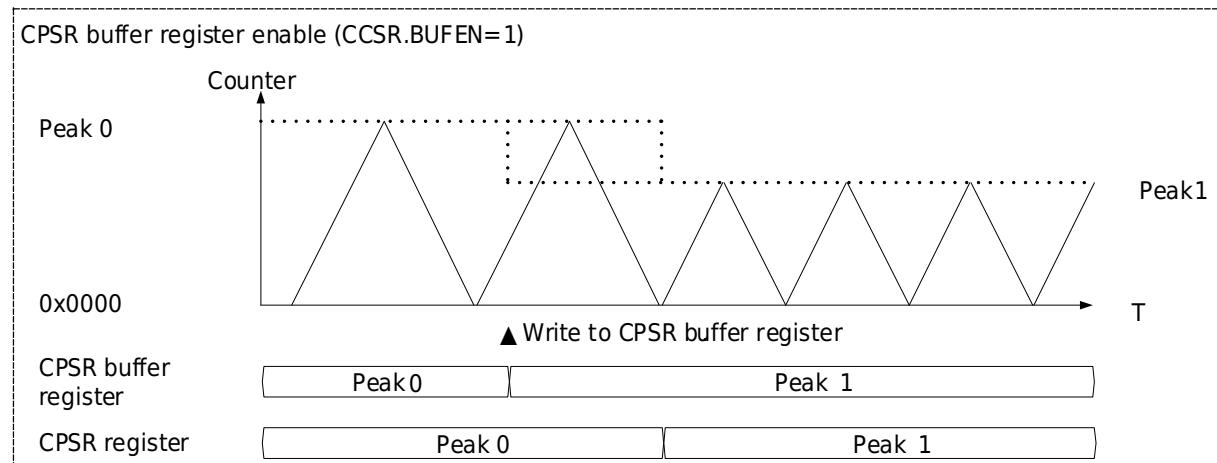


Figure 19-10 Modify The Triangular Wave Count Period When The Cache Is Enabled

As shown in the figure, in triangulation mode, when the buffer function is enabled, the written data is transferred from the buffer register to the CPSRregister when the counter stops or the next counter count value is "0x0000". The change in the counter cycle begins after the write operation completes the next counter cycle.

19.3.2.2 General Comparison Register Cache Function

Both the common baseline value register (OCCR) and the common mode control register (OCMR) have the buffer register function. When the buffer function is enabled, the transfer is loaded to the OCCR and OCMRregister at the specified transmission time. The OCCR buffer function can be used

to change the comparison value synchronously during counting, and the OCMR buffer function can be used to change the internal PWM output synchronously during counting.

- a) When the output compares the link transfer barring with the counter interval response function, the buffer value is loaded to register at the set count state.
The load at this time is independent of the counter interval counter.

Figure 19-11 OCCR Buffered Data Transfer (Periodic Interval Response Link Disabled) shows the waveform ($x = L$ or H) when the general output comparison OCCR buffer function is enabled, the counter is loaded at zero (OCER.CxBUFEN = 01), and the counter interval response link is disabled (OCER.LMCx = 0).

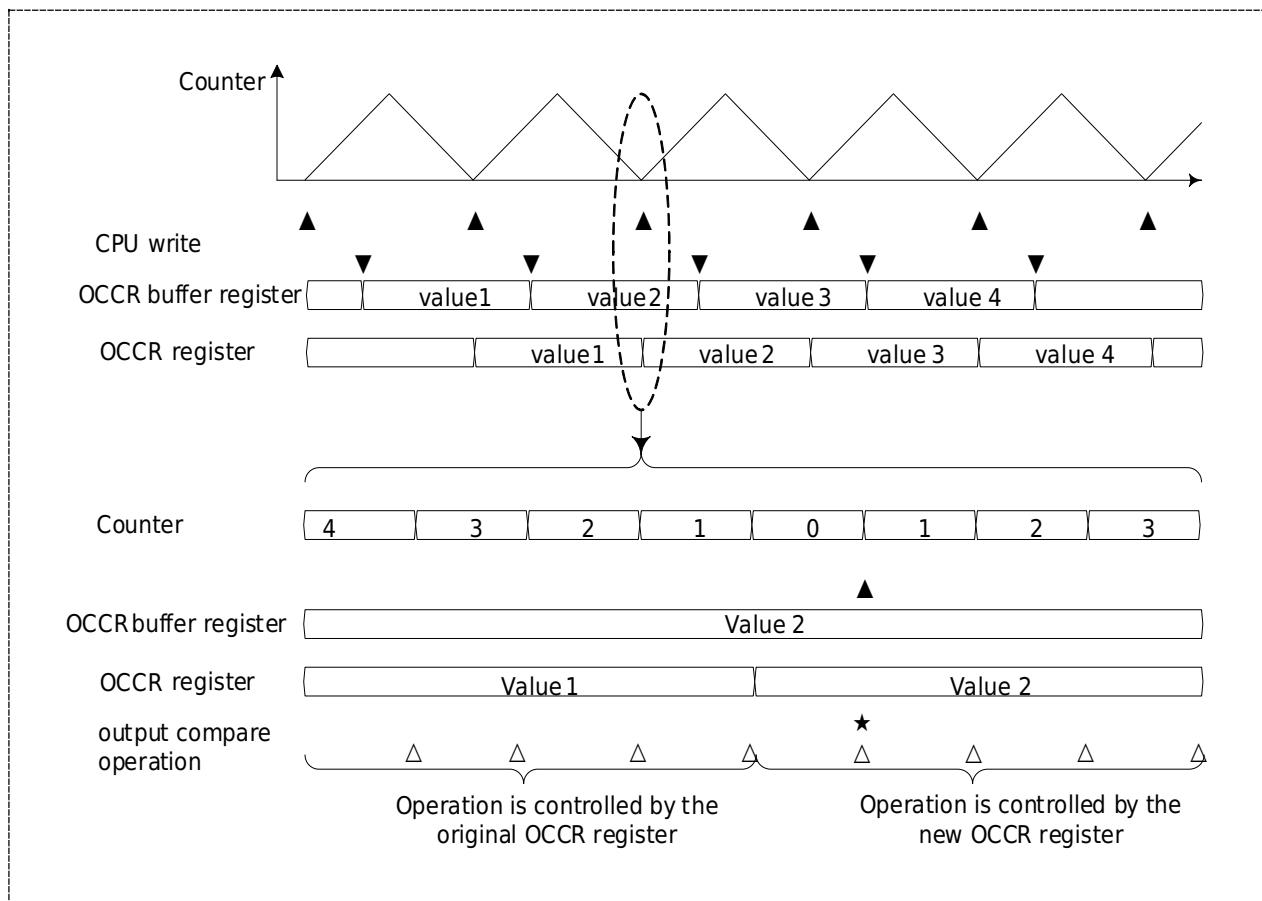


Figure 19-11 OCCR Buffer Data Transmission (When The Cycle Interval Response Link Is Disabled)

The first half of the diagram is a global diagram, and the second half is an enlarged diagram during the transfer operation.

The counter is in the triangular wave counting mode, and the underflow interrupt is generated at the moment of the mark ▲ (counting valley point). At ▼ time, the CPU rewrites the OCCR register, and the written data is stored in the OCCR buffer register. After that, when the counter count is 0x0000, the data is loaded from the buffer register to the OCCR register, and the interrupt flag IRQZF is generated.

At the Δ moment, the output comparison is performed to change the PWM output and the 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 = 0x0000), the port output performs the operation based on the new OCCR data. Perform the operation according to the original OCCR data before the time.

The figure shows the OCCR buffer register transfer operation at the counting valley point, and the OCMR buffer register transfer operation is similar. Similarly, the transfer operation at the counting peak is similar. The new data takes effect immediately after the transmission time (the new write data will control the PWM output and interrupt flag).

b) When the counter interval response link is enabled, the buffer register transfer operation is performed when the buffer value is in the set count state and the period interval counter count value is 0.

Figure 19-12 OCCR Buffered Data Transfer (Periodic Interval Response Link Enabled) shows the waveform ($x = L$ or H) when the general output compares the OCCR buffer function enabled, the counter zero value loaded (OCER.CxBUFEN = 01), and the counter interval response link enabled (OCER.LMCx = 1).

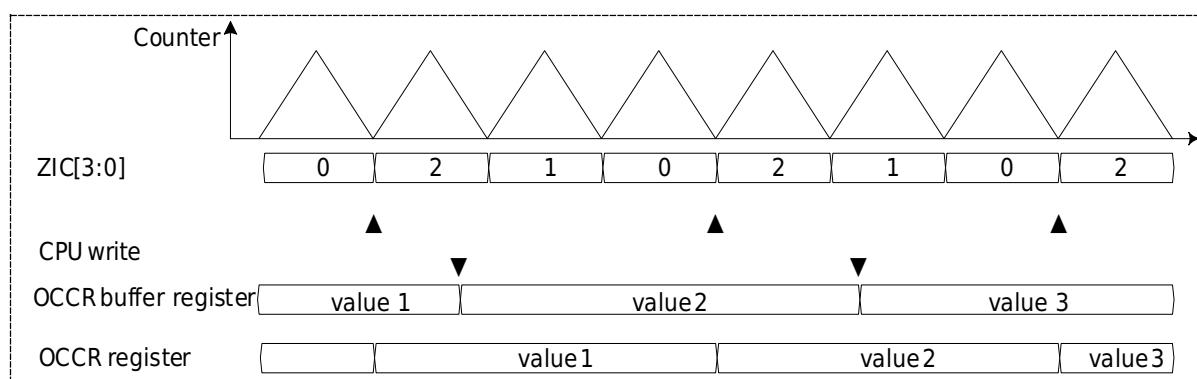


Figure 19-12 OCCR Buffer Data Transmission (Period Interval Response Link Enable)

The counter is in the triangular wave counting mode, the cycle interval counter (CVPR.ZIC) counts down from 2 to 0, and the underflow interrupt is generated at the moment of the mark Δ (counting valley point). At \blacktriangledown time, the CPU rewrites the OCCR register, and the written data is stored in the OCCR buffer register. After that, when the counter count value is 0x0000 and the period interval 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 figure shows the OCCR buffer register transfer operation at the counting valley point, and the OCMR buffer register transfer operation is similar. Similarly, the transfer operation at the counting peak is similar. The new data takes effect immediately after the transmission time (the new write data will control the PWM output and interrupt flag).

When using channel link operation mode, enabling both OCCRuh and OCCRul buffers can produce various PWM output waveforms.

As shown in Figure 19-13, The case of changing the OCMRregister value to produce different output waveforms, TIM4_< t >_OUL, when the output comparison registerOCCRuh and OCCRul remains unchanged is illustrated in Output comparison buffer data transfer (OCMR buffer enable).

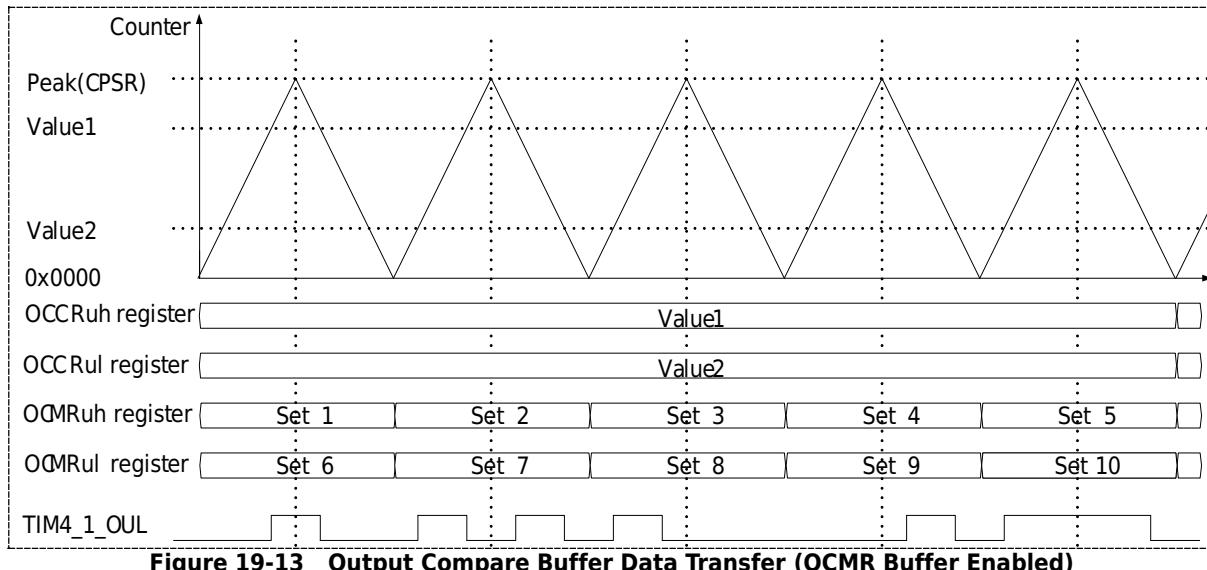


Figure 19-13 Output Compare Buffer Data Transfer (OCMR Buffer Enabled)

19.3.2.3 Special comparison register cache function

Both the dedicated reference register (SCCR) and the dedicated mode control register (SCMR) have a buffer work register. When the buffer function is enabled, the CPU writes the values of the SCCR and SCMR buffer registers to the SCCR and SCMR registers in the set counter state.

- a) **When the counter interval response link transfer is disabled, the buffer transfer operation is only related to the counter state and is not affected by the counter interval counter.**

As shown in Figure 19-14, it enables the register buffering function to disable the counter periodic interval response link transmission (SCSR. LMC=0), and transfers the counter zero value (SCSR. BUFEN=01) to the SCCR and SCMR registers.

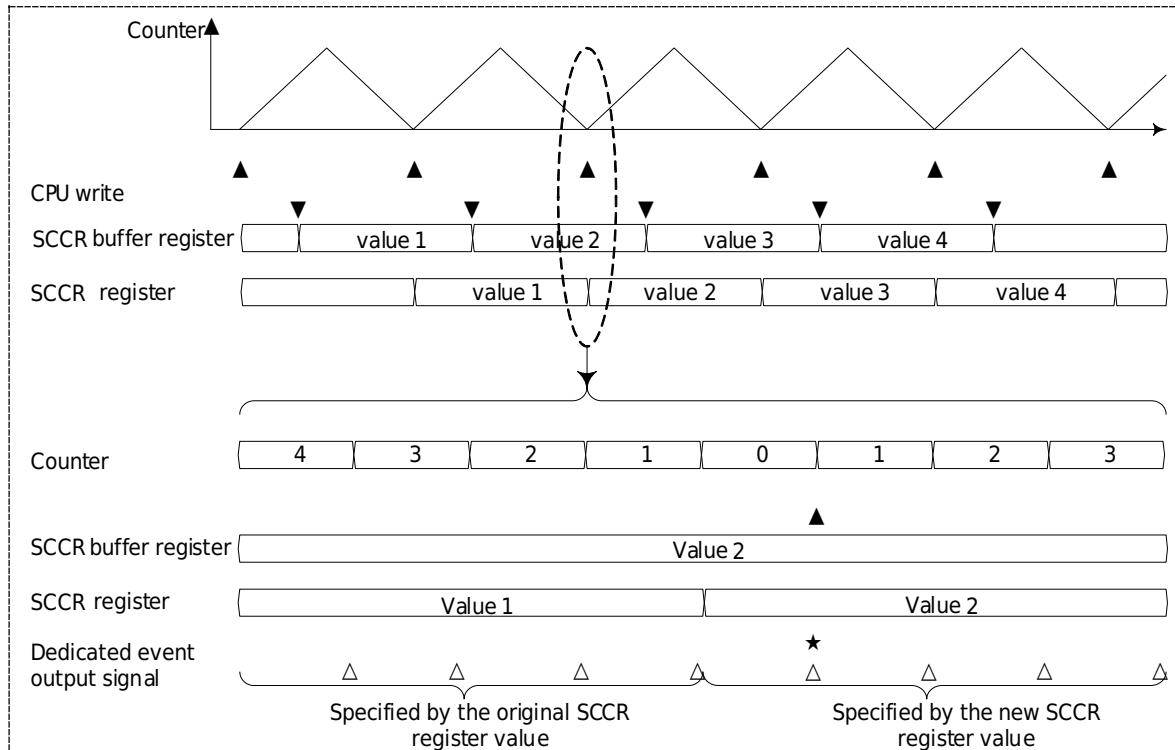


Figure 19-14 SCCR Buffer Transmission Operation (Period Interval Response Link Transmission Disabled)

The upper half of the diagram is a global diagram, and the lower half of the diagram is a partial magnification of the buffer register transfer operation.

The counter is in the triangular wave counting mode, and the underflow interrupt is generated at the moment of the mark ▲ (counting valley point). At ▼ time, the CPU rewrites the SCCR register, and the written data is stored in the SCCR buffer register. After that, when the counter count is 0x0000, the data loading operation is performed, from the buffer register to the SCCR register, and the interrupt flag IRQZF is generated.

At Δ time, the comparison operation is performed according to the set SCCR register value and the count value. After the time (count = 0x0000), the dedicated event output signal performs the operation based on the new SCCR data. Perform the operation according to the SCCR data before the time.

The diagram shows the SCCR buffer register transfer operation at the counting valley point, and the SCMR buffer register transfer operation is similar to the SCCR buffer register transfer operation. Similarly, the transfer operation at the counting peak is similar. The new data takes effect immediately after the transmission time (the new write data sets the dedicated event output signal and interrupt flag).

- b) When the counter interval response link is enabled, the buffer register transfer operation is performed when the buffer value is in the set count state and the period interval counter count value is 0.**

Figure 19-15 SCCR Buffer Transfer Operation (Periodic Interval Response Link Transfer Enabled) shows a schematic diagram of enabling SCCR buffering, loading at zero counter values (SCSR.BUFEN = 01), and enabling transmissions across response links during the counter cycle (SCSR.LMC = 1).

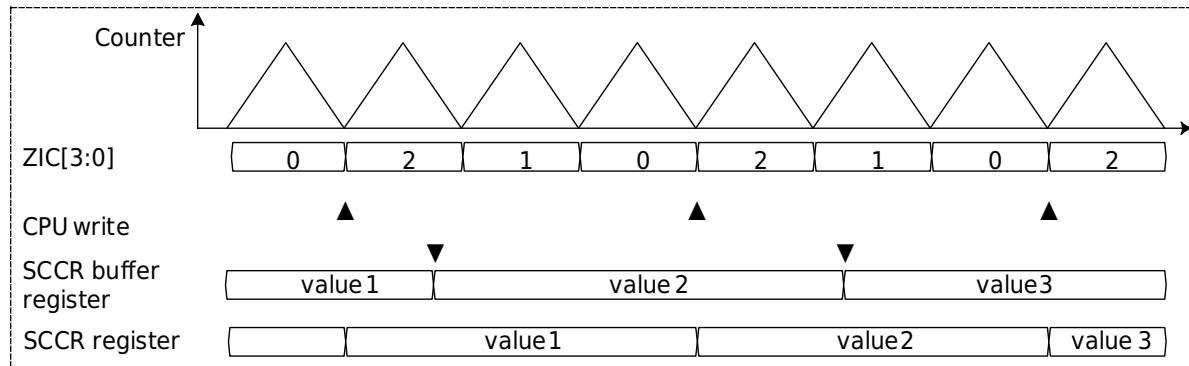


Figure 19-15 SCCR Buffer Transmission Operation (When Periodic Interval Response Link Transmission Is Enabled)

The counter is in the triangular wave counting mode, the cycle interval counter (CVPR.ZIC) counts down from 2 to 0, and the underflow interrupt is generated at the moment of the mark ▲ (counting valley point). At ▼ time, 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 period interval counter (CVPR.ZIC) is 0, the data is loaded from the buffer register to the SCCR register, and the interrupt flag IRQZF is generated.

19.3.3 General PWM output

19.3.3.1 Independent PWM output

The output states of OCCRxh, OCCRxl and OCMRxh and OCMLxl ports ($x = u, v, w$) can be set respectively in the pass-through mode (POCR.PWMMD = 00). At this point, the PWM output of each port is independently controlled. Figure 19-16 and Figure 19-17 are examples of independent PWM outputs for unit 1's sawtooth and triangle waves, respectively.

Note:

- The pass-through mode refers to the internal output signal (in_opxh, in_opxl) generated by comparing the values of the common reference registers (OCCRxh, OCCRxl), which is directly output to the corresponding ports (TIM4_<t>_OXH, TIM4_<t>_OXL) ($X = U, V, W, x = u, v, w$).

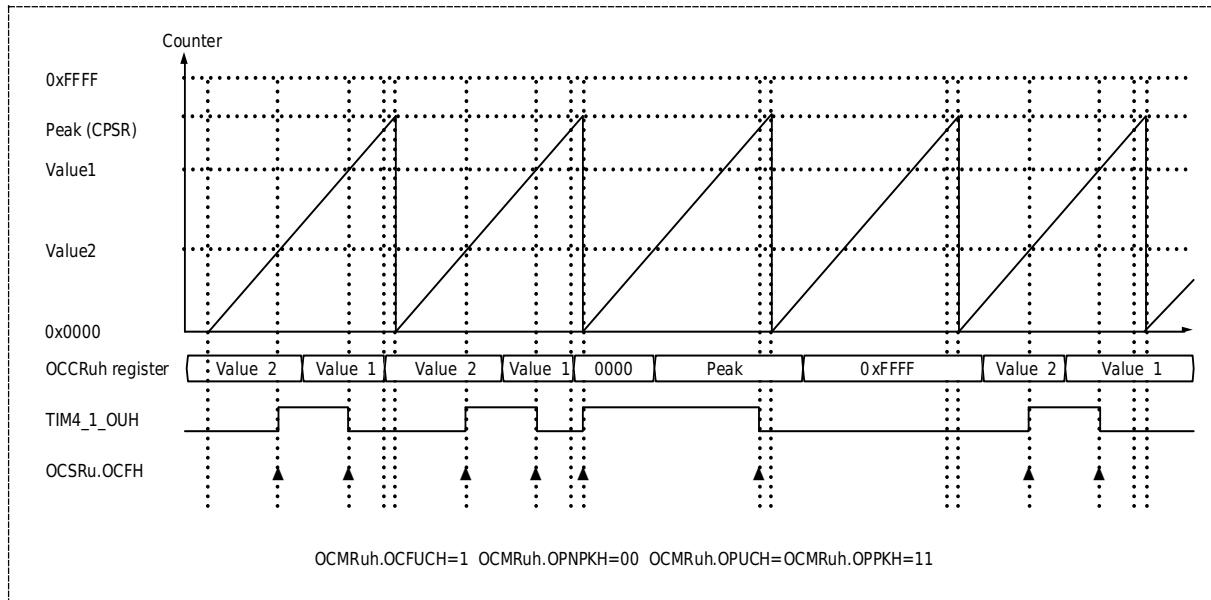


Figure 19-16 Sawtooth Wave Independent PWM Output Example

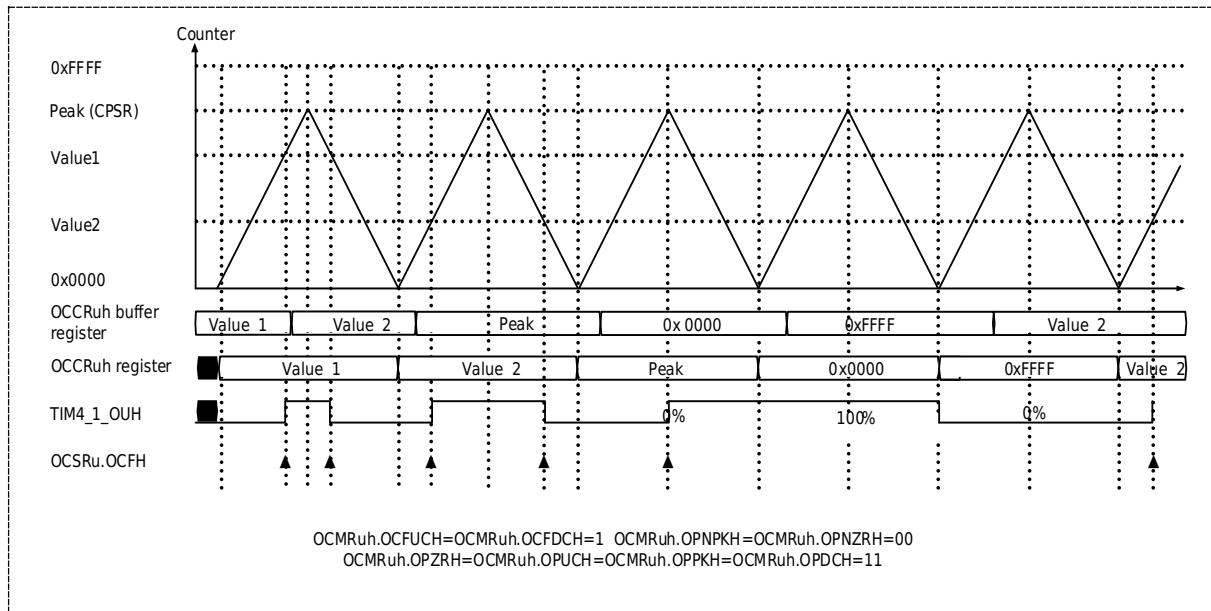


Figure 19-17 Triangle Wave Independent PWM Output Example

19.3.3.2 Extended PWM Output

The output state of TIM4_< t >_OXL in the pass-through mode (POCR.PWMMD = 00) can also be determined by the extension bit (bit32 ~ 16) in the OCMRxIregister. The extension bit is set to be correlated with the reference value of OCCRxh, thus realizing the extension PWM output (X = U, V, W, x = u, v, w) on the TIM4_< t >_OXL port. Figure 19-18 Shown is the PWM output of TIM4_<t>_OEH and TIM4_<t>_OUL ports in this mode.

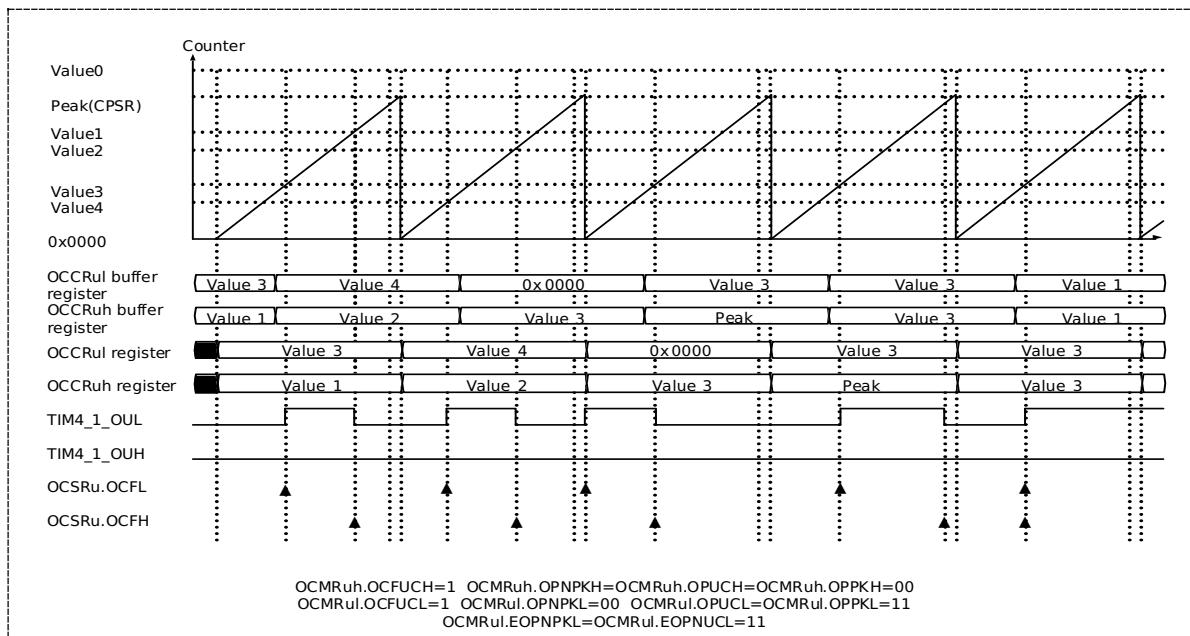


Figure 19-18 Sawtooth Wave Extended PWM Output

Note:

- Independent PWM output mode, the port state of TIM4_< t >_OXL is determined by bit15 ~ bit4 bits of OCMRxIregister, which is only related to the reference value of OCCRxI (X = U, V, W, x = u, v, w).

19.3.3.3 Complementary PWM Output

In the pass-through mode (POCR.PWMMD = 00), the reference values of OCCRxh and OCCRxl ($x = u, v, w$) are set directly to realize the output of a pair of complementary PWM waveforms to ports, and the three groups of ports can be set in the same way to realize the output of 3-phase complementary PWM, as shown in Figure 17-19 Software Implementation of Complementary PWM Output. As shown in Figure 19-19.

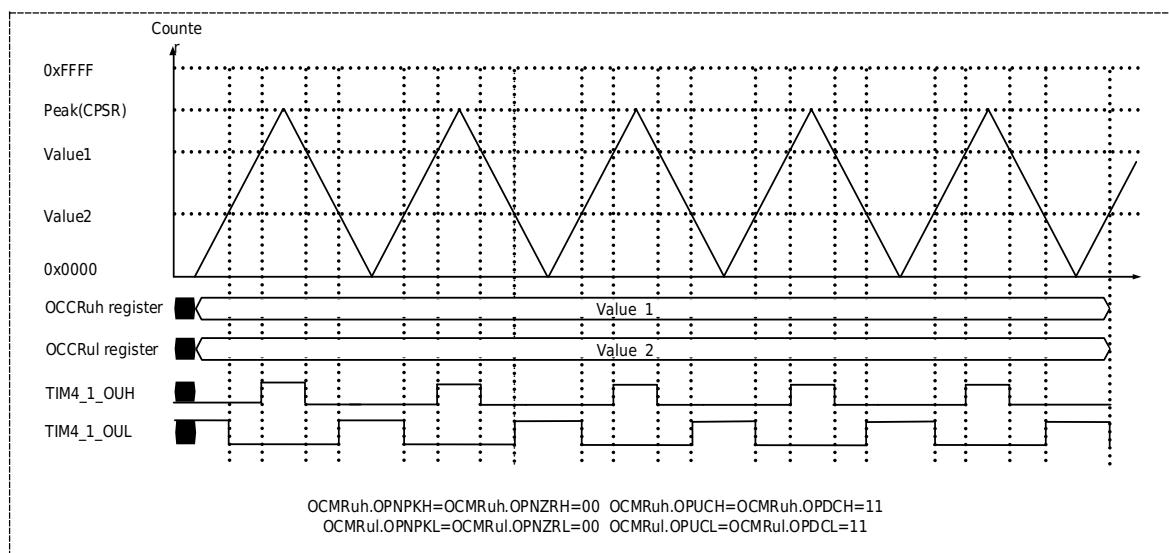


Figure 19-19 Software Complementary PWM Output

Realization of Complementary PWM Output by Hardware Setting

In the dead-zone timer mode (POCR.PWMMD = 01), the values of common reference register (OCCRxl) are matched to the internal output signal (in _ opxl) and PWM dead-zone control register (PDAR/PDBR).

In this mode, the polarity of $\text{TIM4}_{< t >}_{\text{OXH}}$ port output is the same as in _ opxl, and the polarity of $\text{TIM4}_{< t >}_{\text{OXL}}$ port output is opposite to in _ opxl ($X = U, V, W, x = u, v, w$).

Figure 19-20 Shown is an example of complementary PWM output in dead-band timer mode.

If the rising edge of in _ opxl is detected, the output of $\text{TIM4}_{< t >}_{\text{OXL}}$ becomes low, the dead zone counter loads the set value of PDBRxregister and starts the decrement count. When the count value becomes 0x0000, the counter stops and $\text{TIM4}_{< t >}_{\text{OXH}}$ outputs high level. If an in _ opxl falling edge is detected and the $\text{TIM4}_{< t >}_{\text{OXH}}$ output becomes low, the dead-zone counter loads the set value of the PDARxregister and starts the decrement count. When the count value becomes 0x0000, the counter stops and the $\text{TIM4}_{< t >}_{\text{OXL}}$ output is high ($X = U, V, W, x = u, v, w$).

By setting PWM dead zone to control registerPDAR and PDBR, output rising and falling dead zone time can be set accordingly.

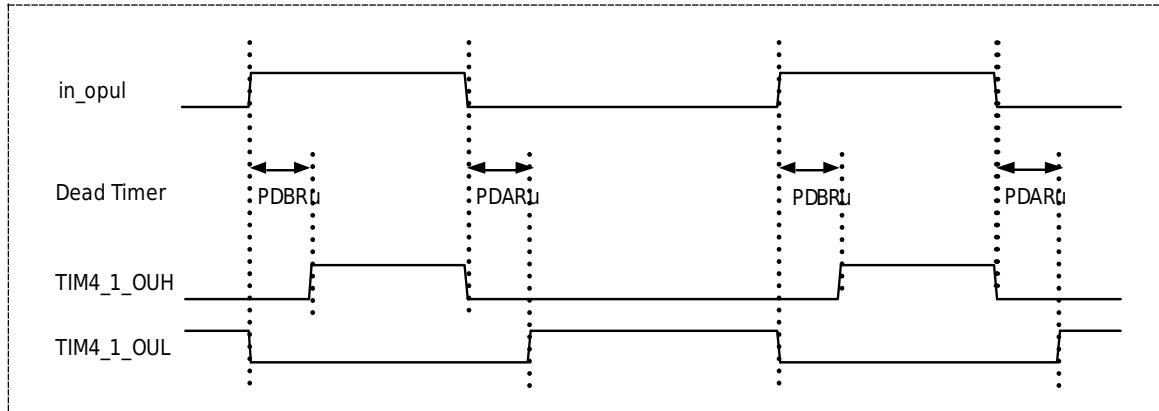


Figure 19-20 Complementary PWM output in dead-band timer mode

When the pulse width of in _ opxl is less than the dead time set by PDBR, only the output of TIM4 _ < t > _ OXL becomes low. The condition for the output level of TIM4 _ < t > _ OXL to become high is after the dead time set by PDARregister after the in _ opxl falling edge. In this case, the TIM4 _ < t > _ OXH output remains low.

When the pulse width of in _ opxl 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 PDBRregister after in _ opxl rising edge. In this case, the TIM4 _ < t > _ OXL output will remain at low level (X = U, V, W, x = u, v, w). As shown in Figure 19-21.

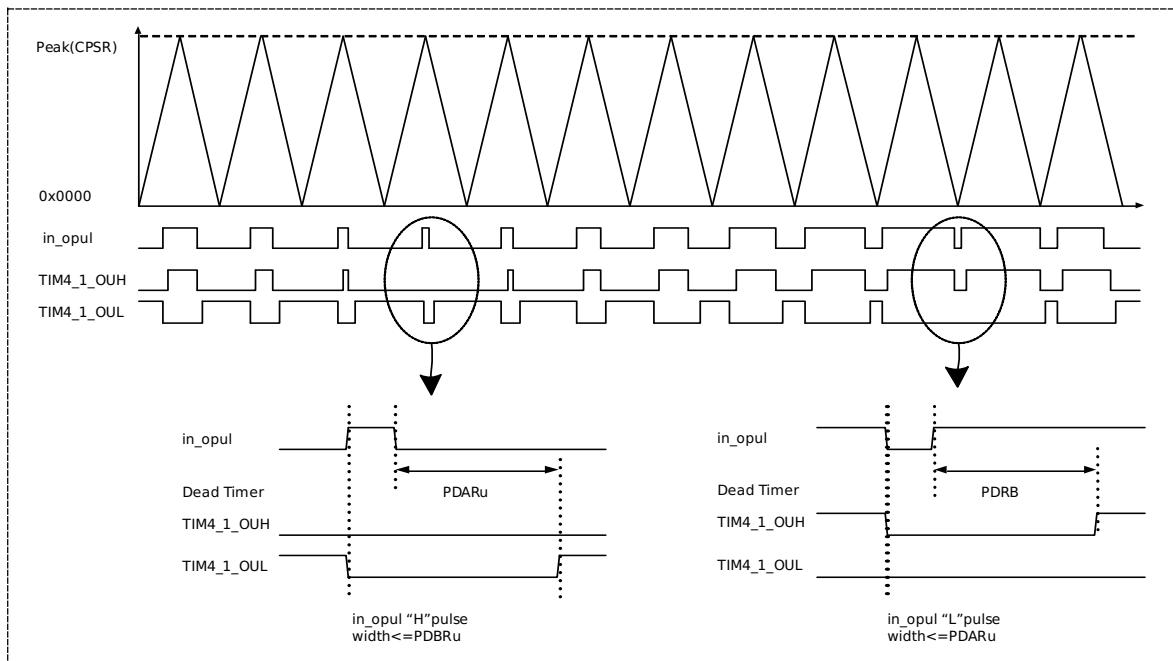


Figure 19-21 Waveform output in dead-band timer mode when the pulse width is abnormal

On the basis of the dead-zone output mode of the above hardware, the pulse width of in _ opxl signal can be monitored to realize the filtering control of in _ opxl signal. This dead-zone output implementation of in _ opxl band pulse width filtering is called dead-zone counter filtering mode (POCR.PWMMMD = 10) (x = u, v, w).

In dead-zone 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 larger than the time set by registerPFSRn, 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) through the dead-time timer mode.

If the rising edge of the signal in _ opxl is detected, the filter counter loads the PFSRRegister value and initiates the high level width of the measure in _ opxl. When the high level pulse width of in _ opxl is greater than the time set by the registerPFSR, after the time set by the PFSR, the TIM4 _ < t > _ OXL output becomes low, the dead zone counter loads the set value of the PDBRRegister and initiates the decrement count, and when the count value becomes 0x0000, the counter stops and causes the TIM4 _ < t > _ OXH output to be high. If the signal in _ opxl falling edge is detected, the filter counter loads the PFSRRegister value and initiates the low-level width of the measure in _ opxl. When the low-level pulse width of in _ opxl is greater than the time set by the registerPFSR, the TIM4 _ < t > _ OXH output becomes low after the time set by the PFSR, the dead zone counter loads the PDARRegister setting value and initiates the decrement count. When the count value becomes 0x0000, the counter stops and causes the TIM4 _ < t > _ OXL output to be high. When the level pulse width of in _ opxl is less than the time set by registerPFSR, the output TIM4 _ < t > _ OXH and TIM4 _ < t > _ OXL will remain unchanged (X = U, V, W, x = u, v, w).

Figure 19-22 Shown is an example of complementary PWM output in dead-band counter filter mode.

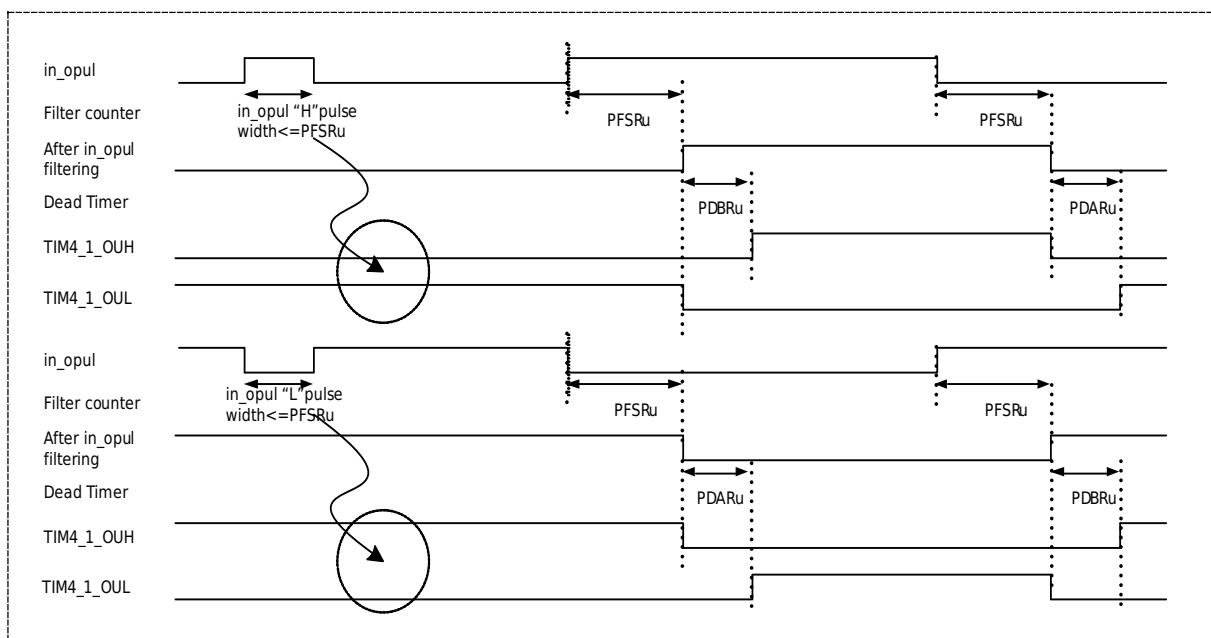


Figure 19-22 Complementary PWM output in dead-band timer Filtering mode

19.3.4 Periodic Interval Response

The underflow interrupt mask counter is used to mask the number of times the underflow flag bit (CCSR.IRQZF) is set. The underflow interrupt mask counter (CVPR.ZIC[3:0]) operates as a down counter and loads the value set by CVPR.ZIM[3:0] at the beginning, when CVPR.ZIC[3:0] = "When 0", the underflow flag (CCSR.IRQZF) is set to "1".

The overflow interrupt mask counter is used to mask the number of times the overflow flag bit (CCSR.IRQPF) is set. The overflow interrupt mask counter (CVPR.PIC[3:0]) operates as a down counter and loads the value set by CVPR.PIM[3:0] at the beginning, when CVPR.PIC[3:0] = "When 0", the overflow flag bit (CCSR.IRQPF) is set to "1".

As shown Figure 19-23 as shown below, it is the timing diagram of setting IRQZF and IRQPF at the time of periodic interval response.

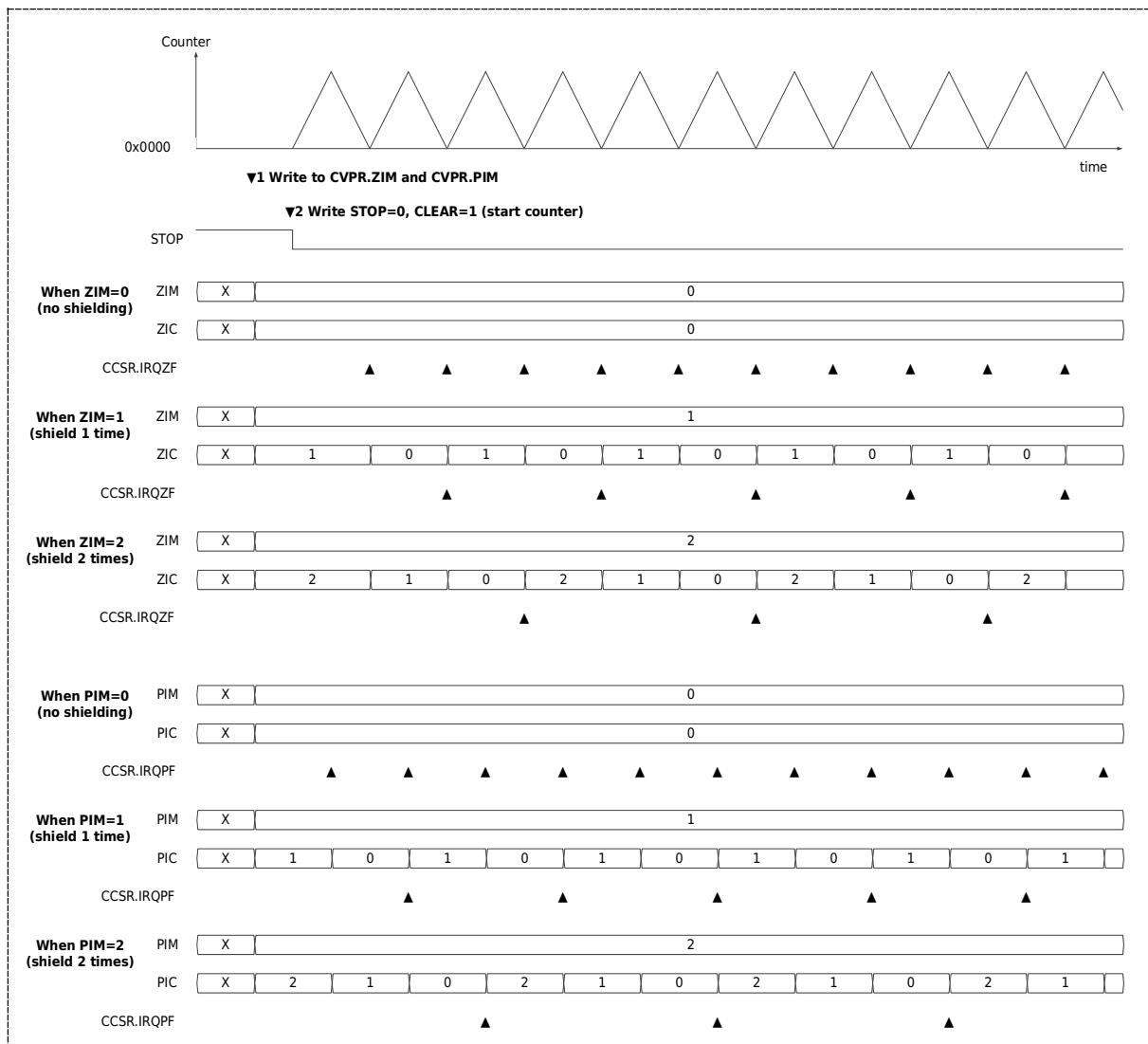


Figure 19-23 Cycle Interval Response Timing Diagram

When counter stops, it writes CVPR.ZIM and CVPR.PIM initialization values, which are immediately reflected internal counter (CVPR.ZIC, CVPR.PIC).

2 Initialize and initiate counter (STOP = 0 and CLEAR = 1), counter starts from zero after bus reset or after software initialize CLEAR = 1, at which point the CCSR.IRQZF flag is not set immediately, and then whenever the count value of interrupt shielded counter is 0x0000 and the count value of counter is 0x0000 and CPSR, the flag "is the time when CCSR.IRQZF or SR.CCIRQPF is set.

Note:

- In the counter run, CVPR.ZIM and CVPR.PIM are written, and this setting does not immediately react to interrupt shielded counter (CVPR.ZIC and CVPR.PIC). If a soft reset is written (CLEAR = 1), the written CVPR.ZIM and CVPR.PIM values are immediately loaded as the initial value of the interrupt mask counter.

A comparison match event (dedicated event output) for a dedicated baseline register (SCCRm) also has a periodic interval response function. When the dedicated control status register SCSR is set to compare mode output (EVTMS=0), the EVT event enable is overflow point (ZEN), up-counting time (UEN), underflow point (PEN) and down-counting time (DEN) . When the cycle interval function is not used, after the enable is enabled, the dedicated event output signal will be set when the corresponding counting match event occurs. Among them, the overflow point matching event and the underflow point matching event do not support the periodic interval function. When using the cycle interval function, in the triangle wave mode, when UEN=1 or DEN=1 and the count value reaches the value set by the SCMRm register, an EVT event will be generated in one effective cycle; UEN=1 and DEN=1 and the count value reaches the value set by the SCMRm register. When the value set by the SCMRm register is set, two EVT events will be generated in one valid period. As shown in Figure 19-24 below is the periodic interval response output graph of the dedicated event output signal.

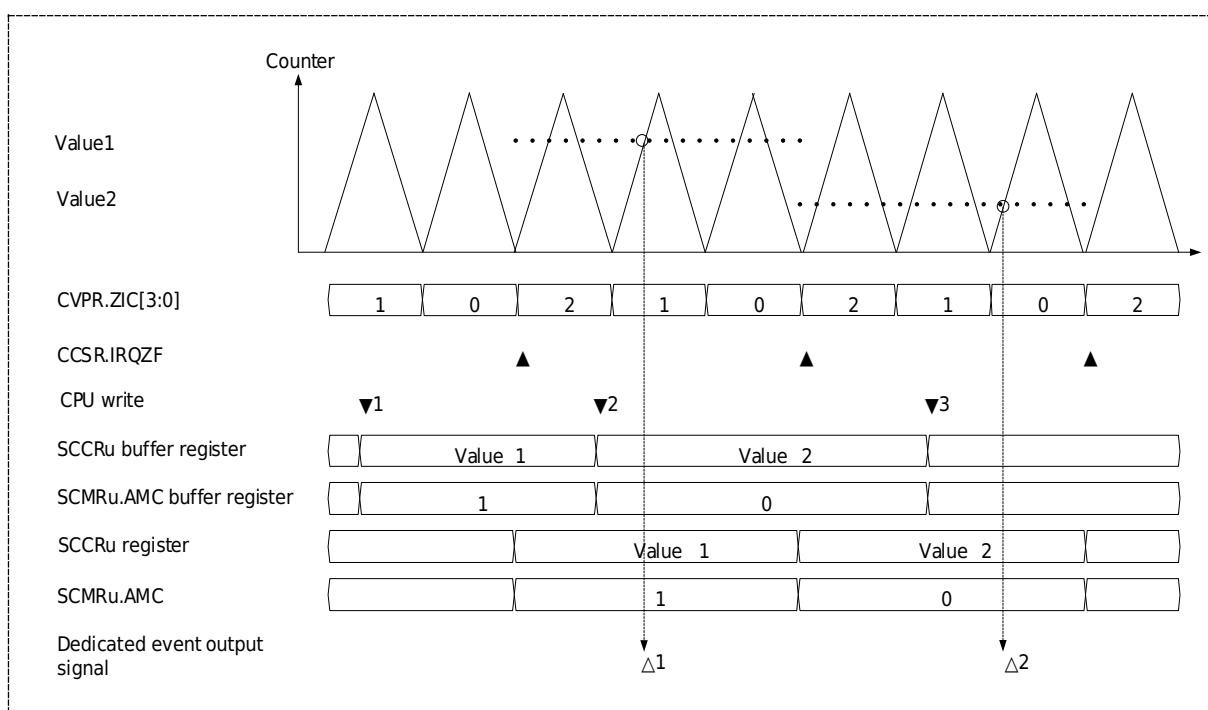


Figure 19-24 Dedicated Event Output Signal Periodic Interval Response Output

Counter is a triangular wave count mode, and the underflow interrupt shielded counter (CVPR.ZIC) is decremented from 2 to 0. Underflow interrupt occurs at the moment.

Write value 1 to SCCR_u buffer register at time 1. Meanwhile, MZCE = 1, MPCE = 0, AMC = 0001 write SCMR_u buffer register. Turn on the count-up EVT enable (UEN=1) in the SCSR register, and the cycle interval response function link enable (LMC=1) and set the underflow buffer (BUFEN=01), the buffer register SCCR_u and SCMR_u transfer operation at the time of ▲ implement. Because when counting up, count value = SCCR_u = Value 1 and MZCE = 1 and AMC = 1, at time △ 1, the dedicated event output signal is set.

At time 2, write value 2 to SCCR_u buffer register. Meanwhile, MZCE = 1, MPCE = 0, AMC = 0000 write SCMR_u buffer register. After that, the buffer register SCCR_u and SCMR_u transfer operations are performed at the moment. Because when counting up, MZCE=1 and AMC=0 and counting value=SCCR_u=Value 2, at time △ 2, the dedicated event output signal is set.

19.3.5 EMB Control

Each Timer 4 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 from the EMB side (refer to EMB chapter).

The output state of the three groups of PWM ports in each unit can be changed to a pre-set state if the abnormal EMB events from EMB are detected during normal output. The preset port state can be output high-impedance state, output low level, output high level, maintain normal output, and maintain the previous state unchanged. (set by ECER.EMBVAL and ECSR.HOLD).

For example, if an EMB event occurs during the normal output of the PWM port of Timer4 when ECER.EMBVAL= 01 is set, the output of the PWM port becomes high resistance.

19.4 Interrupt and Event Description

19.4.1 Count Comparison Match Interrupt

There are 6 common baseline values register (OCCRm), which can be compared with the count values to produce matching valid signals. When the count comparison matches, the OCSRn. OCFH and OCSRn. OCFL bits in the generic control state register (OCSRn) are set to 1. At this time, if OCSRn.OCIEH and OCSRn.OCIEL are set to enable interrupts, the corresponding interrupt requests (TMR4_U<t>_GCMmn, m=U, V, W; n=H, L) will also be triggered.

19.4.2 Count Cycle Match Interrupt

When the sawtooth wave counts up to the overflow point, the triangle wave counts to the valley point, or the triangle wave counts to the peak point, the CCSR.IRQPF or CCSR.IRQZF bit of the control status register (CCSR) will be set to 1. If you set CCSR IRQOPEN or CCSR IRQZEN bit enable interrupt at this time, the count period match interrupt (TMR 4 _ U < t > _ GOVF and TMR 4 _ U < t > _ GUDF) can be triggered at the corresponding time point.

19.4.3 Heavy Load Count Matching Interrupt

When the reload function is valid, TMR4_PFSRn is used as the period count value of the reload timer. When counting is enabled, the register TMR4_PFSRn is reloaded to the initial value of the counter, and the decrement operation is performed. After one cycle is completed, a reload count interrupt request is generated. The RCSR.RTIFU, RCSR.RTIFV, and RCSR.RTIFW bits in the Control Status Register (RCSR) will be set to 1, respectively. If RCSR.RTIDU, RCSR.RTIDV and RCSR.RTIDW interrupt masks are invalid, the corresponding heavy load count match interrupt request (TMR4_U < t >_RLOm, m = U, V, W) will also be triggered.

19.4.4 Dedicated Comparison Matching Event

The six special reference registers (SCCRm) of Timer4 produce six special event output signals, which can be used to select and trigger other modules, such as starting ADC.

In the process of clock counting, if the specific reference value (SCCRm) occurs the matching event of count comparison (TMR4_U < t >_SCMmn, m = U, V, W; N = H, L), which generates a valid request signal that can be configured on 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 compare the start mode or delay start mode. When the start-up mode is compared (SCSR.EVTMS = 0), the specific event output signal is output directly after the SCCR count match event is generated. OCCRxh or OCCRxl (selected by the SCSR.EVTDS bit) are generated at the time of the delayed start-up mode (SCSR.EVTMS = 1); X = u, v, w). After SCCR, the output signal of the special event is output. TheFigure 19-25 following shows an example of the request output of the dedicated event output signal in the delayed start mode.

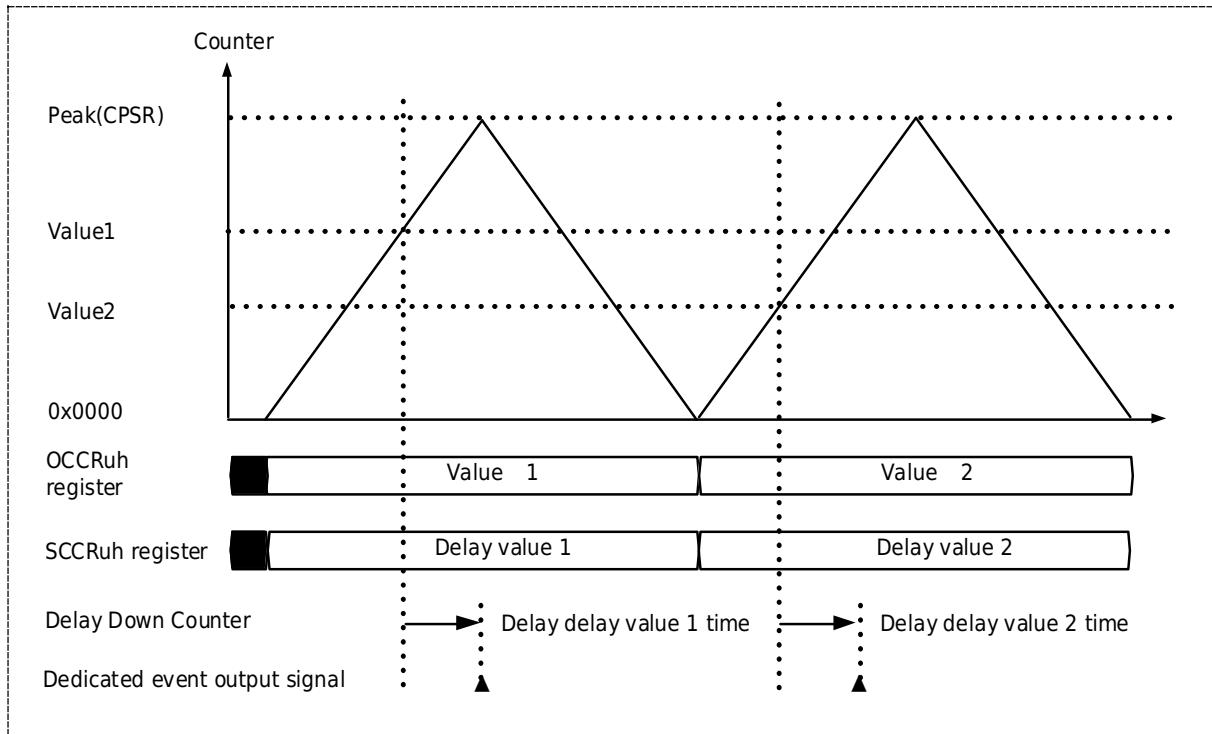


Figure 19-25 Output Timing of Dedicated Event Output Signal in Delayed Start Mode

Note:

- In a delay count run, if an OCCR and counter match event occurs again, the delay counter reloads the count value and decrements the count again. Therefore, if the OCCR match event time interval is less than the set delay time SCCR, the request signal output by the dedicated event may not always be generated.

19.5 Register Description

Table 19-3 shows the register list of the Timer4 module.

BASE ADDR: 0x4001_7000 (U1) , 0x4002_4800 (U2) , 0x4002_4C00 (U3)

Table 19-3 Register List

Register name	Symbol	Offset	Bit width	Reset value
Register count value	TMR4_CNTR	0x0046	16	0x0000
Cycle reference register	TMR4_CPSR	0x0042	16	0xFFFF
Control state	TMR4_CCSR	0x0048	16	0x0040
Valid period register	TMR4_CVPR	0x004A	16	0x0000
Generic baseline registerUH	TMR4_OCCRuh	0x0002	16	0x0000
Generic baseline registerUL	TMR4_OCCRul	0x0006	16	0x0000
Generic baseline registerVH	TMR4_OCCRvh	0x000A	16	0x0000
Generic baseline registerVL	TMR4_OCCRvl	0x000E	16	0x0000
Generic baseline registerWH	TMR4_OCCRwh	0x0012	16	0x0000
Generic baseline registerWL	TMR4_OCCRwl	0x0016	16	0x0000
Generic control state registerU	TMR4_OCSRu	0x0018	16	0xFF00
Generic control state registerV	TMR4_OCSRv	0x001C	16	0xFF00
Generic control state registerW	TMR4_OCSRw	0x0020	16	0xFF00
Generic extension control registerU	TMR4_OCERu	0x001A	16	0x0000
Generic extension control registerV	TMR4_OCERv	0x001E	16	0x0000
Generic Extended Control RegisterW	TMR4_OCERw	0x0022	16	0x0000
Generic mode control registerUH	TMR4_OCMRuh	0x0024	16	0x0000
Generic mode control registerUL	TMR4_OCMRul	0x0028	32	0x0000_0000
Generic mode control registerVH	TMR4_OCMRvh	0x002C	16	0x0000
Generic mode control registerVL	TMR4_OCMRvl	0x0030	32	0x0000_0000
Generic mode control registerWH	TMR4_OCMRwh	0x0034	16	0x0000
Generic mode control registerWL	TMR4_OCMRwl	0x0038	32	0x0000_0000
Dedicated baseline registerUH	TMR4_SCCRuh	0x00B2	16	0x0000
Special baseline registerUL	TMR4_SCCRul	0x00B6	16	0x0000
Dedicated baseline registerVH	TMR4_SCCRvh	0x00BA	16	0x0000
Dedicated baseline registerVL	TMR4_SCCRvl	0x00BE	16	0x0000
Special baseline registerWH	TMR4_SCCRwh	0x00C2	16	0x0000
Dedicated baseline registerWL	TMR4_SCCRwl	0x00C6	16	0x0000
Special control state registerUH	TMR4_SCSRuh	0x00C8	16	0x0000
Special control status registerUL	TMR4_SCSRul	0x00CC	16	0x0000
Special control state registerVH	TMR4_SCSRvh	0x00D0	16	0x0000
Special control state registerVL	TMR4_SCSRvl	0x00D4	16	0x0000
Special control state registerWH	TMR4_SCSRwh	0x00D8	16	0x0000

Special control state registerWL	TMR4_SCSRwl	0x00DC	16	0x0000
Special mode control registerUH	TMR4_SCMRuh	0x00CA	16	0xFF00
Special mode control registerUL	TMR4_SCMRul	0x00CE	16	0xFF00
Special mode control registerVH	TMR4_SCMRvh	0x00D2	16	0xFF00
Special mode control registerVL	TMR4_SCMRvl	0x00D6	16	0xFF00
Special mode control registerWH	TMR4_SCMRwh	0x00DA	16	0xFF00
Special mode control registerWL	TMR4_SCMRwl	0x00DE	16	0xFF00
PWM Basic Control RegisterU	TMR4_POCRu	0x0098	16	0xFF00
PWM Basic Control RegisterV	TMR4_POCRv	0x009C	16	0xFF00
PWM Basic Control RegisterW	TMR4_POCRw	0x00A0	16	0xFF00
PWM filtering control registerU	TMR4_PFSRu	0x0082	16	0x0000
PWM Filtering Control RegisterV	TMR4_PFSRv	0x008A	16	0x0000
PWM Filtering Control RegisterW	TMR4_PFSRw	0x0092	16	0x0000
PWM dead zone control registerAU	TMR4_PDARu	0x0084	16	0x0000
PWM dead zone control registerBU	TMR4_PDBRu	0x0086	16	0x0000
PWM dead zone control registerAV	TMR4_PDARv	0x008C	16	0x0000
PWM dead zone control registerBV	TMR4_PDBRv	0x008E	16	0x0000
PWM dead zone control registerAW	TMR4_PDARw	0x0094	16	0x0000
PWM dead zone control registerBW	TMR4_PDBRw	0x0096	16	0x0000
Overload control status register	TMR4_RCSR	0x00A4	16	0x0000
EMB control status register	TMR4_ECSR	0x00F0	16	0x0000
EMB Extended Control Register	TMR4_ECER	U1: (0x4005_5408) U2: (0x4005_540C) U3: (0x4005_5410)	32	0x0000_0000

Note:

- In the detailed description of the following registers, m=uh, ul, vh, vl, wh, wl, n=u, v, w. The registers pointed to by m respectively correspond to the output control of ports TIM4_<t>_OUH, TIM4_<t>_OUL, TIM4_<t>_OVH, TIM4_<t>_OVL, TIM4_<t>_OWH, TIM4_<t>_OWL, etc. ; The registers indicated by n correspond to the output control of ports TIM4_<t>_OUx, TIM4_<t>_OVx, TIM4_<t>_OWx, etc. when the function is realized, where x=H or L, and the specific control of H or L is in these There is a corresponding symmetry bit in the register.

19.5.1 Count Value Register (TMR4 _ CNTR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNTR[15:0]															
Bit	Marking	Place name		Function											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 the register In a count, the bit indicates the current counter count value Note: You cannot write a value to the register in a count											R/W

19.5.2 Periodic Reference Register (TMR4 _ CPSR)

Reset value: 0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CPSR[15:0]															
Bit	Marking	Place name		Function											Read and write
b15~b0	CPSR[15:0]	General cycle reference		Counter count cycle value Note: When reading data from the local area, it is not the value of the buffer register, but the value of the CPSR register											R/W

19.5.3 Control State Register (TMR4 _ CCSR)

Reset value: 0x0040

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
Bit	Marking		Place name		Function										Read and write			
ECKEN	IRQZF	IRQZEN	-	-	-	IRQPF	IRQPEN	BUFEN	STOP	MODE	CLEAR	CKDIV[3:0]						
b15	ECKEN		Timer selection		0: Internal PCLK1 clock 1: External TIM4_<t>_CLK port input clock Note 1: The bit is set when counter stops Note 2: When using the external TIM4_<t>_CLK port input clock, after writing STOP="1", the first edge of the external input clock is considered invalid, and the counting action starts from the second edge, rising edge and falling edge are valid edges.										R/W			
b14	IRQZF		Underflow state		0: No count underflow 1: Occurrence count underflow Note 1: When the periodic interval response function is used, the setting condition of the bit is set by the periodic interval counter set 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 IRQZF from producing interrupt to CPU 1: Allow IRQZF to generate interrupt to CPU										R/W			
b12~b10	Reserved		-		Read as "0", write as "0"										R/W			
b9	IRQPF		Overflow state		0: No count overflow 1: Occurrence count overflow Note 1: When the cycle interval response function is used, the setting condition of this bit is set by the cycle interval counter set 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			
b8	IRQPEN		Overflow interrupt enable		0: Disable IRQPF from producing interrupt to CPU 1: Allow IRQPF to generate interrupt to CPU										R/W			
b7	BUFEN		Cache enable		0: Disable CPSR caching 1: Enable CPSR caching										R/W			
b6	STOP		Counter enable		0: Counter Startup 1: Counter stopped										R/W			
b5	MODE		Waveform mode		0: Sawtooth mode (counting up only) 1: Triangular wave mode										R/W			
b4	CLEAR		Counter reset		0: No operation 1: Counter zeroing Note: This bit is always read at 0										R/W			
b3~b0	CKDIV		Counting clock division frequency		This bit indicates the counting clock division of the basic counter 0000: The counting clock is PCLK1 0001: Counting clock is PCLK1/2 0010: Counting clock is PCLK1/4 0011: Counting clock is PCLK1/8 0100: Counting clock is PCLK1/16 0101: Counting clock is PCLK1/32 0110: Counting clock is PCLK1/64 0111: Counting clock is PCLK1/128 1000: Counting clock is PCLK1/256 1001: The count clock is PCLK1/512 1010: The counting clock is PCLK1/1024 Please do not set other values Note: The counting clock source is the external TIM4_<t>_CLK port input clock, and the frequency division setting is invalid											R/W		

19.5.4 Valid Period Register (TMR4 _ CVPR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
PIC[3:0]	ZIC[3:0]	PIM[3:0]											ZIM[3:0]			
Bit	Marking	Place name		Function											Read and write	
b15~b12	PIC[3:0]	Overflow interrupt shielding state											The current number of overflow interrupts that need to be masked Note 1: When PIM is rewritten when the counter is stopped, PIC is immediately updated to the new PIM value Note 2: If the PIM is rewritten before the overflow occurs after the counting starts, the PIC will be updated to the new PIM value when it overflows, and the overflow interrupt will still be triggered. PIC is decremented each time thereafter overflow Note 3: If PIM is rewritten in the overflow interrupt, the PIC will be updated to the new PIM value at the next overflow, and the next overflow interrupt will still be triggered. PIC is decremented each time thereafter overflow			R
b11~b8	ZIC[3:0]	Underflow interrupt shielding state											The current number of underflow interrupts that need to be masked Note 1: ZIM is rewritten when the counter is stopped, then ZIC is immediately updated to the new ZIM value Note 2: If the ZIM is rewritten before the overflow occurs after the counting starts, the ZIC will be updated to the new ZIM value when it overflows, and the overflow interrupt will still be triggered. After each overflow ZIC is decremented Note 3: If ZIM is rewritten in the overflow interrupt, the PIC will be updated to the new ZIM value at the next overflow, and the next overflow interrupt will still be triggered. After each overflow ZIC is decremented			R
b7~b4	PIM[3:0]	Overflow interrupt shielding setting											Set the number of overflow interrupts for shielding			R/W
b3~b0	ZIM[3:0]	Underflow interrupt mask setting											Set the number of underflow interrupts of shielded			R/W

19.5.5 Generic Baseline Register (TMR4 _ OCCRm)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OCCR[15:0]															
Bit	Marking	Place name		Function											Read and write
b15~b0	OCCR[15:0]	Common baseline value		Common baseline value Note: When reading data from the local area, it is not the value of the buffer register but the value of the OCCR register											R/W

19.5.6 General Control State Register (TMR4 _ OCSRn)

Reset value: 0xFF00

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
								OCF L	OCF H	OCIE L	OCIE H	OCP L	OCP H	OCE L	OCE H
Bit	Marking	Place name	Function										Read and write		
b15~b8	Reserved	-	Read as "1", write as "1"										R/W		
b7	OCFL	Count Matching L	0: Counter count value is not equal to OCCRxi set value 1: Counter count is equal to OCCRxi set (x=u,v,w) Note: The bit must be valid when OCEL = 1										R/W		
b6	OCFH	Count Matching H	0: Counter count value is not equal to OCCRxh set value 1: Counter count is equal to OCCRxh set (x=u,v,w) Note: The bit must be valid when OCEH = 1										R/W		
b5	OCIEL	Count Matching L interrupt Enable	0: OCFL does not interrupt when set 1: Interrupt occurs when OCFL is set										R/W		
b4	OCIEH	Count Matching H interrupt Enable	0: No interrupt occurs when OCFH is set 1: Interrupt occurs when OCFH is set										R/W		
b3	OCPL	Port status when OCEL=0	0: When OCEL=0, if this bit writes "0", TIM4_<t>_OxL will output a low level 1: When OCEL=0, if this bit writes "1", TIM4_<t>_OxL will output a high level (m=U,V,W) Note: When OCEL=1, the write operation is invalid										R/W		
b2	OCPH	Port status when OCEH=0	0: When OCEH=0, if this bit writes "0", TIM4_<t>_OxH will output a low level 1: When OCEH=0, if this bit writes "1", TIM4_<t>_OxH will output a high level (m=U,V,W) Note: When OCEH=1, the write operation is invalid										R/W		
b1	OCEL	Port output enable L	0: TIM4_<t>_OxL output is invalid, the port status is determined by OCPL 1: TIM4_<t>_OxL output enable, port status is determined by OCMRyL setting and OCFL status (x=U,V,W, y=u,v,w)										R/W		
b0	OCEH	Port output enable H	0: TIM4_<t>_OxH output is invalid, the port status is determined by OCPH 1: TIM4_<t>_OxH output enable, port status is determined by OCMRyH setting and OCFH status (x=U,V,W, y=u,v,w)										R/W		

19.5.7 Generic Extended Control Register (TMR4 _ OCERn)

Reset value: 0x0000

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	MLBUF EN[1:0]	MHBUF EN[1:0]	CLBUF EN[1:0]	CHBUF EN[1:0]				
Bit	Marking	Place name		Function										Read and write	
b15~b14	Reserved	-		Read as "0", write as "0"										R/W	
b13	MCECL	Overflow point match enable L		0: When the count value reaches the overflow point (CNTR=CPSR) and OCCRxl > CPSR, it is forbidden to generate compare match L 1: When the count value reaches the overflow point (CNTR=CPSR) and OCCRxl > CPSR, it is regarded as a compare match L (x=u,v,w)										R/W	
b12	MCECH	Overflow point match enable H		0: When the count value reaches the overflow point (CNTR=CPSR) and OCCRxl > CPSR, it is forbidden to generate compare match H 1: When the count value reaches the overflow point (CNTR=CPSR) and OCCRxl > CPSR, it is regarded as a compare match H (x=u,v,w)										R/W	
b11	LMML	Cycle Interval Response link L		0: Interval response function link is invalid, OCMRxI cache transfer is determined by MLBUFEN setting 1: The periodic interval response function link is valid, and the buffer transmission of OCMRxI must also satisfy CVPR.PIC[3:0]=0000 (when the count overflows) or CVPR.ZIC[3:0]= on the basis of the MLBUFEN setting. 0000 (count underflow) (x=u,v,w)										R/W	
b10	LMMH	Cycle Interval Response link H		0: The periodic interval response function link is invalid, and the buffer transmission of OCMRxH is determined by the setting of MHBUFEN[1:0] 1: The periodic interval response function link is valid, and the buffer transmission of OCMRxH must also satisfy CVPR.PIC[3:0]=0000 (when the count overflows) or CVPR.ZIC[3:0]=0000 (count underflow) (x=u,v,w)										R/W	
b9	LMCL	Cycle Interval Response link L		0: The periodic interval response function link is invalid, and the buffer transmission of OCCRxl is determined by the setting of CLBUFEN[1:0] 1: The periodic interval response function link is valid, and the buffer transmission of OCCRxl must also satisfy CVPR.PIC[3:0]=0000 (when the count overflows) or CVPR.ZIC[3:0]=0000 (count underflow) (x=u,v,w)										R/W	
b8	LMCH	Cycle Interval Response link H		0: The periodic interval response function link is invalid, and the buffer transmission of OCCRxh is determined by the setting of CHBUFEN[1:0] 1: The periodic interval response function link is valid, and the buffer transmission of OCCRxh must also satisfy CVPR.PIC[3:0]=0000 (when the count overflows) or CVPR.ZIC[3:0]=0000 (count underflow) (x=u,v,w)										R/W	
b7~b6	MLBUFEN[1:0]	OCMRxl Cache Transport		00: OCMRxI cache register values are written directly to OCMRxI 01: OCMRxI cache register values are written to OCMRxI at count underflow 10: The value of the OCMRxI cache register is written to OCMRxI when the count overflows 11: The value of the OCMRxI cache register is written to OCMRxI on count underflow or overflow (x=u,v,w)										R/W	
b5~b4	MHBUFEN[1:0]	OCMRxh Cache Transfer		00: OCMRxh Cache register value written directly to OCMRxh 01: OCMRxh Cache register values are written to OCMRxh on count underflow 10: OCMRxh Cache register values are written to OCMRxh on count overflow 11: OCMRxh Cache register values are written to OCMRxh on count underflow or overflow (x=u,v,w)										R/W	

b3~b2	CLBUFEN[1:0]	OCCRxl Cache Transport	00: OCCRxl cache register values are written directly to OCCRxl	R/W
			01: OCCRxl cache register values are written to OCCRxl at count underflow	
			10: The value of the OCCRxl cache register is written to OCCRxl when the count overflows	
			11: The value of the OCCRxl cache register is written to OCCRxl on count underflow or overflow (x=u,v,w)	
b1~b0	CHBUFEN[1:0]	OCCRxh Cache Transport	00: OCCRxh Cache register value written directly to OCCRxh	R/W
			01: OCCRxh Cache register values are written to OCCRxh on count underflow	
			10: OCCRxh Cache register values are written to OCCRxh on count overflow	
			11: OCCRxh Cache register values are written to OCCRxh on count underflow or overflow (x=u,v,w)	

19.5.8 General Mode Control Register (TMR4 _ OCMRm)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OPN ZRH[1:0]	OPN PKH[1:0]	OP ZRH[1:0]	OP UCH[1:0]	OP PKH[1:0]	OP DCH[1:0]	OP ZRH	OCF UCH	OCF PKH	OCF DCH						

Note: This register bit description is used for OCMRuh, OCMRvh, OCMRwh

Bit	Marking	Place name	Function	Read and write
b15~b14	OPNZRH[1:0]	Underflow point port state H	<p>Condition: Count underflow & OCCRxh count mismatch (x = u, v, w) 00: The TIM4 _ < t > _ OyH port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyH port bit output high level when conditions are met 10: TIM4 _ < t > _ OyH port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyH port bit output inversion when conditions are met (y=U,V,W)</p>	R/W
b13~b12	OPNPKH[1:0]	Overflow point port state H	<p>Condition: Count overflow & OCCRxh count mismatch (x = u, v, w) 00: The TIM4 _ < t > _ OyH port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyH port bit output high level when conditions are met 10: TIM4 _ < t > _ OyH port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyH port bit output inversion when conditions are met (y=U,V,W)</p>	R/W
b11~b10	OPZRH[1:0]	Underflow point port state H	<p>Condition: Count underflow & OCCRxh count match (x = u, v, w) 00: The TIM4 _ < t > _ OyH port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyH port bit output high level when conditions are met 10: TIM4 _ < t > _ OyH port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyH port bit output inversion when conditions are met (y=U,V,W)</p>	R/W
b9~b8	OPUCH[1:0]	Up Count Port State H	<p>Condition: Counter Up Count & OCCRxh Count Match (x = u, v, w) 00: The TIM4 _ < t > _ OyH port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyH port bit output high level when conditions are met 10: TIM4 _ < t > _ OyH port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyH port bit output inversion when conditions are met (y=U,V,W)</p>	R/W
b7~b6	OPPKH[1:0]	Overflow point port state H	<p>Condition: Count Overflow & OCCRxh Count Match (x = u, v, w) 00: The TIM4 _ < t > _ OyH port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyH port bit output high level when conditions are met 10: TIM4 _ < t > _ OyH port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyH port bit output inversion when conditions are met (y=U,V,W)</p>	R/W
b5~b4	OPDCH[1:0]	Down count port state H	<p>Condition: Counter Down Count & OCCRxh Count Match (x = u, v, w) 00: The TIM4 _ < t > _ OyH port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyH port bit output high level when conditions are met 10: TIM4 _ < t > _ OyH port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyH port bit output inversion when conditions</p>	R/W

			are met (y=U,V,W)	
b3	OCFZRH	Underflow point OCFH state H	Condition: Count underflow & OCCRxh count match (x = u, v, w) 0: OCSR.OCFH bit remains unchanged when the condition is satisfied 1: OCSR.OCFH position bit when conditions are met	R/W
b2	OCFUCH	Up count OCFH state H	Condition: Counter Up Count & OCCRxh Count Match (x = u, v, w) 0: OCSR.OCFH bit remains unchanged when the condition is satisfied 1: OCSR.OCFH position bit when conditions are met	R/W
b1	OCFPKH	Overflow point OCFH state H	Condition: Count Overflow & OCCRxh Count Match (x = u, v, w) 0: OCSR.OCFH bit remains unchanged when the condition is satisfied 1: OCSR.OCFH position bit when conditions are met	R/W
b0	OCFDCH	Down count OCFH state H	Condition: Counter Down Count & OCCRxh Count Match (x = u, v, w) 0: OCSR.OCFH bit remains unchanged when the condition is satisfied 1: OCSR.OCFH position bit when conditions are met	R/W

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EOPN ZRL[1:0]		EOPN PKL[1:0]		EOP ZRL[1:0]		EOP UCL[1:0]		EOP PKL[1:0]		EOP DCL[1:0]		EOPN UCL[1:0]		EOPN DCL[1:0]	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OPN ZRL[1:0]		OPN PKL[1:0]		OP ZRL[1:0]		OP UCL[1:0]		OP PKL[1:0]		OP DCL[1:0]		OCF ZRL	OCF UCL	OCF PKL	OCF DCL

Note: This Register Bit Description is Used for OCMRul, OCMRvl and OCMRwl

Bit	Marking	Place name	Function	Read and write
b31~b30	EOPNZRL[1:0]	Extended underflow point port state L	Condition: Count underflow & OCCRxl count mismatch & OCCRxh count match (x = u, v, w) 00: TIM4 _ < t > _ OyL port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyL port bit output high level when conditions are met 10: TIM4 _ < t > _ OyL port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b29~b28	EOPNPKL[1:0]	Expanded overflow point port state L	Condition: Count Overflow & OCCRxl Count Mismatch & OCCRxh Count Match (x = u, v, w) 00: TIM4 _ < t > _ OyL port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyL port bit output high level when conditions are met 10: TIM4 _ < t > _ OyL port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b27~b26	EOPZRL[1:0]	Extended underflow point port state L	Condition: Count underflow & OCCRxl count match & OCCRxh count match (x = u, v, w) 00: TIM4 _ < t > _ OyL port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyL port bit output high level when conditions are met 10: TIM4 _ < t > _ OyL port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b25~b24	EOPUCL[1:0]	Expansion up count port status L	Condition: Counter Up Count & OCCRxl Count Match & OCCRxh Count Match (x = u, v, w) 00: TIM4 _ < t > _ OyL port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyL port bit output high level when	R/W

			conditions are met 10: TIM4_<t>_OyL port bit output low level when condition is satisfied 11: TIM4_<t>_OyL port bit output inversion when conditions are met (y=U,V,W)	
b23~b22	EOPPKL[1:0]	Expanded overflow point port state L	Condition: Count Overflow & OCCRxI Count Match & OCCRxH Count Match ($x = u, v, w$) 00: TIM4_<t>_OyL port position remains unchanged when the condition is satisfied 01: TIM4_<t>_OyL port bit output high level when conditions are met 10: TIM4_<t>_OyL port bit output low level when condition is satisfied 11: TIM4_<t>_OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b21~b20	EOPDCL[1:0]	Expansion Down Count Port Status L	Condition: Counter Down Count & OCCRxI Count Match & OCCRxH Count Match ($x = u, v, w$) 00: TIM4_<t>_OyL port position remains unchanged when the condition is satisfied 01: TIM4_<t>_OyL port bit output high level when conditions are met 10: TIM4_<t>_OyL port bit output low level when condition is satisfied 11: TIM4_<t>_OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b19~b18	EOPNUCL[1:0]	Expansion up count port status L	Condition: Counter up count & OCCRxI count mismatch & OCCRxH count match ($x = u, v, w$) 00: TIM4_<t>_OyL port position remains unchanged when the condition is satisfied 01: TIM4_<t>_OyL port bit output high level when conditions are met 10: TIM4_<t>_OyL port bit output low level when condition is satisfied 11: TIM4_<t>_OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b17~b16	EOPNDCL[1:0]	Expansion Down Count Port Status L	Condition: Counter Down Count & OCCRxI Count Mismatch & OCCRxH Count Match ($x = u, v, w$) 00: TIM4_<t>_OyL port position remains unchanged when the condition is satisfied 01: TIM4_<t>_OyL port bit output high level when conditions are met 10: TIM4_<t>_OyL port bit output low level when condition is satisfied 11: TIM4_<t>_OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b15~b14	OPNZRL[1:0]	Underflow point port state L	Condition: Count underflow & OCCRxI count mismatch & OCCRxH Count mismatch ($x = u, v, w$) 00: TIM4_<t>_OyL port position remains unchanged when the condition is satisfied 01: TIM4_<t>_OyL port bit output high level when conditions are met 10: TIM4_<t>_OyL port bit output low level when condition is satisfied 11: TIM4_<t>_OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b13~b12	OPNPKL[1:0]	Overflow point port status L	Condition: Count Overflow & OCCRxI Count Mismatch & OCCRxH Count Mismatch ($x = u, v, w$) 00: TIM4_<t>_OyL port position remains unchanged when the condition is satisfied 01: TIM4_<t>_OyL port bit output high level when conditions are met 10: TIM4_<t>_OyL port bit output low level when condition is satisfied 11: TIM4_<t>_OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b11~b10	OPZRL[1:0]	Underflow point port state L	Condition: Count underflow & OCCRxI count match & OCCRxH count mismatch ($x = u, v, w$) 00: TIM4_<t>_OyL port position remains unchanged when the condition is satisfied 01: TIM4_<t>_OyL port bit output high level when conditions are met 10: TIM4_<t>_OyL port bit output low level when condition	R/W

			is satisfied 11: TIM4 _ < t > _ OyL port bit output inversion when conditions are met (y=U,V,W)	
b9~b8	OPUCL[1:0]	Up count port status L	Condition: Counter up count & OCCRxI count match & OCCRxh count mismatch (x = u, v, w) 00: TIM4 _ < t > _ OyL port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyL port bit output high level when conditions are met 10: TIM4 _ < t > _ OyL port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b7~b6	OPPKL[1:0]	Overflow point port status L	Condition: Count Overflow & OCCRxI Count Match & OCCRxh Count Mismatch (x = u, v, w) 00: TIM4 _ < t > _ OyL port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyL port bit output high level when conditions are met 10: TIM4 _ < t > _ OyL port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b5~b4	OPDCL[1:0]	Down count port status L	Condition: Counter Down Count & OCCRxI Count Match & OCCRxh Count Mismatch (x = u, v, w) 00: TIM4 _ < t > _ OyL port position remains unchanged when the condition is satisfied 01: TIM4 _ < t > _ OyL port bit output high level when conditions are met 10: TIM4 _ < t > _ OyL port bit output low level when condition is satisfied 11: TIM4 _ < t > _ OyL port bit output inversion when conditions are met (y=U,V,W)	R/W
b3	OCFZRL	Underflow point OCFL state L	Condition: Count underflow & OCCRxI count match (x = u, v, w) 0: OCSR.OCFL bit remains unchanged when the condition is satisfied 1: OCSR.OCFL position bits when conditions are met	R/W
b2	OCFUCL	Count up OCFL status L	Condition: Counter Up Count & OCCRxI Count Match (x = u, v, w) 0: OCSR.OCFL bit remains unchanged when the condition is satisfied 1: OCSR.OCFL position bits when conditions are met	R/W
b1	OCFPKL	Overflow point OCFL state L	Condition: Count Overflow & OCCRxI Count Match (x = u, v, w) 0: OCSR.OCFL bit remains unchanged when the condition is satisfied 1: OCSR.OCFL position bits when conditions are met	R/W
b0	OCFDCL	Count down OCFL status L	Condition: Counter Down Count & OCCRxI Count Match (x = u, v, w) 0: OCSR.OCFL bit remains unchanged when the condition is satisfied 1: OCSR.OCFL position bits when conditions are met	R/W

Note:

- When reading data from the local area, it is not the value of the buffer register, but the value of the OCMRRegister.
- TIM4 _ < t > _ OyL can be determined by the count value of OCCRxI and counter (standalone mode of operation), or the count value of OCCRxh and counter, and the count value of OCCRxI and counter (linked mode of operation). Write the same 12-bit value to the bit [31: 20] and bit [15: 4] of register OCMRxl and write OCMRxl [19: 16] to "0000". At this time, the output of TIM4 TIM4_<t>_OyL is not affected by OCCRxh, but only determined by OCCRxI. This mode is called standalone operating mode - channel yH is determined by OCCRxh and channel yL is determined by OCCRxI. If the above conditions are not met in

the stand-alone operation mode, the link operation mode-channel yL output is affected by both OCCRxh and OCCRxI.

(x=u,v,w, y=U,V,W)

19.5.9 Special Reference Register (TMR4 _ SCCRm)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SCCR[15:0]															
Bit	Marking	Place name	Function											Read and write	
b15~b0	SCCR[15:0]	Dedicated baseline value	Dedicated baseline (compare startup mode) or delay baseline (delay startup mode) Note: When reading data from the local area, it is not the value of the buffer register but the value of the SCCRregister											R/W	

19.5.10 Special Control State Register (TMR4 _ SCSRm)

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	ZEN	UEN	PEN	DEN	-	-	EVT DS	EVT MS	-	-	LMC	EVTOS[2:0]	BUFEN[1:0]			
b14	UEN	Underflow 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		
b13	PEN	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		
b12	DEN	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		
b11~b10	Reserved	-		Read as "0", write as "0"										R/W		
b9	EVTDS	EVT Delay Object Selection		0: OCCRxh matches as a delay comparison object in delay start mode 1: OCCRxl matches as a delay comparison object in delay start mode (x=u,v,w) Note: The bit is invalid when EVTMS = 0										R/W		
b8	EVTMS	EVT mode selection		0: Compare startup mode (CNTR and SCCR comparison triggers) 1: Delay start mode (compare matching events triggered after SCCR delay)										R/W		
b7~b6	Reserved	-		Read as "0", write as "0"										R/W		
b5	LMC	Periodic interval response link		0: Interval response function link invalid, SCCR cache transfer is determined by BUFEN setting 1: The periodic interval response function link is valid, and the cache transmission of SCCR is based on the setting of BUFEN, and must also satisfy CVPR.PIC[3:0]=0000 (when the count overflows) or CVPR.ZIC[3:0]=0000 (count underflow)										R/W		
b4~b2	EVTOS[2:0]	EVT Output Selection		000: EVT Output Valid for Special Event 0 001: EVT Output of Special Event 1 is Valid 010: EVT Output of Special Event 2 is Valid 011: EVT Output of Special Event 3 is Valid 100: EVT Output Valid for Special Event 4 101: EVT Output of Special Event 5 is Valid Please do not set other values										R/W		
b1~b0	BUFEN[1:0]	SCCR & SCMR Cache Transport		00: SCCR, SCMR Cache register values are written directly to SCCR, SCMR 01: SCCR, SCMR Cache register values are written to SCCR, SCMR when counting underflow 10: SCCR, SCMR Cache register values are written to SCCR, SCMR when count overflows 11: SCCR, SCMR Cache register values are written to SCCR, SCMR when counting underflow or overflow										R/W		

19.5.11 Special Mode Control Register (TMR4_SCMRm)

Reset value: 0xFF00

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								MPCE	MZCE	-	-	AMC[3:0]			
Bit	Marking		Place name		Function								Read and write		
b15~b8	Reserved		-		Read as "1", write as "1"								R/W		
b7	MPCE		Periodic interval response enable		0: No comparison between AMC and CVPR.PIC 1: Enable comparison between AMC and CVPR.PIC								R/W		
b6	MZCE		Periodic interval response enable		0: No comparison between AMC and CVPR.ZIC 1: Enable comparison between AMC and CVPR.ZIC								R/W		
b5~b4	Reserved		-		Read as "0", write as "0"								R/W		
b3~b0	AMC[3:0]		Special event output interval value		This bit sets the 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 area, it is not the value of the buffer register, but the value of the SCMRregister.

19.5.12 PWM Basic Control Register (TMR4 _ POCRn)

Reset value: 0xFF00

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0						
Reserved								LVLS[1:0]		PWMMMD[1:0]		-		DIVCK[2:0]							
Bit	Marking		Place name		Function								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 TIM4_<t>_OxH nor TIM4_<t>_OxL are inverted 01: Output of TIM4_<t>_OxH and TIM4_<t>_OxL are inverted 10: The output of TIM4_<t>_OxH is reversed, and the output of TIM4_<t>_OxL is not reversed. 11: The output of TIM4_<t>_OxH is not reversed, the output of TIM4_<t>_OxL is reversed.								R/W								
b5~b4	PWMMMD[1:0]		PWM output mode		00: Pass-through mode 01: Dead-zone timer mode 10: Dead-zone timer filtering mode 11: Prohibitions								R/W								
b3	Reserved		-		Read as "0", write as "0"								R/W								
b2~b0	DIVCK[2:0]		Counting clock division frequency		This bit indicates the frequency division of the counting clock between the filter counter and the dead-zone counter. 000: The counting clock is PCLK1 001: Counting clock is PCLK1/2 010: Counting clock is PCLK1/4 011: Counting clock is PCLK1/8 100: Counting clock is PCLK1/16 101: Counting clock is PCLK1/32 110: Counting clock is PCLK1/64 111: Counting clock is PCLK1/128								R/W								

19.5.13 PWM Filtering Control Register (TMR4 _ PFSRn)

Reset Value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PFSR[15:0]															
Bit	Marking	Place name	Function										Read and write		
b15~b0	PFSR[15:0]	Filtering initial value	Filter count initial value Note: When the dead-zone timer filter mode is not selected for the PWM waveform output mode, the 16-bit filter counter is used as the 16-bit heavy-duty counter. At this time, the 16-bit filter counter can periodically generate interrupt output, which is independent of the PWM waveform generator function.										R/W		

19.5.14 PWM Dead Zone Control Register (TMR4 _ PDA < B > Rn)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PDA/BR[15:0]															
Bit	Marking	Place name	Function										Read and write		
b15~b0	PDA/BR[15:0]	Dead zone initial value	Dead zone count initial value										R/W		

19.5.15 Overload Control Status Register (TMR4 _ RCSR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RTS W	RTE W	RTIC W	RTIF W	RTS V	RTE V	RTIC V	RTIF V	RTS U	RTE U	RTIC U	RTIF U	-	RTID W	RTID V	RTID U
Bit	Marking	Place name	Function										Read and write		
b15	RTSW	Heavy-duty counter stops	0: No operation 1: Stop heavy-duty counterW and clear RTIFW Note: The bit is always 0 when read										R/W		
b14	RTEW	Heavy-duty counter starting W	0: Heavy-duty counterW stopped 1: Heavy-duty counterW startup										R/W		
b13	RTICW	Zero count matching state W	0: No operation 1: Clear RTIFW flag bit Note: The bit is always 0 when read										R/W		
b12	RTIFW	Count Matching Status W	0: Overloaded counter count does not match PFSRw 1: Overloaded counter counts match PFSRw										R		
b11	RTSV	Heavy-duty counter stop V	0: No operation 1: Stop heavy-duty counterV and clear RTIFV Note: The bit is always 0 when read										R/W		
b10	RTEV	Heavy-duty counter starting V	0: Heavy-duty counterV stop 1: Heavy-duty counterV startup										R/W		
b9	RTICV	Zero count matching state V	0: No operation 1: Clear RTIFV flag bit Note: The bit is always 0 when read										R/W		
b8	RTIFV	Count matching state V	0: Overloaded counter count does not match PFSRv 1: Overloaded counter counts match PFSRv										R		
b7	RTSU	Heavy-duty counter stop U	0: No operation 1: Stop heavy-duty counterU and clear RTIFU Note: The bit is always 0 when read										R/W		
b6	RTEU	Heavy-duty counter startup U	0: Heavy-duty counterU stopped 1: Heavy-duty counterU startup										R/W		
b5	RTICU	Zero count matching state U	0: No operation 1: Clear RTIFU flag bit Note: The bit is always 0 when read										R/W		
b4	RTIFU	Count matching state U	0: Overloaded counter count does not match PFSRu 1: Overloaded counter counts match PFSRu										R		
b3	Reserved	-	Read as "0", write as "0"										R/W		
b2	RTIDW	Heavy-duty interrupt shielding W	0: When the overload function is valid, the overload interruptW output is valid. 1: When the overload function is active, the overload interruptW output is invalid										R/W		
b1	RTIDV	Heavy-duty interrupt shielding V	0: When the overload function is valid, the overload terminal V output is valid. 1: When the overload function is active, the overload interruptV output is invalid										R/W		
b0	RTIDU	Heavy-duty interrupt shielding U	0: When the overload function is active, the overload interruptU output is active 1: When the overload function is active, the overload interruptU output is invalid										R/W		

19.5.16 EMB Control State Register (TMR4_ECSR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										HOLD	Reserved				
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b15~b8	Reserved	-	Read as "0", write as "0"	R/W											
b7	HOLD	PWM hold	0: When an EMB input event is detected, the PWM outputs normally 1: When an EMB input event is detected, the current PWM output state will be maintained and will not change	R/W											
b6~b0	Reserved	-	Read as "0", write as "0"	R/W											

19.5.17 EMB Extended Control Register (TMR4_ECER)

Reset value: 0x0000

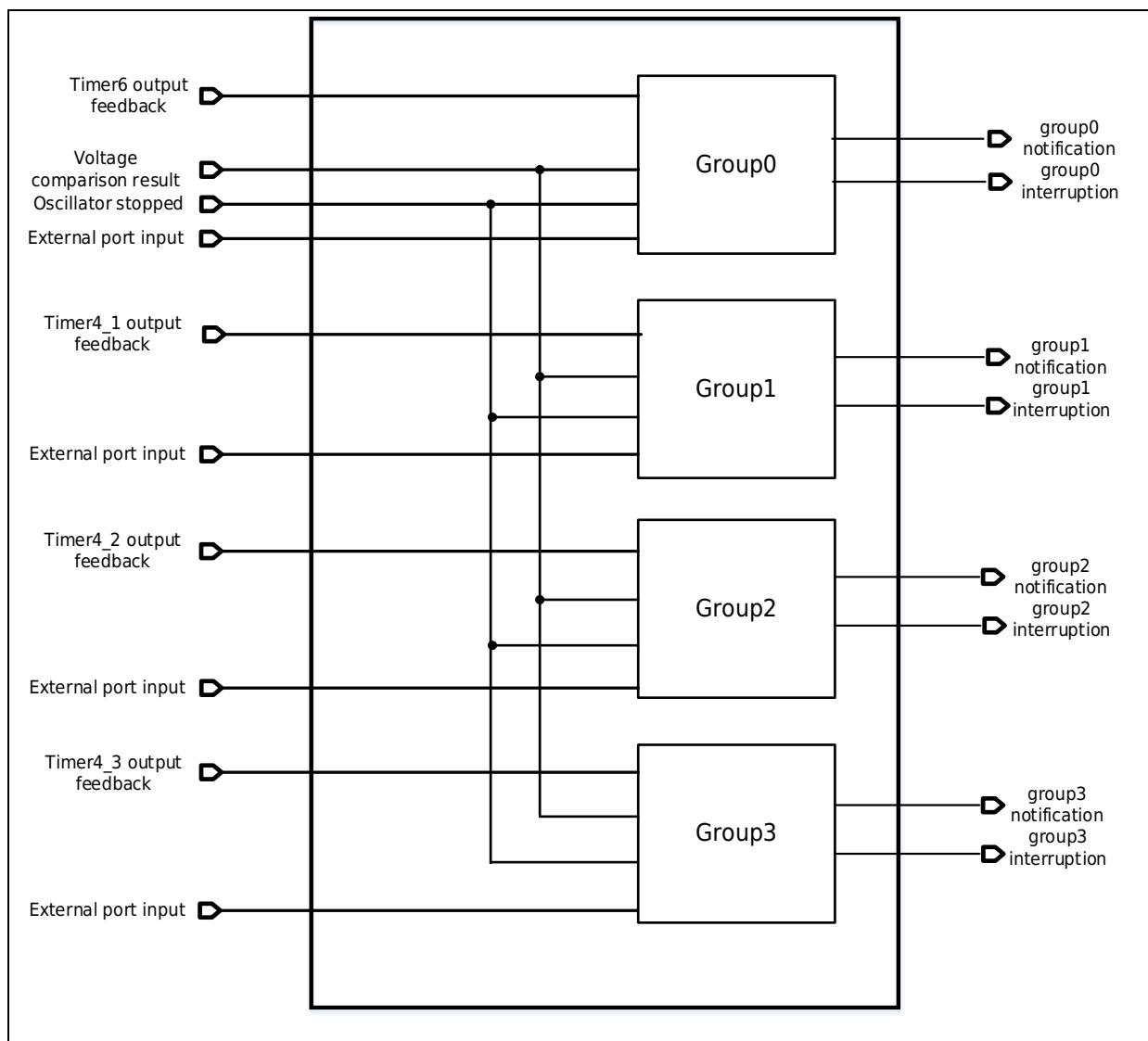
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved														EMBVAL[1:0]	
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b15~b2	Reserved	-	Read as "0", write as "0"	R/W											
b1~b0	EMBVAL[1:0]	EMB State Control	00: When an EMB event occurs, the state of the PWM port is determined by the setting of the ECSR.HOLD bit 01: When an EMB event occurs, the output of the PWM port becomes Hiz 10: When an EMB event occurs, the PWM port output is fixed at low level 11: When an EMB event occurs, the PWM port output is fixed at high level	R/W											

20 Emergency Brake Module (EMB)

20.1 Introduction

The Emergency Brake Module is a functional module that notifies the timer to stop the output of PWM signals to the external motor when certain conditions are met. The following events are used to generate notifications:

- 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



20.2 Functional Description

20.2.1 Overview

The EMB is used to output a notification signal to the timer module (Timer4, Timer6) with PWM function when a certain condition is met, and notify the timer module to close the current PWM output. The EMB module has 4 groups (group), among which group0 corresponds to Timer6, which is selected and used by the Timer6 register setting, and group1~3 corresponds to the 3 units of Timer4.

20.2.2 Stop PWM Signal Output When Input Level Of External Port Changes

Each group of EMB allocates a port to stop PWM signal output when the input level of the external port changes. The port assignments are shown in the table below.

Functional name	Function	Corresponding timer module
EMB_IN1	Group 0 port input control signal	Timer6
EMB_IN2	Group 1 port input control signal	Timer4_1
EMB_IN3	Group 2 port input control signal	Timer4_2
EMB_IN4	Group 3 port input control signal	Timer4_3

According to the setting of the control register EMB_CTLx (x=0~3), EMB can send a notification signal when the port level is high (INVSEL=0) or when the port level is low (INVSEL=1). At the same time, the port is equipped with digital filtering function, which can be set as required. When a qualified level change occurs on the port, EMB will generate a notification signal to notify Timer6 and Timer4. After receiving the notification, Timer6 and Timer4 can set the output port to high level, low level or high impedance according to the register setting. state. EMB can also generate interrupt requests. When the port level becomes invalid, the user can clear the notification signal by writing the EMB status reset register (EMB_STATCLR x) (x=0~3), so that Timer6 and Timer4 can resume outputting PWM waves.

20.2.3 Stop PWM Signal Output when the Level of PWM Output Port is in the Same Phase (Same High or Same Low)

EMB can monitor the complementary PWM output signals of Timer6 and Timer4. When the output signals are both high or low, EMB will generate a notification signal to Timer6 and Timer4. 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 notification. EMB can also generate interrupt requests. Among them, group0 can be used to monitor the complementary PWM output signal of Timer6, and group1-3 can be used to monitor the complementary PWM output signal of Timer4_1/Timer4_2/Timer4_3. When the same-high or same-low condition of the port is released, that is, after the PWMSF bit of the EMB status register (EMB_STATx) ($x=0\sim 3$) is reset, the user can write the EMB status reset register (EMB_STATCLR x) ($x=0\sim 3$) to Clear the notification signal so that Timer6 and Timer4 resume outputting PWM waves.

Port name	Function	Corresponding group	EMB_CTLx control bits	EMB_PWM LVx control bit
TIM6Am($m=1\sim 3$)	Complementary PWM Output Signal of Timer6	group0	PWMSEN[2:0]	PWMLV[2:0]
TIM6Bm($m=1\sim 3$)				
TIM4OUH_1	Complementary PWM Output Signal of Timer4_1	group1	PWMSEN[2]	PWMLV[2]
TIM4OUL_1			PWMSEN[1]	PWMLV[1]
TIM4OVH_1			PWMSEN[0]	PWMLV[0]
TIM4OVL_1			PWMSEN[2]	PWMLV[2]
TIM4OWH_1			PWMSEN[1]	PWMLV[1]
TIM4OWL_1			PWMSEN[0]	PWMLV[0]
TIM4OUH_2	Complementary PWM Output Signal of Timer4_2	group2	PWMSEN[2]	PWMLV[2]
TIM4OUL_2			PWMSEN[1]	PWMLV[1]
TIM4OVH_2			PWMSEN[0]	PWMLV[0]
TIM4OVL_2			PWMSEN[2]	PWMLV[2]
TIM4OWH_2			PWMSEN[1]	PWMLV[1]
TIM4OWL_2			PWMSEN[0]	PWMLV[0]
TIM4OUH_3	Complementary PWM Output Signal of Timer4_3	group3	PWMSEN[2]	PWMLV[2]
TIM4OUL_3			PWMSEN[1]	PWMLV[1]
TIM4OVH_3			PWMSEN[0]	PWMLV[0]
TIM4OVL_3			PWMSEN[2]	PWMLV[2]
TIM4OWH_3			PWMSEN[1]	PWMLV[1]
TIM4OWL_3			PWMSEN[0]	PWMLV[0]

20.2.4 Stop PWM Signal Output According To Voltage Comparator Comparison Result

Each group of the EMB can send a notification signal to Timer6 and Timer4 according to the comparison result of the voltage comparator. For the setting of the voltage comparator output result, refer to the voltage comparator section. When the voltage comparator comparison result flag is set, EMB will generate a notification signal to Timer6 and Timer4. 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 notification. EMB can also generate interrupt requests. After the voltage comparator flag is reset, the user can clear the notification signal by writing the EMB status reset register (EMB_STATCLR x) ($x=0\sim3$), so that Timer6 and Timer4 can resume outputting PWM waves.

20.2.5 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. For the setting of external oscillator stopping oscillating, please refer to the voltage comparator section. When the external oscillator stop oscillating flag is activated, EMB will generate a notification signal to Timer6 and Timer4. 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 notification. EMB can also generate interrupt requests. When the external oscillator stop oscillating flag is reset, the user can clear the notification signal by writing the EMB status reset register (EMB_STATCLR x) ($x=0\sim3$), so that Timer6 and Timer4 resume outputting PWM waves.

20.2.6 Write Register Software Control PWM Signal Output

The EMB software output enable control register (EMB_SOEx) ($x=0\sim3$) allows the user to send a notification signal to Timer6 and Timer4 by directly setting and resetting the software, and no interrupt request will be generated when the software controls the PWM output.

20.3 Register Description

Register Overview

Name	Abbreviation	Note	Offset address
EMB output control register	EMB_CTL	Conditions for controlling the enable of each output	0x0
EMB Feedback Level Selection Register	EMB_PWMLV	Effective Level of Selective PWM Feedback Signal	0x4
EMB software output enable control register	EMB_SOE	Software Output Forbidding Event	0x8
EMB status register	EMB_STAT	Indicates the output enabled state	0xC
EMB state reset register	EMB_STATCLR	Clear output enable state	0x10
EMB interrupt enable register	EMB_INTEN	Control interrupt enable	0x14

20.3.1 EMB Control Register 0 (EMB_CTL0)

The register is a single write register that can only be written once after a reset.

Address: 0x4001_7C00

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16								
INVSEL	NFEN	NFSEL[1:0]		Reserved																			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0								
Reserved															PWMSEN2	PWMSEN1	PWMSEN0	OSCS_TPEN	Res.	CMPE_N3	CMPE_N2	CMPE_N1	PORTINEN

Bit	Marking	Place name	Function	Read and write
b31	INVSEL	Port input active level selection	0: High level valid 1: Low level valid	R/W
b30	NFEN	Port input digital filter enable	0: Invalid filter 1: Filter is valid	R/W
b29~b28	NFSEL[1:0]	Filtering Clock Selection for Digital Filters	00: Using bus clock filtering 01: 8 Frequency Division Filter Using Bus Clock 10: 32 Frequency Division Filter with Bus Clock 11: 128 Frequency Division Filter with Bus Clock	R/W
b27~b9	Reserved		Read 0 when reading, write 0 when writing	R
b8	PWMSEN2	TIM6A/B_3 short circuit output control enable	0: Invalid output control when short circuit 1: Output control effective when short circuit	R/W
b7	PWMSEN1	TIM6A/B_2 short circuit output control enable	0: Invalid output control when short circuit 1: Output control effective when short circuit	R/W
b6	PWMSEN0	TIM6A/B_1 short circuit output control enable	0: Invalid output control when short circuit 1: Output control effective when short circuit	R/W
b5	OSCSTPEN	Oscillator Stop Output Control Enable	0: Output control invalid when oscillator stops oscillating 1: The output control is valid when oscillator stops oscillating	R/W
b4	Reserved		Read 0 when reading, write 0 when writing	R
b3	CMPEN3	CMP3 Voltage comparator comparison result control enable	0: Invalid voltage comparator output control 1: Voltage comparator output control is effective	R/W
b2	CMPEN2	CMP2 Voltage comparator comparison result control enable	0: Invalid voltage comparator output control 1: Voltage comparator output control is effective	R/W
b1	CMPEN1	CMP1 Voltage comparator comparison result control enable	0: Invalid voltage comparator output control 1: Voltage comparator output control is effective	R/W
b0	PORTINEN	Port input control enable	0: Invalid port input control 1: Port input control is valid	R/W

20.3.2 EMB Control Register 1~3 (EMB_CTL1~3)

The register is a single write register that can only be written once after a reset.

Address: 0x4001_7C20, 0x4001_7C40, 0x4001_7C60

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
INVSEL	NFEN	NFSEL[1:0] Reserved													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															PORTEIN
Bit	Marking	Place name		Function											Read and write
b31	INVSEL	Port input active level selection		0: High level valid 1: Low level valid											R/W
b30	NFEN	Port input digital filter enable		0: Invalid filter 1: Filter is valid											R/W
b29~b28	NFSEL[1:0]	Filtering Clock Selection for Digital Filters		00: Using bus clock filtering 01: 8 Frequency Division Filter Using Bus Clock 10: 32 Frequency Division Filter with Bus Clock 11: 128 Frequency Division Filter with Bus Clock											R/W
b27~b9	Reserved	Read 0 when reading, write 0 when writing													R
b8	PWMSEN2	U phase short circuit output control enable of TIM4 x(x=1~3)		0: Invalid output control when short circuit 1: Output control effective when short circuit											R/W
b7	PWMSEN1	V phase short circuit output control enable of TIM4 x(x=1~3)		0: Invalid output control when short circuit 1: Output control effective when short circuit											R/W
b6	PWMSENO	W phase short circuit output control enable of TIM4 x(x=1~3)		0: Invalid output control when short circuit 1: Output control effective when short circuit											R/W
b5	OSCSTPEN	Oscillator Stop Output Control Enable		0: Output control invalid when oscillator stops oscillating 1: The output control is valid when oscillator stops oscillating											R/W
b4	Reserved	Read 0 when reading, write 0 when writing													R
b3	CMPEN3	CMP3 Voltage comparator comparison result control enable		0: Invalid voltage comparator output control 1: Voltage comparator output control is effective											R/W
b2	CMPEN2	CMP2 Voltage comparator comparison result control enable		0: Invalid voltage comparator output control 1: Voltage comparator output control is effective											R/W
b1	CMPEN1	CMP1 Voltage comparator comparison result control enable		0: Invalid voltage comparator output control 1: Voltage comparator output control is effective											R/W
b0	PORTINEN	Port input control enable		0: Invalid port input control 1: Port input control is valid											R/W

20.3.3 EMB Feedback Level Selection Register 0 (EMB_PWMVL0)

The register is a single write register that can only be written once after a reset

Address: 0x4001_7C04

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Bit	Marking	Place name			Function								Read and write		
b31~3	Reserved				Read 0 when reading, write 0 when writing								R		
b2	PWMLV2	TIM6A/B_3 output active level selection			0: Low to active 1: High to active								R/W		
b1	PWMLV1	TIM6A/B_2 output active level selection			0: Low to active 1: High to active								R/W		
b0	PWMLV0	TIM6A/B_1 output active level selection			0: Low to active 1: High to active								R/W		

20.3.4 EMB Feedback Level Selection Register 1~3 (EMB_PWMLV1~3)

The register is a single write register that can only be written once after a reset

Address: 0x4001_7C24, 0x4001_7C44, 0x4001_7C64

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Bit	Marking	Place name	Function												Read and write
b31~3	Reserved		Read 0 when reading, write 0 when writing												R
b2	PWMLV2	U phase output active level selection of TIM4_x(x=1~3)	0: Low to active 1: High to active												R/W
b1	PWMLV1	V phase output active level selection of TIM4_x(x=1~3)	0: Low to active 1: High to active												R/W
b0	PWMLV0	W phase output active level selection of TIM4_x(x=1~3)	0: Low to active 1: High to active												R/W

20.3.5 EMB Software Output Enable Control Register (EMB_SOEx) (x=0~3)

Address: 0x4001_7C08, 0x4001_7C28, 0x4001_7C48, 0x4001_7C68

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Bit	Marking	Place name	Function												Read and write
b31~1	Reserved		Read 0 when reading, write 0 when writing												R
b0	SOE	Software control output	0: PWM normal output 1: PWM stop output												R/W

20.3.6 EMB Status Register (EMB_STATx) (x=0~3)

Address: 0x4001_7C0C, 0x4001_7C2C, 0x4001_7C4C, 0x4001_7C6C

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Bit	Marking	Place name	Function	Read and write											
b31~6	Reserved		Read 0 when reading, write 0 when writing	R											
b5	PWMST	PWM output state	0: No PWM output in phase 1: PWM output in phase	R											
b4	PORTINST	Port input control status	0: Port input control is invalid 1: Port input control is active	R											
b3	OSF	Oscillator Stop Shock Stop PWM Output State	0: No stop of PWM output due to oscillator stop 1: Current stop PWM output due to oscillator stop concussion	R											
b2	CMPF	The state of stopping PWM output by a voltage comparator.	0: No PWM output is stopped due to voltage comparator comparison 1: Current stop of PWM output due to comparison result of voltage comparator	R											
b1	PWMSF	The state of the PWM output stopped in the same phase	0: Current PWM output is not stopped due to the same phase of PWM output feedback 1: The current PWM output is stopped due to the same phase of PWM output feedback	R											
b0	PORTINF	Port Input Control Stopped PWM Output State	0: No PWM output stopped due to port input control 1: Current stop of PWM output due to port input control	R											

20.3.7 EMB Status Reset Register (EMB_STATCLR x) ($x=0\sim3$)

Address: 0x4001_7C10, 0x4001_7C30, 0x4001_7C50, 0x4001_7C70

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Bit	Marking	Place name	Function	Read and write											
b31~4	Reserved		Read 0 when reading, write 0 when writing	R											
b3	OSFCLR	Reduction EMB_STAT.OSF	0: Nothing 1: When CMU.MOSCSTP=0, clear the EMB_STAT.OSF bit and restore the PWM output stopped due to the oscillator stop oscillating	W											
b2	CMPFCLR	Reduction EMB_STAT.CMPF	0: Nothing 1: When CMPMONn.OMON=0, clear the EMB_STAT.CMPF bit and restore the PWM output stopped due to the comparison result of the voltage comparator	W											
b1	PWMSFCLR	Reduction EMB_STAT.PWMSF	0: Nothing 1: When EMB_STAT.PWMST = 0, clear EMB_STAT.PWMSF bits and resumes PWM output stopped by the same phase of PWM output feedback	W											
b0	PORTINFCLR	Reduction EMB_STAT.PORTINF	0: Nothing 1: When EMB_STAT.PORTINST = 0, clear EMB_STAT.PORTINF bits and resumes PWM output stopped due to port input control	W											

20.3.8 EMB Interrupt Permission Register (EMB_INENx) (x=0~3)

Address: 0x4001_7C14, 0x4001_7C34, 0x4001_7C54, 0x4001_7C74

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~4	Reserved		Read 0 when reading, write 0 when writing	R
b3	OSINTEN	Oscillator Stop Shock Stop PWM Interrupt Enable	0: Oscillator stops concussion and stops PWM without terminating 1: Oscillator interrupts when the shock stops the PWM	R/W
b2	CMPINTEN	Voltage comparator stops terminating PWM	0: Voltage comparator comparison results stop PWM without terminating 1: Interrupt when voltage comparator results stop PWM	R/W
b1	PWMINTEN	Interrupt enable of PWM output in same phase stop PWM	0: No interrupt occurs when the PWM output stops in the same phase 1: Interrupt when PWM output stops in the same phase	R/W
b0	PORTINTEN	Port input control stop PWM interrupt enable	0: No interrupt occurs when port input control stops PWM 1: Interrupt when port input control stops PWM	R/W

21 General Timer (TimerA)

21.1 Introduction

General Timer A (Timer A) is a timer with 16-bit count width and 8 PWM outputs. The timer supports two waveform modes of triangle wave and sawtooth wave, and can generate various PWM waveforms; supports counter synchronous start; comparison reference value register supports buffer function; supports inter-unit cascading to realize 32-bit counting; supports 2-phase quadrature encoding counting and 3-phase quadrature encoding counts. This series of products is equipped with 6 units TimerA, which can realize a maximum of 48 PWM outputs.

21.2 Basic Block Diagram

The basic functions and features of TimerA are as shown Table 21-1.

Table 21-1 Basic Functions and Features of TimerA

Waveform mode	Sawtooth wave, triangular wave
Basic functions	• Direction of increments and decrements
	• Synchronous start counter
	• Benchmark caching function
	• 32-bit cascade count
	• Orthogonal coding count
	• 8-Channel PWM Output
	• Compare Matched Event Output
Interrupt type	• Compare match interrupt
	• Periodic matching interrupt

The basic block diagram of TimerA is as Figure 21-1 shown. In the figure, "< t >" is the unit number, that is, "< t >" is 1 to 6. In this section, when "< t >" is mentioned later, it refers to the unit number and is not repeated

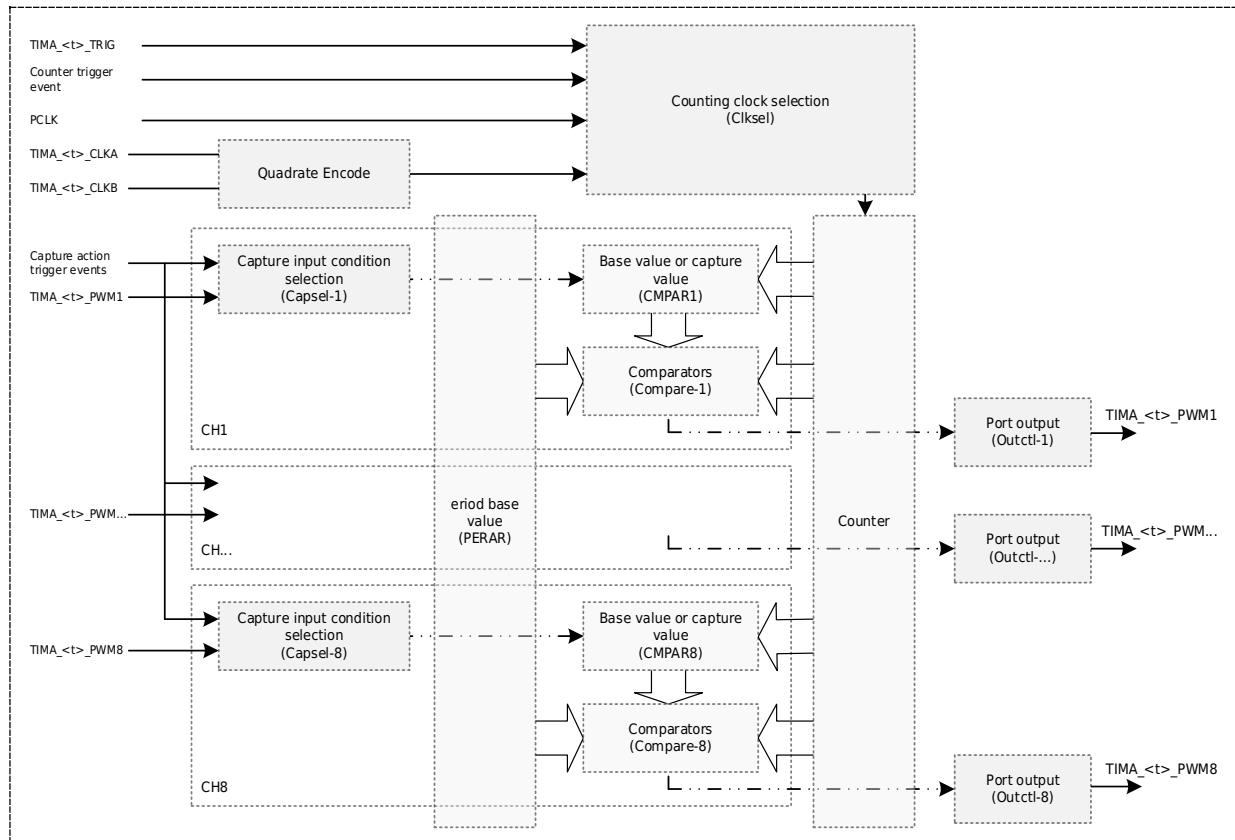


Figure 21-1 TimerA Basic Block Diagram

Table 21-2 Shown is the list of input and output ports of TimerA.

Table 21-2 TimerA port list

Port name	Direction	Function
TIMA_<t>_PWMr	in or out	Capture input event port or PWM output port ($m = 1 \sim 8$)
TIMA_<t>_CLKA	in	Orthogonal Coded Count Event Input Port
TIMA_<t>_CLKB		
TIMA_<t>_TRIG	in	Hardware Trigger Startup, Stop, Zero Event Input Port

21.3 Functional Description

21.3.1 Basic Action

21.3.1.1 Waveform Mode

TimerA has two basic counting wave modes: Serrated wave mode and triangular wave mode. The basic waveforms of the two waveform modes are shown in Figure 21-2Figure 21-3.

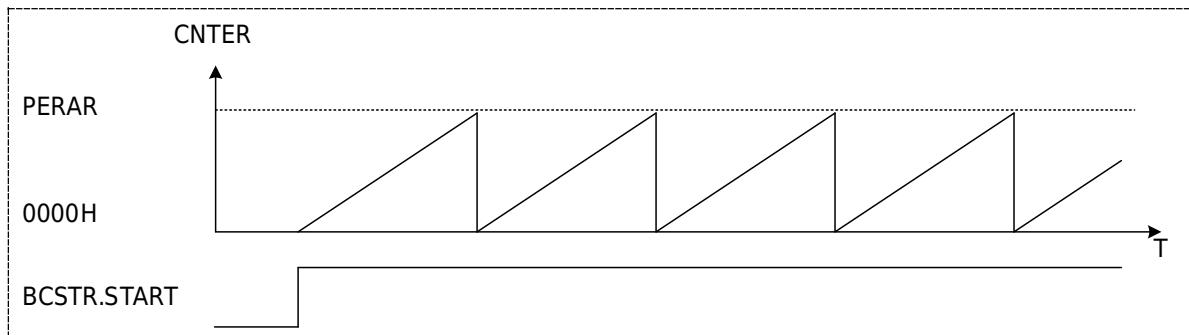


Figure 21-2 Sawtooth Waveform (Counting Up)

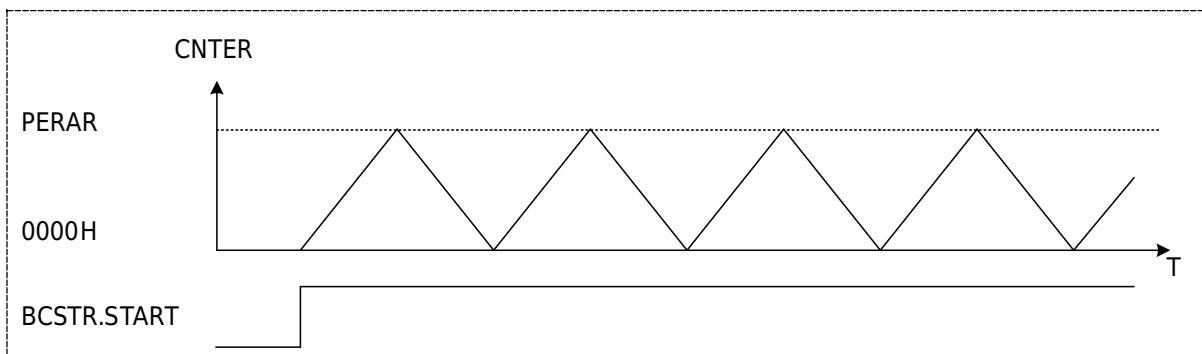


Figure 21-3 Triangle Waveform

21.3.1.2 Comparison Output

Each Timer A unit contains a 8-channel comparison output ($\text{TIMA}_{< t >} \text{ PWM}_n$) that outputs the specified level when the count value matches the baseline value. $\text{TMRA}_{< t >} \text{ CMPAR}_n$ register corresponds to the count baseline value of $\text{TIMA}_{< t >} \text{ PWM}_n$ output port, respectively. The $\text{TIMA}_{< t >} \text{ PWM}_n$ port outputs the specified level ($n = 1 \sim 8$) when the timer count is equal to $\text{TMRA}_{< t >} \text{ CMPAR}_n$.

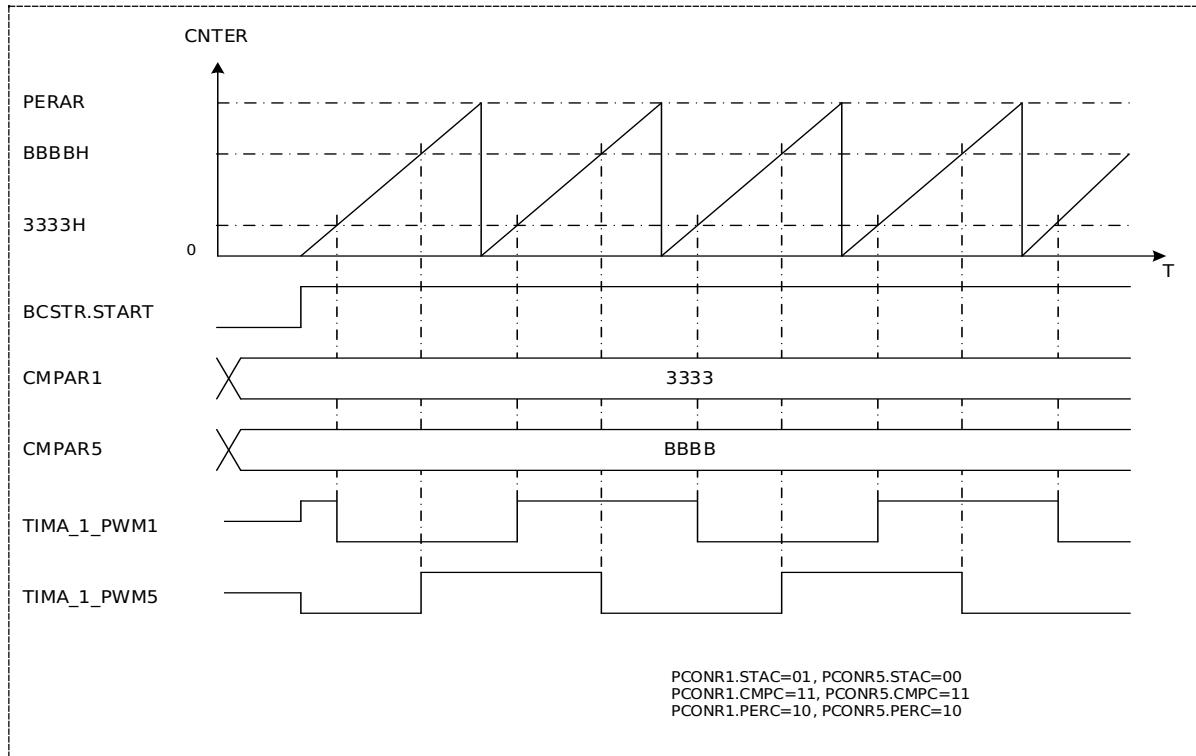


Figure 21-4 Compare Output Action

$\text{TIMA}_{< t >} \text{ PWM}_n$ port is controlled by port control register (PCONR n) STAC, STPC, CMPC, PERC, FORC bit setting ($n = 1 \sim 8$). Figure 21-4 shown an example of the comparison output operation of Unit 1.

21.3.1.3 Capture Input

Each PWM output channel of each Timer A 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. When the corresponding capture input condition is selected and is valid, the current count value is saved in the corresponding register (CMPARn) ($n = 1 \sim 8$).

The capture input condition can be selected from the internal capture action trigger event (selected by the HTSSR1 register), TIMA_<t>_PWMn port input, etc. The specific condition selection can be set by the HICP bit of the capture control register (CCONRn) ($n=1\sim8$). Figure 21-5 shows an example of the capture input action of Unit 2.

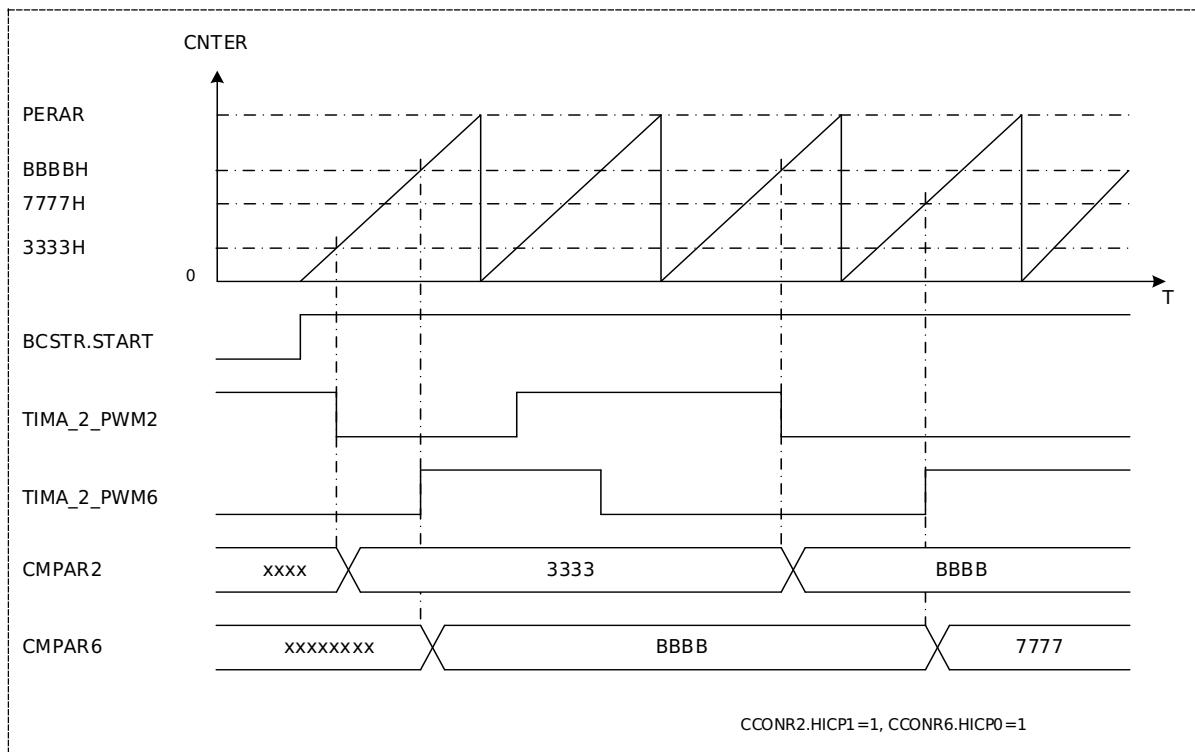


Figure 21-5 Capturing Input Actions

21.3.2 Timer Selection

TimerA's counting clock has the following options:

- a) 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 frequency division of PCLK1 (BCSTR.CKDIV[3:0] setting)
- b) TIMA_<t>_TRIG Port Event Input (HCUPR [9: 8] or HCDOR [9: 8] setting)
- c) Internal counter trigger event input (HCUPR [10] or HCDOR [10] settings)
- d) Count overflow or underflow event input for symmetric units (set in HCUPR [12: 11] or HCDOR [12: 11])
- e) TIMA_<t>_CLKA, TIMA_<t>_CLKB port orthogonal code input (HCUPR [7: 0] or HCDOR [7: 0] setting)

The counting timer is a software counting mode, and the counting timer is b, c, d, e hardware counting mode. When the counting clock selects d, it is mostly used in the revolution counting mode of three-phase quadrature encoding counting (see [Position overflow counting] and [Mixed counting] chapters), and it can also be used for cascade counting. 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.

21.3.3 Synchronous Startup

The 6-unit TimerA on this product can realize software synchronous start or hardware synchronous start. Unit 2~unit 6 can choose to start synchronously with unit 1. When the BCSTR.SYNST bit in unit 2~unit 6 is set to 1, the synchronous start function of the corresponding unit and unit 1 is valid. At this time, if the software sets the BCSTR.START bit of unit 1 to 1, the counter of the synchronized unit (unit 2~unit 6) starts software synchronous counting; if any bit in HCONR.HSTA1~0 of the hardware setting unit 1 is 1, and when the corresponding hardware event of unit 1 occurs, the counter of the synchronized unit (unit 2~unit 6) starts hardware synchronization counting. When the hardware synchronous counting start function is selected, the corresponding bits of HCONR.HSTA1~0 of the synchronized unit (unit 2~unit 6) must also be set to valid.

Figure 21-6 shows an example of software synchronous startup when BCSTR.SYNST=1 is set for Unit 2, Unit 4, and Unit 5.

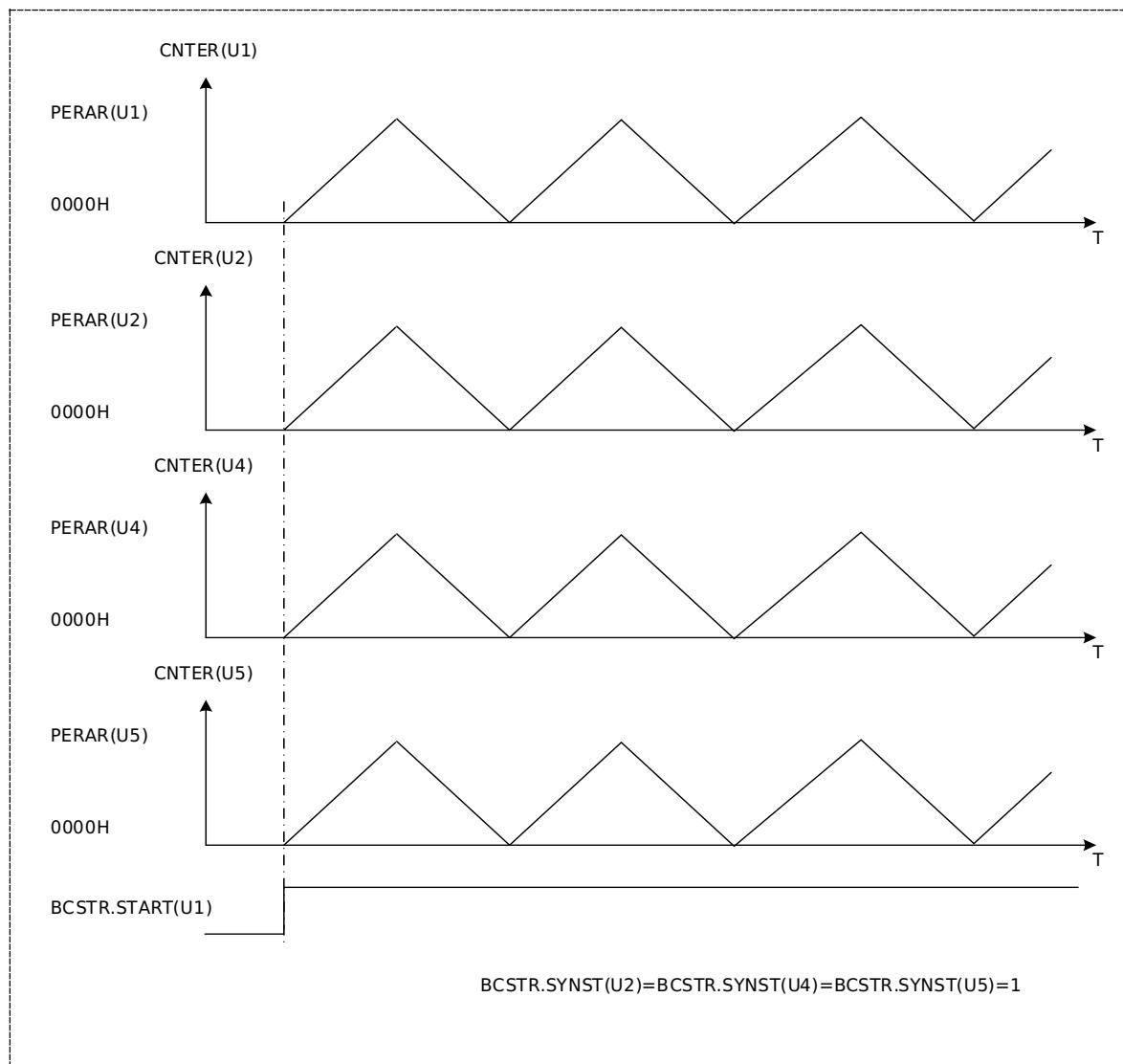


Figure 21-6 Software Synchronization Action

21.3.4 Digital Filtering

The input ports of TIMA_<t>_CLKA, TIMA_<t>_CLKB, TIMA_<t>_TRIG and TIMA_<t>_PWMn have digital filtering functions. The enable of filter function and the selection of filter clock can be realized by setting the corresponding bits of filter control register (FCONR) and capture control register (CCONRn) ($n = 1 \sim 8$).

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. Figure 21-7 shows an example of unit 1's CLKA port filter operation.

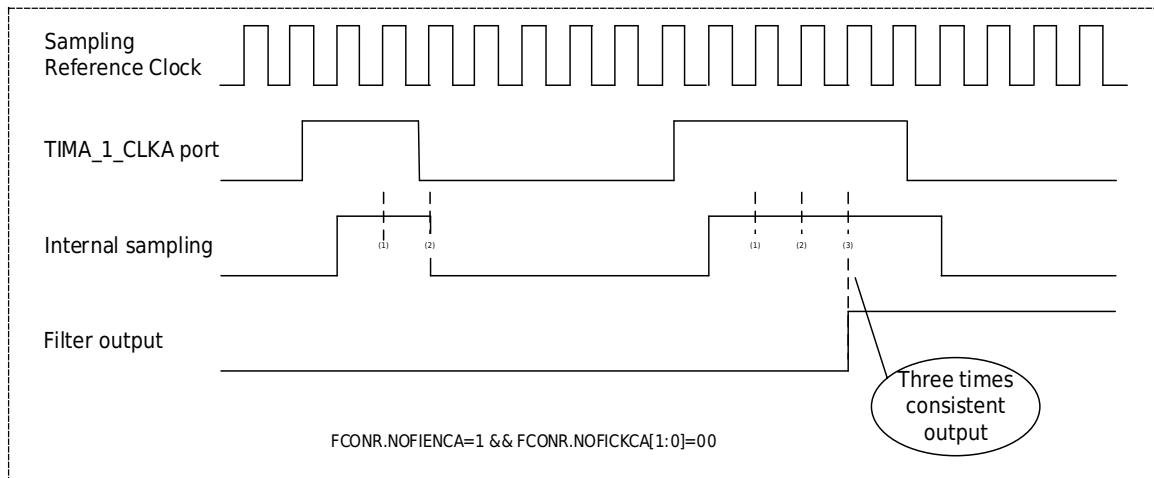


Figure 21-7 Filtering function of the clock input port

21.3.5 Cache Function

Two benchmark registers (CMPARn) of TimerA can implement cache function in pairs ($n = 1 \sim 8$). That is, CMPAR2 is used as the cache reference value of CMPAR1, CMPAR4 is used as the cache reference value of CMPAR3, CMPAR6 is used as the cache reference value of CMPAR5, and CMPAR8 is used as the cache reference value of CMPAR7. The cache control register (BCONRm) realizes the control of four sets of cache functions respectively ($m=1\sim4$).

When the BEN bit of the cache control register (BCONRm) is set, the cache function becomes valid ($m=1\sim4$). A buffer transfer occurs when the counter counts to a specific time point (CMPAR8/6/4/2->CMPAR7/5/3/1). 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)
- For sawtooth wave count mode (BCSTR.MODE = 0), the counter counts to the overflow point (BCSTR.DIR = 1) or to the underflow point (BCSTR.DIR = 0).
- When the triangle wave counting mode (BCSTR.MODE=1), count to the peak point (BCSTR.DIR=1 && BCONRn.BSE0=1) ($n=1\sim4$)
- When the triangle wave counting mode (BCSTR.MODE=1), count to the valley point (BCSTR.DIR=0 && BCONRn.BSE1=1) ($n=1\sim4$)
- Zeroing occurs when hardware counting mode or sawtooth wave counting mode is used.

The Figure 21-8 shows a schematic diagram of the buffer transmission of Unit 2 in sawtooth mode.

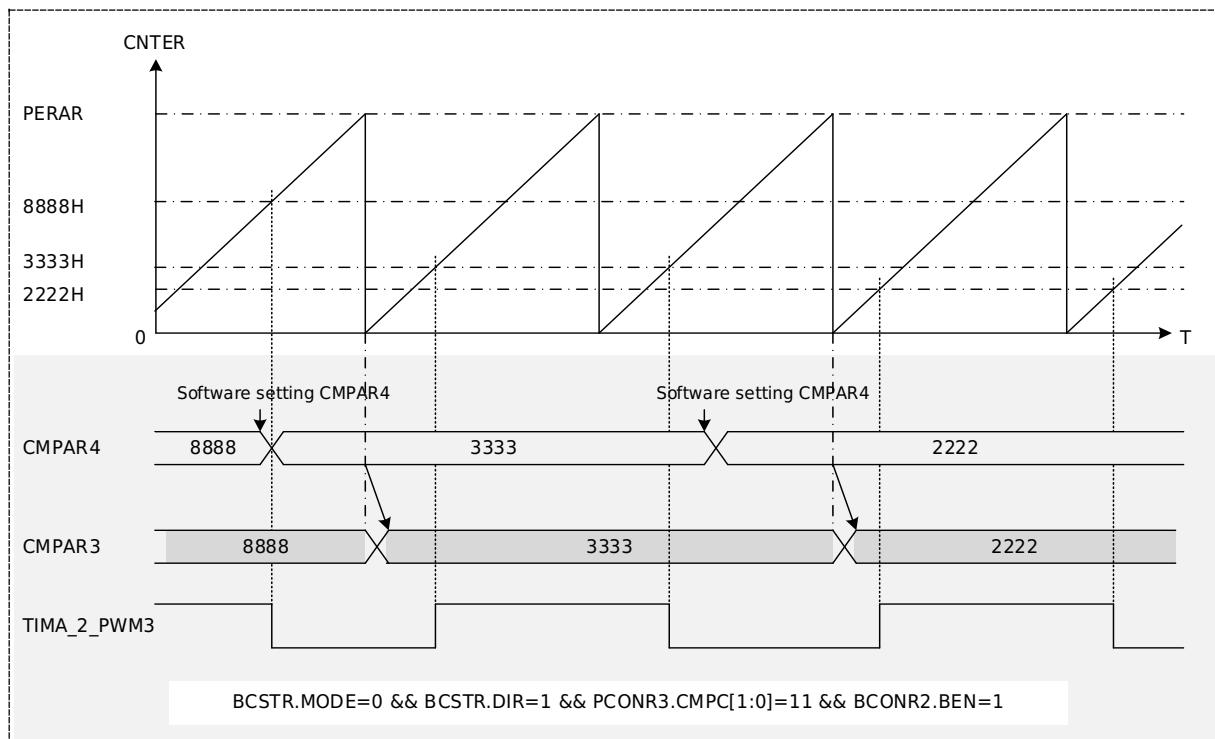


Figure 21-8 Cache action in sawtooth wave mode

21.3.6 Cascade Count

In the Count timer selection chapter, when timer is d), the count clock for this cell is selected as the count overflow (count overflow or count underflow) event for the symmetric cell, at which time the two cell levels combine and implement 32-bit counter. In the cascade counting, the CNTER of the symmetric unit is the low 16-bit counter and the CNTER of the unit is the high 16-bit counter.

For example, in the triangular-wave up-counting mode (BCSTR.MODE = 0, BCSTR.DIR = 1), the counting clock of unit 1 is set to PCLK1, the counting timer of unit 2 is set to the counting overflow event of unit 1 (TMRA_HCUPR.HCUP11 = 1 of unit 2), and the counting of units 1 and 2 (first unit 2, then unit 1) is implemented in cascade. Unit 1 has a low 16-bit counter at CNTER bit and Unit 2 has a high 16-bit counter at CNTER bit. Figure 21-9 shows a schematic diagram of unit 1, 2 cascade counting.

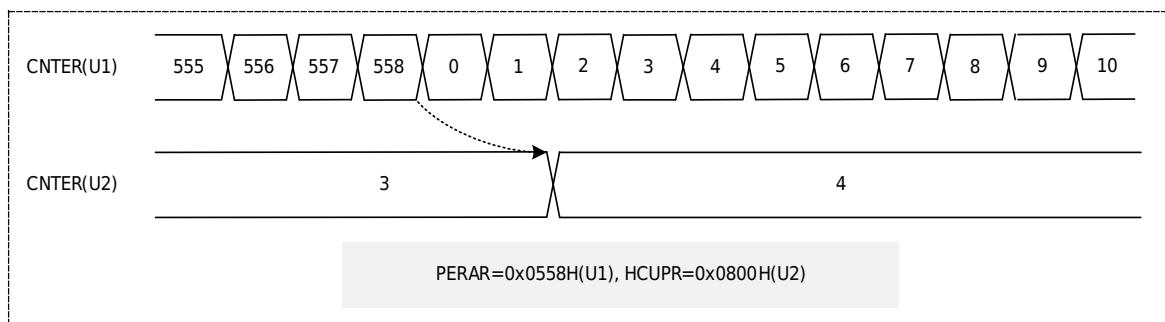


Figure 21-9 32-bit Cascade Counting Action

21.3.7 PWM Output

21.3.7.1 General PWM output

The 8-channel output TIMA_<t>_PWMn inside TimerA can realize different output waveforms ($n=1\sim 8$) through the relevant control bits of the port control register (PCONRn).

Figure 21-10 show an example of the PWM output waveforms of channels 1, 2, 6, and 7 in the triangle wave mode of unit 1.

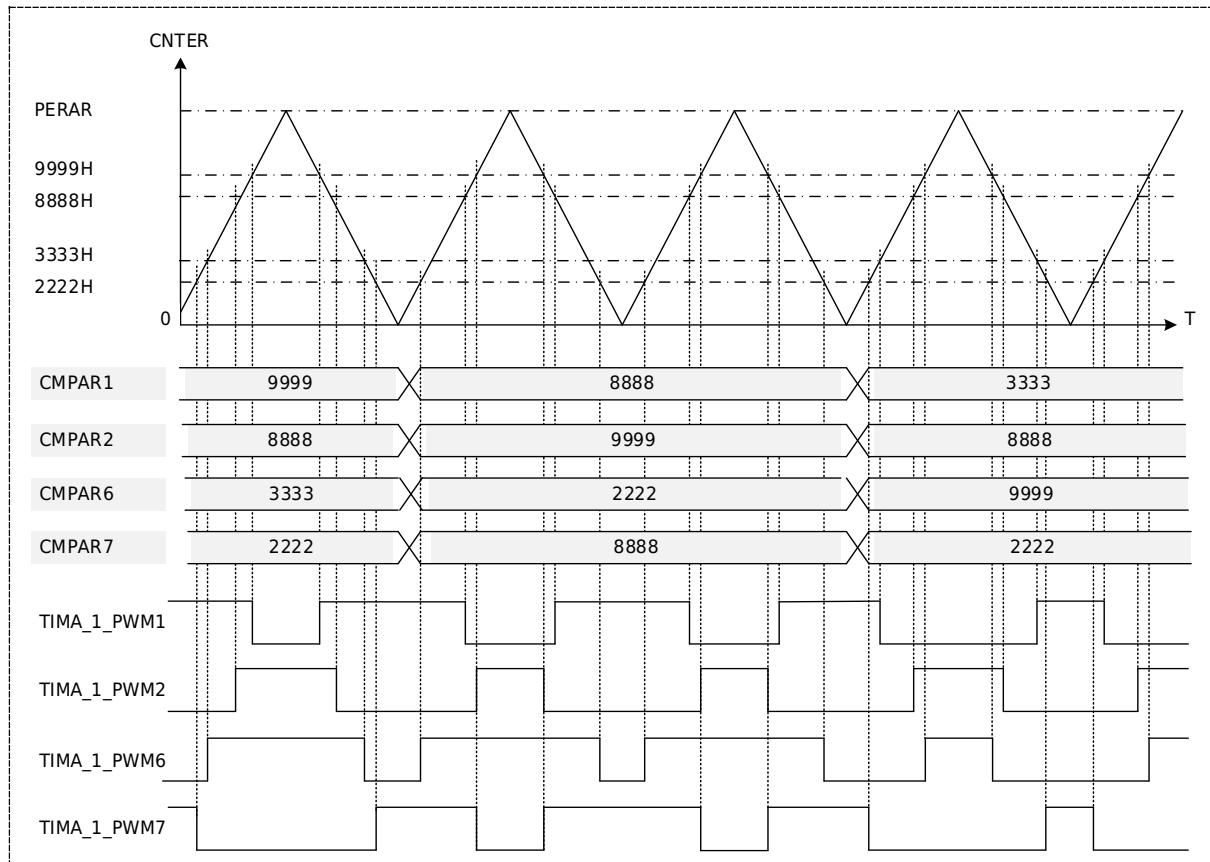


Figure 21-10 General PWM Output Example

21.3.7.2 Single Pulse PWM Output

In the sawtooth wave counting mode, it is possible to generate a single pulse within one cycle by setting the flip when the reference value compares and matches (PCONR.CMPC=11) and the flip when the period reference value compares and matches (PCONR.PERC=11). PWM output.

Figure 21-12 Shown is an example of a single-pulse PWM output waveform in the triangle wave mode of unit 1.

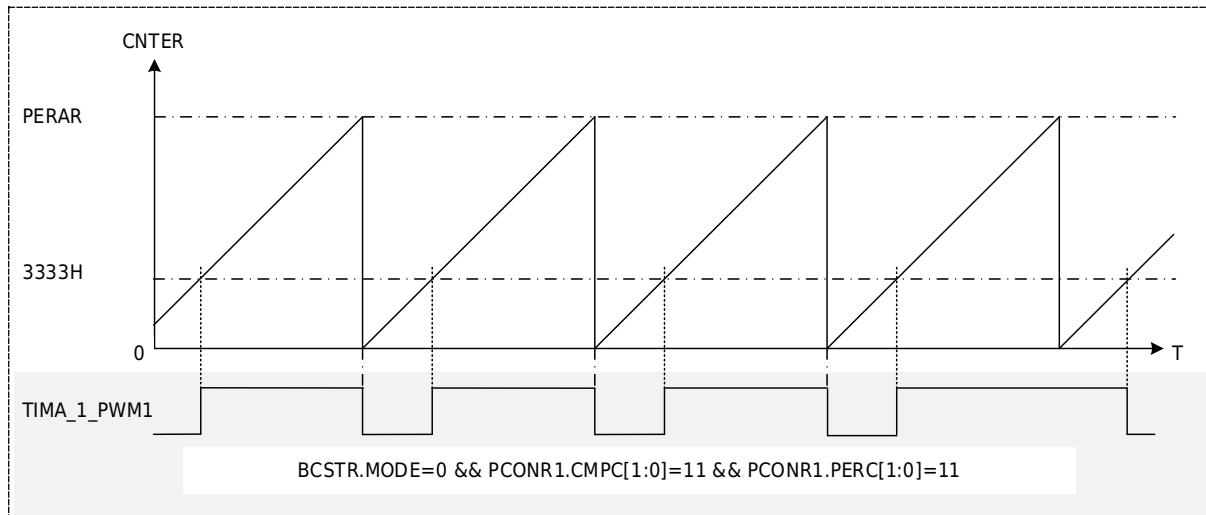


Figure 21-11 Single pulse PWM output

21.3.8 Orthogonal Coding Count

The TIMA_<t>_CLKA input is regarded as the AInput, the TIMA_<t>_CLKB input is regarded as the BInput, and the TIMA_<t>_TRIG input is regarded as the ZInput, and TimerA can realize the quadrature encoding count of 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 two units can realize the revolution counting mode, and the unit used for position counting is called position counting unit, which is used for revolution counting. The unit is called revolution counting unit. In revolution counting mode, every two units are combined with each other (combination of units 1 and 2; combination of units 3 and 4; combination of units 5 and 6), and the position counting unit and revolution counting unit in the combination can be specified arbitrarily.

The counting conditions of AIN and BIN are enabled by setting the orthogonal relationship between TIMA_<t>_CLKA and TIMA_<t>_CLKB in the hardware increment event selection register (HCUPR) and hardware decrement event selection register (HCDOR); ZIN's The input action realizes the clearing of the position timer by setting the clear enable bit of the hardware trigger event selection register (HCONR) of the position counting unit, and realizes the counting of the revolution timer by setting the hardware increment event selection register (HCUPR) of the revolution unit. .

21.3.8.1 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 the AIN or BIN port, as shown in Figure 21-11 below.

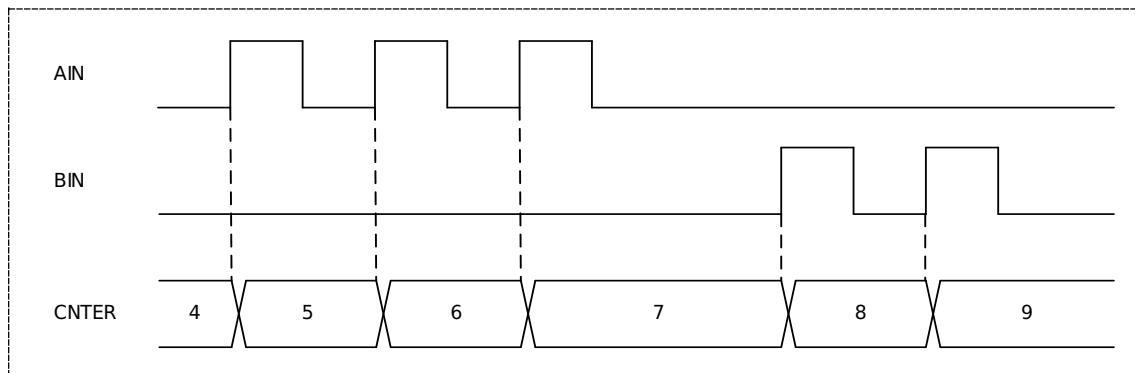


Figure 21-11 Position Mode - Basic Counting

Phase Difference Count

Phase difference counts are counted according to the phase relationship between AIN and BIN. According to different settings, 1-fold counting, 2-fold counting, 4-fold counting, etc. can be realized, as shown in the following Figure 21-12~Figure 21-14

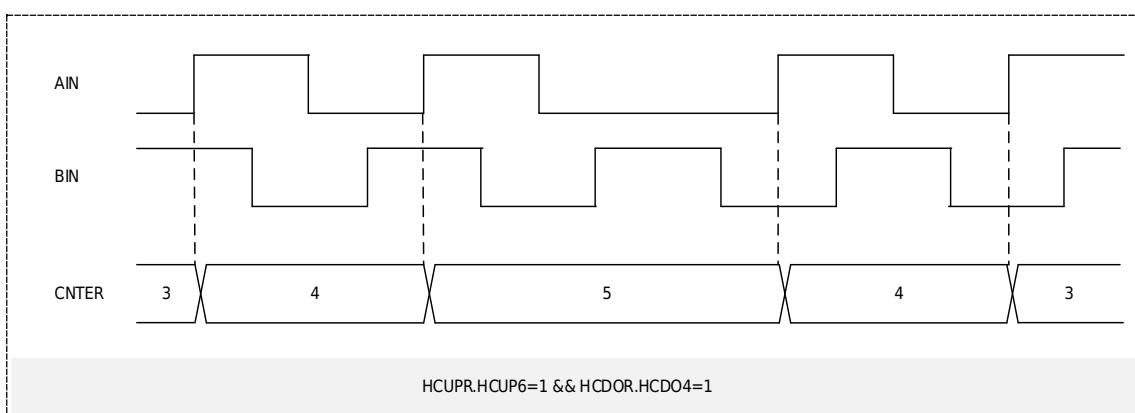


Figure 21-12 Position Counting Mode - Phase Difference Counting (1 Times Counting)

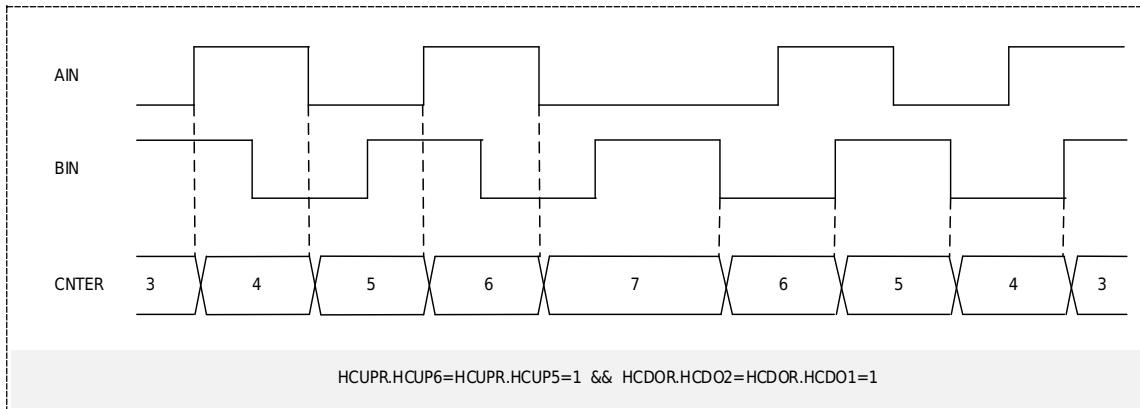


Figure 21-13 Position Counting Mode - Phase Difference Counting (2 Times Counting)

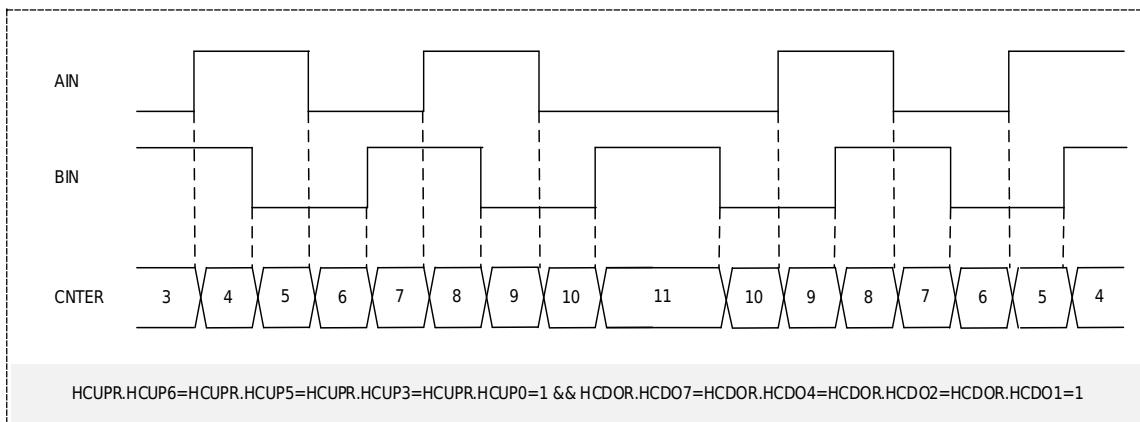


Figure 21-14 Position Counting Mode - Phase Difference Counting (4 Times Counting)

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 the Figure 21-15 below.

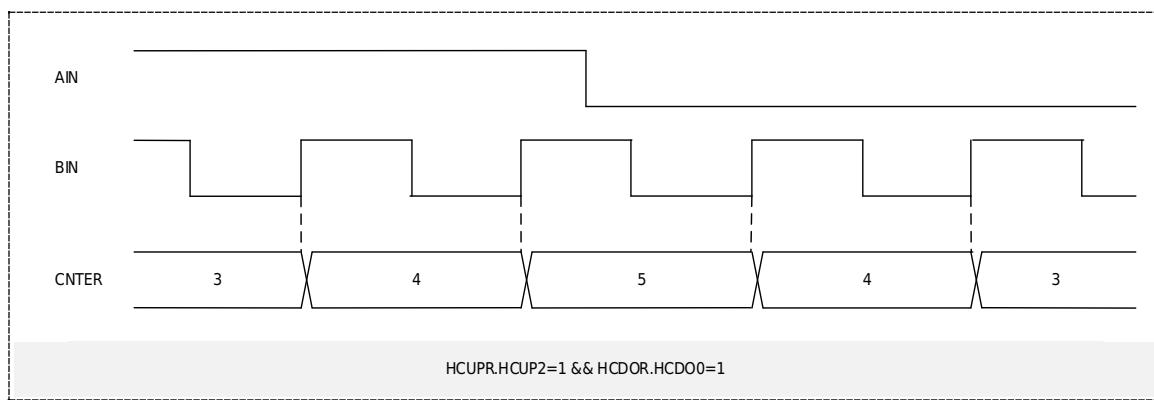


Figure 21-15 Position Counting Mode - Direction Counting

21.3.8.2 Revolution Counting Mode

Orthogonal encoding revolution counting mode refers to the addition of ZInput events on the basis of AIN and BIN counting to realize the judgment of the number of revolutions, etc. In the revolution counting mode, according to the counting method of the revolution timer, the Z-phase counting function, the position overflow counting function and the mixed counting function can be realized.

Z-Phase Count

Z-phase counting refers to the counting action that the revolution counting unit counts according to the input of ZIN, and the position counting unit is cleared at the same time. As follows:

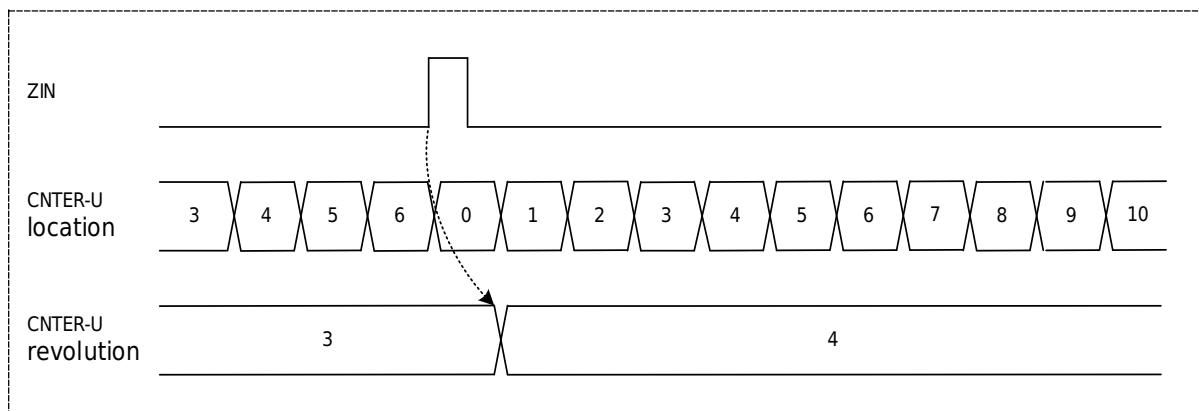


Figure 21-16 Revolution Counting Mode - Z Phase Counting

Position Overflow Count

Position overflow counting means that when the counting of the position counting unit overflows or underflows, an overflow event is generated, thereby triggering 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 bit12~11 of the hardware increment (decrement) event selection register (HCUPR or HCDOR) of the revolution counting unit is enabled, and the overflow event of the position counting unit can trigger the revolution counting unit to achieve a count. As shown in Figure 21-17 below.

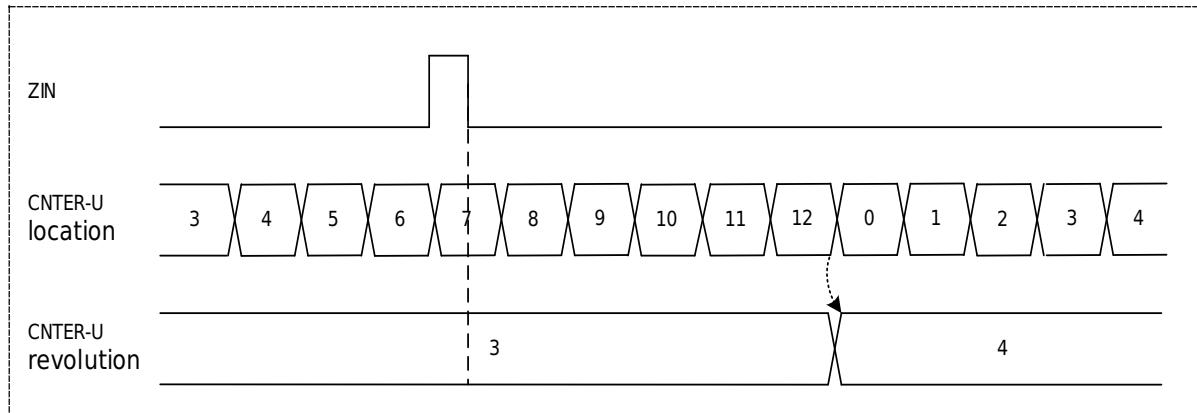


Figure 21-17 Revolution Counting Mode - Position Overflow Counting

Mixed Count

Mixed counting refers to the counting action that combines the above two counting methods of Z-phase counting and position overflow counting, and its realization method is also a combination of the above two counting methods. As shown in Figure 21-18 below.

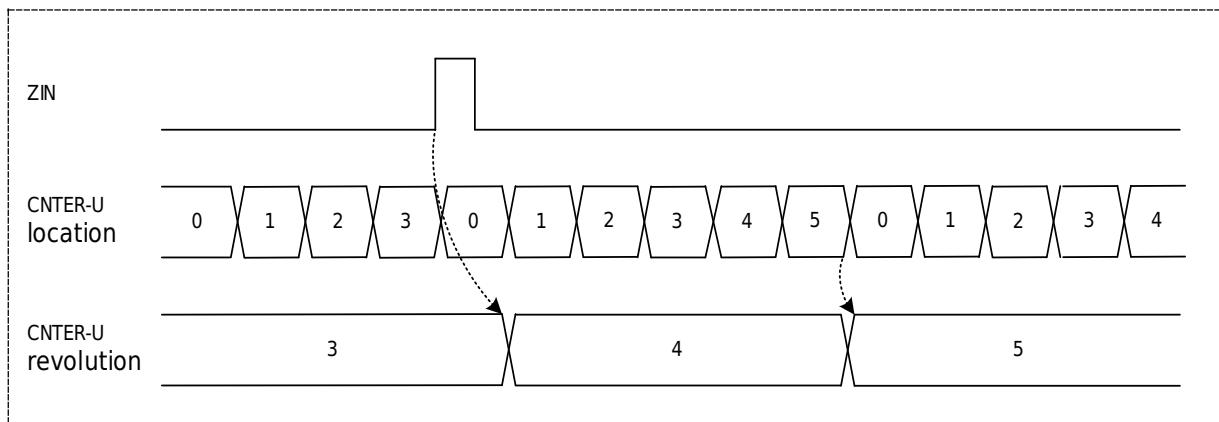


Figure 21-18 Revolution Counting Mode - Mixed Counting

21.4 Interrupt and Event Description

TimerA contains three interrupt outputs and three event outputs: One comparison match interrupt and event, two period match interrupt and event.

21.4.1 Compare Matched Interrupt and Events

The corresponding bit (STFLR.CMPFn) in the status flag register (STFLR) is set to 1 when the comparison of the baseline value register (CMPARn) and the count value matches. In this case, if the corresponding bit (ICONR.ITENn) of the interrupt control register (ICONR) is set to 1, the corresponding interrupt request (TMRA_U<t>_CMP) is triggered; If the corresponding bit (ECONR. ETENn) of the event control register (ECONR) is set to 1, the corresponding event request (TMRA_U<t>_CMP) is triggered (n = 1 ~ 8).

When the capture control register (CCONRn) selected capture input valid condition occurs, the capture input action occurs. In this case, if the corresponding bit (ICONR.ITENn) of the interrupt control register (ICONR) is set to 1, the corresponding interrupt request (TMRA_U<t>_CMP) is triggered; If the corresponding bit (ECONR. ETENn) of the event control register (ECONR) is set to 1, the corresponding event request (TMRA_U<t>_CMP) is triggered (n = 1 ~ 8).

Comparison-match interrupt and comparison-match events of the eight baseline values within each cell are not output independently. Comparison-match interrupt is aggregated into a interrupt module by OR logic (refer to INTC section), and comparison-match events are aggregated into an event output by OR logic to select another module to trigger.

21.4.2 Periodic Matching Interrupt and Events

The OVFF or UDFF bits of the control state register (BCSTR) are set to 1. In this case, if the BCSTR.ITENOVF or BCSTR.ITENUDF bit is set to 1 enable interrupt, the periodic matching interrupt (TMRA_U<t>_OVF and TMRA_U<t>_UDF) can be triggered at the corresponding periodic point to be output to the interrupt module (INTC). The periodic matching events (TMRA_U<t>_OVFand TMRA_U<t>_UDF) output is used to select and trigger other modules.

21.5 Register description

Table 21-3 show the register list of the TimerA module.

BASE ADDR:

0x4001_5000 (U1) , 0x4001_5400 (U2) , 0x4001_5800 (U3) ,

0x4001_5C00 (U4) , 0x4001_6000 (U5) , 0x4001_6400 (U6)

Table 21-3 Register List

Register name	Symbol	Offset	Bit width	Reset value
General count register	TMRA_CNTER	0x0000	16	0x0000
Cycle reference register	TMRA_PERAR	0x0004	16	0xFFFF
Compare Base Value Register 1	TMRA_CMPAR1	0x0040	16	0xFFFF
Compare Base Value Register 2	TMRA_CMPAR2	0x0044	16	0xFFFF
Compare Base Value Register 3	TMRA_CMPAR3	0x0048	16	0xFFFF
Compare Base Value Register 4	TMRA_CMPAR4	0x004C	16	0xFFFF
Compare Base Value Register 5	TMRA_CMPAR5	0x0050	16	0xFFFF
Compare Base Value Register 6	TMRA_CMPAR6	0x0054	16	0xFFFF
Compare Base Value Register 7	TMRA_CMPAR7	0x0058	16	0xFFFF
Compare Base Value Register 8	TMRA_CMPAR8	0x005C	16	0xFFFF
Control state	TMRA_BCSTR	0x0080	16	0x0002
Interrupt control register	TMRA_ICONR	0x0090	16	0x0000
Event control register	TMRA_ECONR	0x0094	16	0x0000
Filter control register	TMRA_FCONR	0x0098	16	0x0000
Status flag register	TMRA_STFLR	0x009C	16	0x0000
Cache Control Register 1	TMRA_BCONR1	0x00C0	16	0x0000
Cache Control Register 2	TMRA_BCONR2	0x00C8	16	0x0000
Cache Control Register 3	TMRA_BCONR3	0x00D0	16	0x0000
Cache Control Register 4	TMRA_BCONR4	0x00D8	16	0x0000
Capture Control Register 1	TMRA_CCONR1	0x0100	16	0x0000
Capture Control Register 2	TMRA_CCONR2	0x0104	16	0x0000
Capture Control Register 3	TMRA_CCONR3	0x0108	16	0x0000
Capture Control Register 4	TMRA_CCONR4	0x010C	16	0x0000
Capture Control Register 5	TMRA_CCONR5	0x0110	16	0x0000
Capture Control Register 6	TMRA_CCONR6	0x0114	16	0x0000
Capture Control Register 7	TMRA_CCONR7	0x0118	16	0x0000
Capture Control Register 8	TMRA_CCONR8	0x011C	16	0x0000
Port Control Register 1	TMRA_PCONR1	0x0140	16	0x0000
Port Control Register 2	TMRA_PCONR2	0x0144	16	0x0000
Port Control Register 3	TMRA_PCONR3	0x0148	16	0x0000

Register name	Symbol	Offset	Bit width	Reset value
Port Control Register 4	TMRA_PCONR4	0x014C	16	0x0000
Port Control Register 5	TMRA_PCONR5	0x0150	16	0x0000
Port Control Register 6	TMRA_PCONR6	0x0154	16	0x0000
Port Control Register 7	TMRA_PCONR7	0x0158	16	0x0000
Port Control Register 8	TMRA_PCONR8	0x015C	16	0x0000
Hardware trigger event selection register	TMRA_HCONR	0x0084	16	0x0000
Hardware Adder Event Selection Register	TMRA_HCUPR	0x0088	16	0x0000
Hardware Decrease Event Selection Register	TMRA_HCDOR	0x008C	16	0x0000

21.5.1 Generic Count Value Register (TMRA_CNTER)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNT[15:0]															
Bit	Marking		Place name			Function									Read and write
b15~b0	CNT[15:0]		Count value			Current timer count									R/W

21.5.2 Periodic Reference Value Register (TMRA_PERAR)

Reset value: 0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PER[15:0]															
Bit	Marking		Place name			Function									Read and write
b15~b0	PER[15:0]		Count cycle value			Set the count cycle value for each round									R/W

21.5.3 Comparison Base Value Register (TMRA_CMPAR1~8)

Reset value: 0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CMP[15:0]															
Bit	Marking		Place name			Function									Read and write
b15~b0	CMP[15:0]		Count baseline			Set baseline									R/W

21.5.4 Control State Register (TMRA _ BCSTR)

Reset value: 0x0002

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UDF F	OVF F	ITEN UDF	ITEN OVF	-	-	-	-	CKDIV[3:0]		SYN ST	MODE	DIR	START		
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b15	UDFF	Underflow mark	0: Count down, no count down 1: Count down, count down	R/W											
b14	OVFF	Overflow mark	0: When counting up, no overflow occurs 1: When counting up, the count overflows	R/W											
b13	ITENUDF	Underflow interrupt enable	0: Count underflow interrupt not enabled 1: Count underflow interrupt enable	R/W											
b12	ITENOVF	Overflow interrupt enable	0: Count overflow interrupt not enabled 1: Count overflow interrupt enable	R/W											
b11~b8	Reserved	-	Read as "0", write as "0"	R/W											
<hr/>															
b7~b4	CKDIV[3:0]	Counting clock selection	0000: PCLK 0001: PCLK/2 0010: PCLK/4 0011: PCLK/8 0100: PCLK/16 0101: PCLK/32 0110: PCLK/64 0111: PCLK/128 1000: PCLK/256 1001: PCLK/512 1010: PCLK/1024	R/W											
b3	SYNST	Synchronous enable	0: The synchronous start function with unit 1 is invalid 1: The synchronous start function with unit 1 is valid Note: The setting of this bit of unit 1 is invalid, and it is "0" when read	R/W											
b2	MODE	Counting mode	0: Sawtooth mode 1: Triangular wave mode	R/W											
b1	DIR	Counting direction	0: Counter down 1: Counter up	R/W											
b0	START	Timer start	0: Timer off 1: Timer Startup Note 1: This bit will automatically change to 0 if the hardware stop condition is valid Note 2: When the synchronous start function of units 2~6 is valid, this bit of the corresponding unit will also be set after the software of unit 1 is started	R/W											

21.5.5 Interrupt Control Register (TMRA_ICONR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								IT EN8	IT EN7	IT EN6	IT EN5	IT EN4	IT EN3	IT EN2	IT EN1
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b15~b8	Reserved	-	Read as "0", write as "0"										R/W		
b7	ITEN8	Count Match interrupt enable 8	0:CMPAR8 register is invalid when it is equal to the count value or when a capture input event occurs 1: CMPar8 This interrupt is enabled when CMPar1register is equal to the count value, or when a capture input event occurs										R/W		
b6	ITEN7	Count Match interrupt enable 7	0:CMPar7 register is invalid when it is equal to the count value or when a capture input event occurs 1: CMPar7 This interrupt is enabled when CMPar1register is equal to the count value, or when a capture input event occurs										R/W		
b5	ITEN6	Count Match interrupt enable 6	0:CMPar6 register is invalid when it is equal to the count value or when a capture input event occurs 1: CMPar6 This interrupt is enabled when CMPar1register is equal to the count value, or when a capture input event occurs										R/W		
b4	ITEN5	Count Match interrupt enable 5	0:CMPar5 register is invalid when it is equal to the count value or when a capture input event occurs 1: CMPar5 This interrupt is enabled when CMPar1register is equal to the count value, or when a capture input event occurs										R/W		
b3	ITEN4	Count Match interrupt enable 4	0:CMPar4 register is invalid when it is equal to the count value or when a capture input event occurs 1: CMPar4 This interrupt is enabled when CMPar1register is equal to the count value, or when a capture input event occurs										R/W		
b2	ITEN3	Count Match interrupt enable 3	0:CMPar3 register is invalid when it is equal to the count value or when a capture input event occurs 1: CMPar3 This interrupt is enabled when CMPar1register is equal to the count value, or when a capture input event occurs										R/W		
b1	ITEN2	Count Match interrupt enable 2	0:CMPar2 register is invalid when it is equal to the count value or when a capture input event occurs 1: CMPar2 This interrupt is enabled when CMPar1register is equal to the count value, or when a capture input event occurs										R/W		
b0	ITEN1	Count Match interrupt enable 1	0:CMPar1 register is invalid when it is equal to the count value or when a capture input event occurs 1: CMPar1 This interrupt is enabled when CMPar1register is equal to the count value, or when a capture input event occurs										R/W		

21.5.6 Event Control Register (TMRA_ECONR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								ET EN8	ET EN7	ET EN6	ET EN5	ET EN4	ET EN3	ET EN2	ET EN1
Bit	Marking	Place name		Function										Read and write	
b15~b8	Reserved	-		Read as "0", write as "0"										R/W	
b7	ETEN8	Count Match Event Enabled 8		0: CMPAR8 register output is invalid when it is equal to a count value or when a capture input event occurs 1: When CMPAR8 register is equal to the count value or a capture input event occurs, the event output is enabled										R/W	
b6	ETEN7	Count Match Event Enabled 7		0: CMPAR7 register output is invalid when it is equal to a count value or when a capture input event occurs 1: When CMPAR7 register is equal to the count value or a capture input event occurs, the event output is enabled										R/W	
b5	ETEN6	Count Match Event Enabled 6		0: CMPAR6 register output is invalid when it is equal to a count value or when a capture input event occurs 1: When CMPAR6 register is equal to the count value or a capture input event occurs, the event output is enabled										R/W	
b4	ETEN5	Count Match Event Enabled 5		0: CMPAR5 register output is invalid when it is equal to a count value or when a capture input event occurs 1: When CMPAR5 register is equal to the count value or a capture input event occurs, the event output is enabled										R/W	
b3	ETEN4	Count Match Event Enabled 4		0: CMPAR4 register output is invalid when it is equal to a count value or when a capture input event occurs 1: When CMPAR4 register is equal to the count value or a capture input event occurs, the event output is enabled										R/W	
b2	ETEN3	Count Match Event Enabled 3		0: CMPAR3 register output is invalid when it is equal to a count value or when a capture input event occurs 1: When CMPAR3 register is equal to the count value or a capture input event occurs, the event output is enabled										R/W	
b1	ETEN2	Count Match Event Enabled 2		0: CMPAR2 register output is invalid when it is equal to a count value or when a capture input event occurs 1: When CMPAR2 register is equal to the count value or a capture input event occurs, the event output is enabled										R/W	
b0	ETEN1	Count Match Event Enabled 1		0: CMPAR1 register output is invalid when it is equal to a count value or when a capture input event occurs 1: When CMPAR1 register is equal to the count value or a capture input event occurs, the event output is enabled										R/W	

21.5.7 Filter Control Register (TMRA_FCONR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	NOFI CKCB[1:0]	NOFI ENCB	-	NOFI CKCA[1:0]	NOFI ENCA	-	-	-	-	-	-	NOFI CKTG[1:0]	NOFI ENTG		
Bit	Marking	Place name	Function										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 filter CB	0: Invalid input filtering function for TIMA_<t>_CLKB port 1: TIMA_<t>_CLKB port input filtering enabled										R/W		
b11	Reserved	-	Read as "0", write as "0"										R/W		
b10~b9	NOFICKCA[1:0]	Filter sampling reference clock selection CA	00: PCLK 01: PCLK/4 10: PCLK/16 11: PCLK/64										R/W		
b8	NOFIENCA	Capture input port filter CA	0: Invalid input filtering function of TIMA_<t>_CLKA port 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 input port filter invalid 1: TIMA_<t>_TRIG input port filtering function enabled										R/W		

21.5.8 Status Flag Register (TMRA _ STFLR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								CMP F8	CMP F7	CMP F6	CMP F5	CMP F4	CMP F3	CMP F2	CMP F1
Bit	Marking	Place name		Function										Read and write	
b15-b8	Reserved	-		Read as "0", write as "0"										R	
b7	CMPF8	Count Match Flags 8		0: The value of the CMPAR8 register is not equal to the count value, and the TIMA_<t>_PWM8 capture completion action has not occurred 1: The value of the CMPAR8 register is equal to the count value, or a TIMA_<t>_PWM8 capture completion action occurs										R/W	
b6	CMPF7	Count Match Flags 7		0: The value of the CMPAR7 register is not equal to the count value, and the TIMA_<t>_PWM7 capture completion action has not occurred 1: The value of the CMPAR7 register is equal to the count value, or a TIMA_<t>_PWM7 capture completion action occurs										R/W	
b5	CMPF6	Count Match Flags 6		0: The value of the CMPAR6 register is not equal to the count value, and the TIMA_<t>_PWM6 capture completion action has not occurred 1: The value of the CMPAR6 register is equal to the count value, or a TIMA_<t>_PWM6 capture completion action occurs										R/W	
b4	CMPF5	Count Match Flags 5		0: The value of the CMPAR5 register is not equal to the count value, and the TIMA_<t>_PWM5 capture completion action has not occurred 1: The value of the CMPAR5 register is equal to the count value, or a TIMA_<t>_PWM5 capture completion action occurs										R/W	
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 a 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 a TIMA_<t>_PWM3 capture completion action occurs										R/W	
b1	CMPF2	Count Match Flags 2		0: The value of CMPAR2register is not equal to the value of the count, and no TIMA_<t>_PWM2 capture completion action occurs 1: CMPAR2register is equal to the count value or TIMA_<t>_PWM2 captures complete action										R/W	
b0	CMPF1	Count Match Flags 1		0: The value of CMPAR1register is not equal to the value of the count, and no TIMA_<t>_PWM1 capture completion action occurs 1: CMPAR1register is equal to the count value or TIMA_<t>_PWM1 captures complete action										R/W	

21.5.9 Cache Control Register (TMRA_BCONR1~4)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
Reserved														BSE1	BSE0	BEN	
Bit	Marking		Place name		Function										Read and write		
b15~b3	Reserved		-		Read as "0", write as "0"										R/W		
b2	BSE1		Triangulation caching option 1		0: Cached value does not pass when triangulation mode counts to valley point 1: When the triangulation mode counts to the valley point, the cached value is transferred, i.e.: CMMARm -> CMPARn (m=2,4,6,8, n=1,3,5,7)										R/W		
b1	BSE0		Triangulation caching option 0		0: Cached value does not pass when triangulation mode counts to peak. 1: When the triangulation mode counts to the peak, the cached value is transferred, i.e.: CMMARm -> CMPARn (m=2,4,6,8, n=1,3,5,7)										R/W		
b0	BEN		Cache enable		0: Invalid Cache Function for CMPARn Benchmark 1: The Cache Function of CMPARn Benchmark (n=1,3,5,7)										R/W		

21.5.10 Capture Control Register (TMRA_CCONR1~8)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	NOFI CKCP[1:0]	NOFI ENCP	-	-	HICP 4	HICP 3	-	HICP 2	HICP 1	HICP 0	-	-	-	CAP MD	

Bit	Marking	Place name	Function	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: Invalid input filtering function of TIMA_<t>_PWMn port 1: TIMA_<t>_PWMn port input filtering enabled (n=1~8)	R/W
b11~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	HICP4	Capture input condition enable 4	0: When the TIMA_<t>_TRIG port input is sampled to the falling edge, channel m+1 does not capture the input action 1: When the TIMA_<t>_TRIG port input is sampled to the falling edge, the channel m+1 generates a capture input action Note: This bit is valid only for 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 TIMA_<t>_TRIG port input is sampled to the rising edge, the channel m+1 does not capture the input action 1: When the TIMA_<t>_TRIG port input is sampled to a rising edge, channel m+1 will generate a capture input action Note: This bit is valid only for 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 the event specified in the TMRA_HTSSR1 register occurs, no capture input action occurs 1: When the event specified in the TMRA_HTSSR1 register occurs, a capture input action is generated	R/W
b5	HICP1	Capture input condition enable 1	0: No capture input action occurs when TIMA_<t>_PWMn port input is sampled to the falling edge 1: When the input of TIMA_<t>_PWMn port is sampled to the falling edge, the capture input action is generated. (n=1~8)	R/W
b4	HICP0	Capture input condition enable 0	0: No capture input action occurs when TIMA_<t>_PWMn port input is sampled to the rising edge 1: When the input of TIMA_<t>_PWMn port reaches the rising edge, the capture input action is generated. (n=1~8)	R/W
b3~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	CAPMD	Functional mode selection	0: Comparison output function 1: Capture input function	R/W

21.5.11 Port Control Register (TMRA_PCONR1~8)

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~b13	Reserved				Read as "0", write as "0"										R/W	
b12	OUTEN		Output enable		0: Invalid output of TIMA_<t>_PWMn port when PWM output function 1: Output Validity of TIMA_<t>_PWMn Port with PWM Output (n=1~8)										R/W	
b11~b10	Reserved				Read as "0", write as "0"										R/W	
b9~b8	FORC[1:0]		Forced port status setting		0x: Set invalid 10: Starting from next cycle, TIMA_<t>_PWMn port output is set to low 11: Starting next cycle, the output of TIMA_<t>_PWMn port is set to high Note 1: The next cycle refers to the hardware counting mode or the zigzag 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: TIMA_<t>_PWMn port output is set to low when count is equal to PERAR 01: TIMA_<t>_PWMn port output is set to high when count is equal to PERAR 10: TIMA_<t>_PWMn port output remains in the previous state when the count value is equal to PERAR 11: TIMA_<t>_PWMn port output is set to inverted level when count is equal to PERAR (n=1~8)										R/W	
b5~b4	CMPC[1:0]		Port status setting when comparison values match		00: TIMA_<t>_PWMn port output is set to low when the count value is equal to CMPARn 01: When the count value is equal to CMPARn, the TIMA_<t>_PWMn port output is set to high 10: When the count value is equal to CMPARn, the TIMA_<t>_PWMn port output remains in the previous state 11: TIMA_<t>_PWMn port output is set to inverted level when the count value is equal to CMPARn (n=1~8)										R/W	
b3~b2	STPC[1:0]		Port status setting when count stops		00: TIMA_<t>_PWMn port output set to low level when count stops 01: When the count stops, the TIMA_<t>_PWMn port output is set to high 10: When the count stops, the TIMA_<t>_PWMn port output remains in the previous state 11: When the count stops, the TIMA_<t>_PWMn port output remains in the previous state (n=1~8)										R/W	
b1~b0	STAC[1:0]		Port status setting at start of count		00: At the start of the count, the TIMA_<t>_PWMn port output is set to low 01: At the beginning of the count, the TIMA_<t>_PWMn port output is set to high 10: At the beginning of the count, the TIMA_<t>_PWMn port output remains in the previous state 11: At the beginning of the count, the TIMA_<t>_PWMn port output remains in the previous state (n=1~8) Note: This bit setting is only valid under the condition of no frequency division (BCSTR.CKDIV=4'h0), please set it to 2'b10 or 2'b11 for other frequency divisions										R/W	

21.5.12 Hardware Trigger Event Selection Register (TMRA_HCONR)

Reset value: 0x0000

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	Marking	Place name	Function	Read and write
b15	HCLE6	Hardware zeroing condition 6	Condition: TIMA_<t>_PWM3 port input sampling to falling edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b14	HCLE5	Hardware zeroing condition 5	Condition: TIMA_<t>_PWM3 port input sampled to rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b13	HCLE4	Hardware zeroing condition 4	Condition: when the unit is unit m, the TRIG port input of unit n is sampled to the falling edge (when m=1, 3, 5, n=2, 4, 6; when m=2, 4, 6, n= 1, 3, 5) 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b12	HCLE3	Hardware zeroing condition 3	Condition: When the unit is unit m, the input of the TRIG port of unit n is sampled to the rising edge (when m=1, 3, 5, n=2, 4, 6; when m=2, 4, 6, n= 1, 3, 5) 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b11	Reserved	-	Read as "0", write as "0"	R/W
b10	HCLE2	Hardware zeroing condition 2	Condition: The event specified in the TMRA_HTSSR0 register occurs 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b9	HCLE1	Hardware zeroing condition 1	Condition: TIMA_TRIG port input to drop edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b8	HCLE0	Hardware zeroing condition 0	Condition: TIMA_TRIG port input sampled to rising edge 0: Hardware zeroing is invalid when conditions match 1: Hardware zeroing is valid when conditions match	R/W
b7	Reserved	-	Read as "0", write as "0"	R/W
b6	HSTP2	Hardware stop condition 2	Condition: The event specified in the TMRA_HTSSR0 register occurs 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b5	HSTP1	Hardware stop condition 1	Condition: TIMA_<t>_TRIG port input to drop edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b4	HSTP0	Hardware stop condition 0	Condition: TIMA_<t>_TRIG port input sampling to rising edge 0: Hardware stop invalid when condition matches 1: Hardware stops working when conditions match	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2	HSTA2	Hardware startup condition 2	Condition: The event specified in the TMRA_HTSSR0 register occurs 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	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) TIMA_1_TRIG port input sample to falling edge (sync start function is valid) 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match Note: Condition 2) Only units 2~6 can be selected, unit 1 is invalid	R/W
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) TIMA_1_TRIG port input sample to rising edge (sync start function is valid) 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match Note: Condition 2) Only units 2~6 can be selected, unit 1 is invalid	R/W

21.5.13 Hardware Added Event Selection Register (TMRA_HCUPR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	HC UP12	HC UP11	HC UP10	HC UP9	HC UP8	HC UP7	HC UP6	HC UP5	HC UP4	HC UP3	HC UP2	HC UP1	HC UP0

Bit	Marking	Place name	Function	Read and write
b15~b13	Reserved	-	Read as "0", write as "0"	R/W
b12	HCUP12	Hardware Added Condition 12	Condition: When this unit is unit m, counting underflow occurs in unit n (when m=1, 3, 5, n=2, 4, 6; when m=2, 4, 6, n=1, 3, 5) 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b11	HCUP11	Hardware Added Condition 11	Condition: When this unit is unit m, count overflow occurs in unit n (when m=1, 3, 5, n=2, 4, 6; when m=2, 4, 6, n=1, 3, 5) 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b10	HCUP10	Hardware Added Condition 10	Condition: The event specified in the TMRA HTSSR0 register occurs 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b9	HCUP9	Hardware Added Condition 9	Condition: TIMA_<t>_TRIG port samples up to falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b8	HCUP8	Hardware Added Condition 8	Condition: TIMA_<t>_TRIG port samples up to rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b7	HCUP7	Hardware Added Condition 7	Condition: When the TIMA_<t>_CLKA port is at high level, the TIMA_<t>_CLKA port is sampled to the falling edge. 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b6	HCUP6	Hardware Added Condition 6	Condition: When the TIMA_<t>_CLKA port is at a high level, the TIMA_<t>_CLKA port is sampled up to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b5	HCUP5	Hardware Added Condition 5	Condition: When the TIMA_<t>_CLKA port is at low level, the TIMA_<t>_CLKA port is sampled to the falling edge. 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b4	HCUP4	Hardware Added Condition 4	Condition: When the TIMA_<t>_CLKA port is at low level, the TIMA_<t>_CLKA port is sampled up to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b3	HCUP3	Hardware Added Condition 3	Condition: When the TIMA_<t>_CLKB port is at high level, the TIMA_<t>_CLKB port is sampled up to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b2	HCUP2	Hardware Added Condition 2	Condition: When the TIMA_<t>_CLKB port is at high level, the TIMA_<t>_CLKB port is sampled up to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b1	HCUP1	Hardware Added Condition 1	Condition: When the TIMA_<t>_CLKA port is at low level, the TIMA_<t>_CLKA port is sampled up to the falling edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W
b0	HCUP0	Hardware Added Condition 0	Condition: When the TIMA_<t>_CLKA port is at low level, the TIMA_<t>_CLKB port is sampled up to the rising edge 0: Hardware add invalid when condition matches 1: Hardware addition is valid when a condition matches	R/W

21.5.14 Hardware Decrease Event Select Register (TMRA_HCDOR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	HC DO12	HC DO11	HC DO10	HC DO9	HC DO8	HC DO7	HC DO6	HC DO5	HC DO4	HC DO3	HC DO2	HC DO1	HC DO0
Bit	Marking	Place name	Function												Read and write
b15~b13	Reserved	-	Read as "0", write as "0"												R/W
b12	HCDO12	Hardware decrement condition 12	Condition: When this unit is unit m, counting underflow occurs in unit n (when m=1, 3, 5, n=2, 4, 6; when m=2, 4, 6, n=1, 3, 5) 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b11	HCDO11	Hardware decrement condition 11	Condition: When this unit is unit m, count overflow occurs in unit n (when m=1, 3, 5, n=2, 4, 6; when m=2, 4, 6, n=1, 3, 5) 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b10	HCDO10	Hardware decrement condition 10	Condition: The event specified in the TMRA_HTSSR0 register occurs 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b9	HCDO9	Hardware decrement condition 9	Condition: TIMA_<t>_TRIG port samples up to falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b8	HCDO8	Hardware decrement condition 8	Condition: TIMA_<t>_TRIG port samples up to rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b7	HCDO7	Hardware decrement condition 7	Condition: When the TIMA_<t>_CLKA port is at high level, the TIMA_<t>_CLKA port is sampled to the falling edge. 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b6	HCDO6	Hardware decrement condition 6	Condition: When the TIMA_<t>_CLKA port is at a high level, the TIMA_<t>_CLKA port is sampled up to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b5	HCDO5	Hardware decrement condition 5	Condition: When the TIMA_<t>_CLKA port is at low level, the TIMA_<t>_CLKA port is sampled to the falling edge. 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b4	HCDO4	Hardware decrement condition 4	Condition: When the TIMA_<t>_CLKA port is at low level, the TIMA_<t>_CLKA port is sampled up to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b3	HCDO3	Hardware decrement condition 3	Condition: When the TIMA_<t>_CLKB port is at high level, the TIMA_<t>_CLKB port is sampled up to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b2	HCDO2	Hardware decrement condition 2	Condition: When the TIMA_<t>_CLKB port is at low level, the TIMA_<t>_CLKB port is sampled up to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b1	HCDO1	Hardware decrement condition 1	Condition: When the TIMA_<t>_CLKB port is at high level, the TIMA_<t>_CLKB port is sampled up to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W
b0	HCDO0	Hardware decrement condition 0	Condition: When the TIMA_<t>_CLKB port is at low level, the TIMA_<t>_CLKB port is sampled up to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrements are effective when conditions match												R/W

22 General Timer (Timer0)

22.1 Introduction

General 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), which can generate compare match events during counting. The event can trigger interrupt, or can be output as an event to control other modules. This series of products is equipped with 2 units of Timer0.

22.2 Basic Block Diagram

The basic block diagram of Timer0 is shown in Figure 22-1.

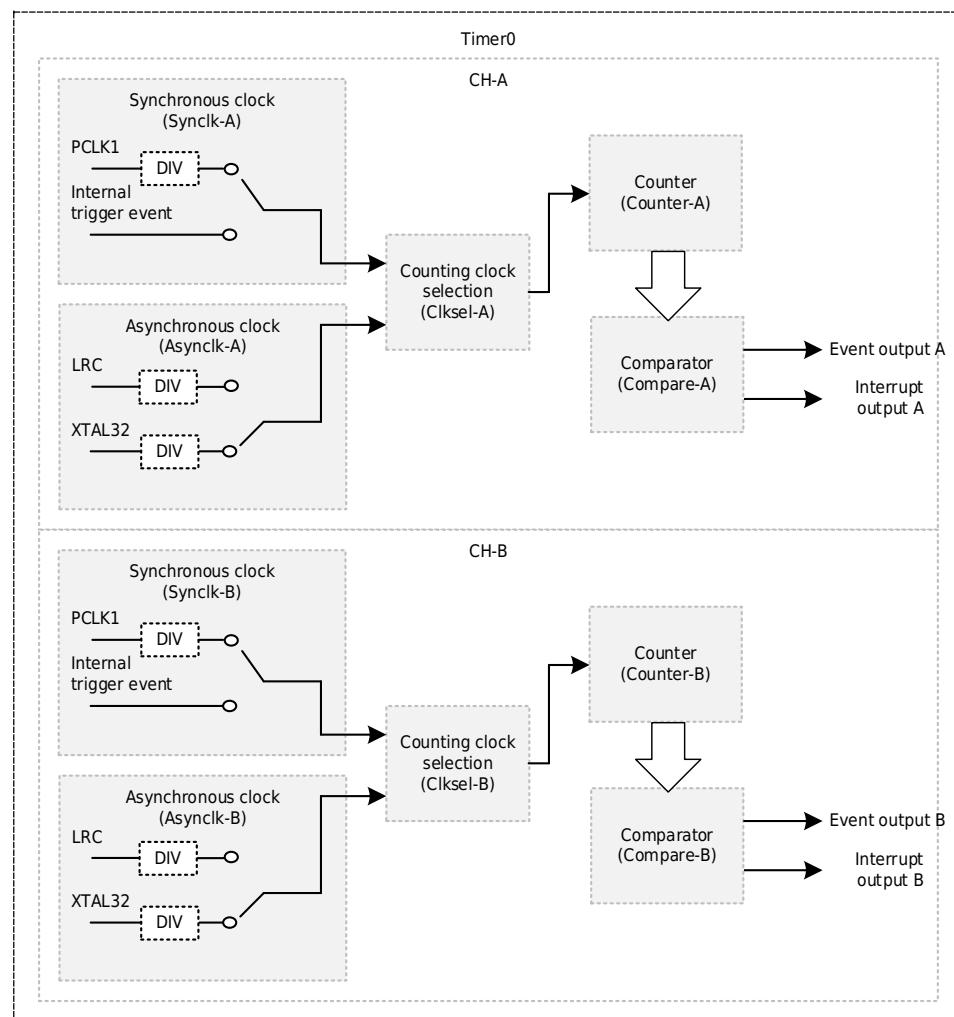


Figure 22-1 Timer0 Basic Block Diagram

22.3 Functional Description

22.3.1 Timer Selection

Timer0 can be counted either synchronously or asynchronously.

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 means that the timer count clock and bus access clock (register read/write operation clock) are non-synchronous timing relationships. The status of a 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.

22.3.1.1 Synchronous Count Timer

In synchronous counting mode (BCONR.SYNSA=0), the clock source can have the following options (BCONR.SYNCLKA setting):

- a) PCLK1 and PCLK1 divided by 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 as the synchronous counting clock (BCONR.SYNCLKA=0 & BCONR.CKDIVA [3:0] set up)
- b) Internal Hardware Trigger Event Input as Sync Count Clock (BCONR.SYNCLKA=1)

22.3.1.2 Asynchronous Count Timer

In the asynchronous counting mode (BCONR.SYNSA=1), the clock source can have the following options (BCONR.ASYNCLK A setting):

- a) LRCtimer input and its 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 frequency division as asynchronous counting clock (BCONR.ASYNCLK A=0 & BCONR.CKDIVA [3:0] setting)
- b) LRCtimer input and its 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 frequency division as asynchronous counting clock (BCONR.ASYNCLK A=1 & BCONR.CKDIVA [3:0]setting)

22.3.2 Basic Counting Action

Each channel of Timer0 can set a reference count value, and a count comparison match event will be generated when the count value is equal to the reference value. As shown in Figure 22-2.

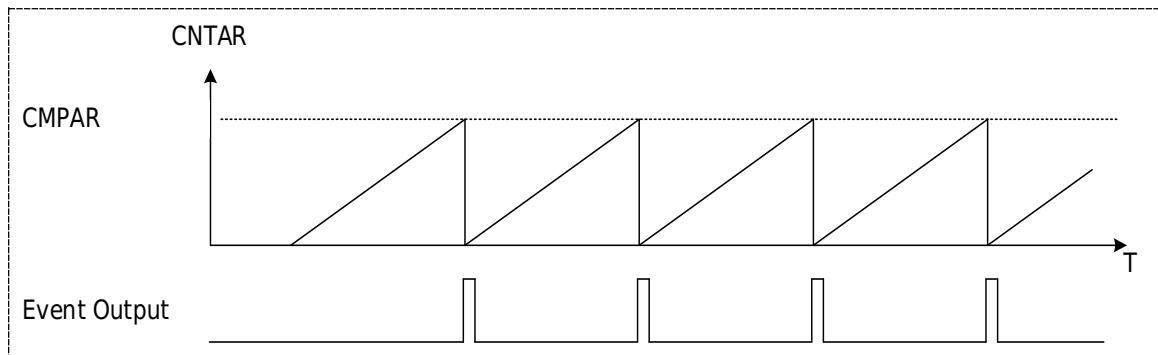


Figure 22-2 Timer0 Counting Timing Diagram

22.3.3 Hardware Trigger Action

Timer0 contains 2 channels has a common internal hardware trigger source that can control the status of the timer (counting, starting, stopping, clearing, etc.) and capture input actions by basic control register (BCONR) related settings

The source selection of the hardware trigger source is realized by entering the corresponding number into the hardware trigger select register (HTSSR). For the specific event correspondence, refer to the [Interrupt Controller (INTC)] chapter. When using the internal hardware trigger function, you need to enable the peripheral circuit trigger function bit (AOS) of the function clock control 0 register (PWC_FCG0) first.

22.4 Interrupt and Event Description

22.4.1 Interrupt Output

A Timer0 contains 2 interrupts, which are count compare match interrupt or input capture interrupt of channel A and channel B respectively.

There are 2 reference value registers (CMPAR, CMPBR), which can be compared with the count value registers (CNTAR, CNTBR) respectively to generate comparison matching valid signals. When the counting comparison matches, the STFLR.CMAF and STFLR.CMBF bits in the status flag register (STFLR) will be 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_Um_GCMn, 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_Um_GCMn, m=1, 2; n=A, B) will be triggered.

Note:

- The compare match interrupt (TMR0_U1_GCMA) of channel A of unit 1 is only available in the asynchronous counting mode (BCONR.SYNSA=1).

22.4.2 Event Output

A Timer0 contains 2 event outputs, which are the count compare match event or capture input event of channel A and channel B respectively.

When a count comparison match or capture input action occurs during the counting process, the corresponding event request (TMR0_Um_GCMn, m=1, 2; n=A, B) output signal will be generated respectively, which can be used to select and trigger other modules.

22.5 Register Description

Table 22-1 shows the register list of the Timer0 module.

BASE ADDR: 0x4002_4000 (U1) , 0x4002_4400 (U2)

Table 22-1 Register List

Register name	Symbol	Offset	Bit width	Reset value
Register count value	TMR0_CNTAR	0x0000	32	0x0000_0000
Register count value	TMR0_CNTBR	0x0004	32	0x0000_0000
Reference value register	TMR0_CMPAR	0x0008	32	0x0000_FFFF
Reference value register	TMR0_CMPBR	0x000C	32	0x0000_FFFF
Basic control register	TMR0_BCONR	0x0010	32	0x0000_0000
Status flag register	TMR0_STFLR	0x0014	32	0x0000_0000

22.5.1 Count Value Register (TMR0_CNTAR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNTA[15:0]															
Bit	Marking		Place name		Function								Read and write		
b31~b16	Reserved		-		Read as "0"								R		
b15~b0	CNTA[15:0]		Count value		Current timer count								R/W		

22.5.2 Reference Value Register (TMR0_CMPAR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CMPA[15:0]															
Bit	Marking		Place name		Function								Read and write		
b31~b16	Reserved		-		Read as "0"								R		
b15~b0	CMPA[15:0]		Reference value		Setting the count reference to produce the Compare Match event								R/W		

22.5.3 Basic Control Register (TMR0 _ BCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
HICP B	HCLE B	HSTP B	HSTA B	-	ASYN CLKB	SYN CLKB	SYN SB		CKDIV B[3:0]	-		INT ENB	CAP MDB	CST B	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
HICP A	HCLE A	HSTP A	HSTA A	-	ASYN CLKA	SYN CLKA	SYN SA		CKDIV A[3:0]	-		INT ENA	CAP MDA	CST A	

Bit	Marking	Place name	Function	Read and write
b31	HICPB	Hardware trigger input capture B	Condition: Internal Hardware Trigger Event Valid 0: Capture input is invalid when a condition matches 1: Capture input is valid when conditions match	R/W
b30	HCLEB	Hardware trigger clear B	Condition: Internal Hardware Trigger Event Valid 0: Timer zeroing is invalid when conditions match 1: Timer zeroing is valid when conditions match	R/W
b29	HSTPB	Hardware trigger stop B	Condition: Internal Hardware Trigger Event Valid 0: Timer stops invalid when condition matches 1: When the condition matches, the timer stops working	R/W
b28	HSTAB	Hardware trigger start B	Condition: Internal Hardware Trigger Event Valid 0: Timer start invalid when condition matches 1: Timer is active when condition matches	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 Note: When the timeout function of the corresponding channel of UART is enabled, the clock source is no longer XTAL32, but the clock tick generated by the UART baud rate generator	R/W
b25	SYNCLKB	Channel B synchronous counting clock source selection	0: PCLK1 1: Internal Hardware Triggering Event	R/W
b24	SYNSB	Channel B counting mode selection	0: Synchronous counting mode 1: Asynchronous counting	R/W
b23~b20	CKDIVB[3:0]	Channel B counting clock frequency division selection	Channel B counting clock frequency division selection: 0000: Timer 0001: Timer/2 0010: Timer/4 0011: Timer/8 0100: Timer/16 0101: Timer/32 0110: Timer/64 0111: Timer/128 1000: Timer/256 1001: Timer/512 1010: Timer/1024 Please do not set other values Note: The timer to be divided can be a variety of timer at asynchronous counting, PCLK1 at synchronous counting.	R/W
b19	Reserved	-	Read as "0", write as "0"	R/W
b18	INTENB	Count match interrupt enable B	0: CMPBR register is equal to the count value (CNTBR), or the interrupt is invalid when a capture input event occurs 1: CMPBR register is equal to the count value (CNTBR), or the interrupt is enabled when a capture input event occurs	R/W
b17	CAPMDB	Function mode selection B	0: Comparison output function 1: Capture input function	R/W
b16	CSTB	Timer start	0: channel B timer off 1: Channel B timer starts Note: This bit automatically changes to 0 when the hardware trigger stop condition is valid	R/W
b15	HICPA	Hardware Trigger Input Capture A	Condition: Internal Hardware Trigger Event Valid 0: Capture input is invalid when a condition matches 1: Capture input is valid when conditions match	R/W
b14	HCLEA	Hardware triggered zeroing A	Condition: Internal Hardware Trigger Event Valid 0: Timer zeroing is invalid when conditions match 1: Timer zeroing is valid when conditions match	R/W
b13	HSTPA	Hardware Trigger Stop A	Condition: Internal Hardware Trigger Event Valid 0: Timer stops invalid when condition matches 1: When the condition matches, the timer stops working	R/W

b12	HSTAA	Hardware Trigger Startup A	Condition: Internal Hardware Trigger Event Valid 0: Timer start invalid when condition matches 1: Timer is active when condition matches	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 Count Timer Selection	0: PCLK1 1: Internal Hardware Triggering Event	R/W
b8	SYNSA	Channel A counting mode selection	0: Synchronous counting mode 1: Asynchronous counting	R/W
b7~b4	CKDIVA[3:0]	Channel A Clock Frequency Division Selection	Channel A count clock frequency division options: 0000: Timer 0001: Timer/2 0010: Timer/4 0011: Timer/8 0100: Timer/16 0101: Timer/32 0110: Timer/64 0111: Timer/128 1000: Timer/256 1001: Timer/512 1010: Timer/1024 Please do not set other values Note: The timer to be divided can be a variety of timer at asynchronous counting, PCLK1 at synchronous counting.	
			R/W	
b3	Reserved	-	Read as "0", write as "0"	R/W
b2	INTENA	Count Match interrupt enable A	0: CMPAR register is equal to the count value (CNTBR), or the interrupt is invalid when a capture input event occurs 1: CMPAR register is equal to the count value (CNTBR), or the interrupt is enabled when a capture input event occurs	R/W
b1	CAPMDA	Functional mode selection A	0: Comparison output function 1: Capture input function	R/W
b0	CSTA	Timer start	0: Channel A timer is off 1: Channel A timer starts Note: This bit automatically changes to 0 when the hardware trigger 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) during asynchronous counting are all provided by the USART module when the TIMEOUT function of the USART module is valid. For details, please refer to Refer to USART chapter introduction.

22.5.4 Status flag register (TMR0_STFLR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															CMBF
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															CMAF
Bit	Marking	Place name	Function	Read and write											
b31~b17	Reserved	-	Read as "0", write as "0"	R											
b16	CMBF	Count match B	0: The value of the CMPBR register is not equal to the count value and no capture input action occurs 1: The value of the CMPBR register is equal to the count value or a capture input action occurs	R/W											
b15~b1	Reserved	-	Read as "0", write as "0"	R											
b0	CMAF	Count Match A	0: The value of the CMPAR register is not equal to the count value and no capture input action occurs 1: The value of the CMPAR register is equal to the count value or a capture input action occurs	R/W											

22.6 Precautions for Use

- 1) In the asynchronous counting operation, first set the BCONR.ASYNCLKA bit to select the asynchronous clock source, then set the BCONR.SYNSA bit to select the asynchronous counting mode, and then start Timer0.
- 2) In the case of selecting asynchronous counting, count value (CNTAR), reference value (CMPAR), start bit (BCONR.CSTA), status bit (STFLR.CMAF). When performing a write action, Timer0 writes the modified value into the corresponding register after 3 asynchronous count clocks after receiving the write action.

23 Real Time Clock (RTC)

23.1 Introduction

A real-time clock (RTC) is a counter that stores time information in BCD format. Record the specific calendar time from 00 to 99. Support 12/24 hour time system, automatically calculate the number of days 28, 29 (leap year), 30 and 31 according to the month and year. Table 23-1 Shown are its basic characteristics.

Table 23-1 Basic Specifications of RTC

Count clock source	External low-speed oscillator (32.768KHz) RTC internal low-speed oscillator (32.768KHz)
Basic functions	• BCD code indicates seconds, minutes, hours, days, weeks, months, years
	• Software start or stop
	• 12/24 hour format optional, leap year automatic recognition
	• Programmable Alarm Clock
	• Distributed/uniform compensation 1Hz clock output
	• Clock error compensation function
Interrupt	Cycle interruption
	Alarm interrupt

23.2 Basic block diagram

The basic block diagram of the RTC is shown in Figure 23-1.

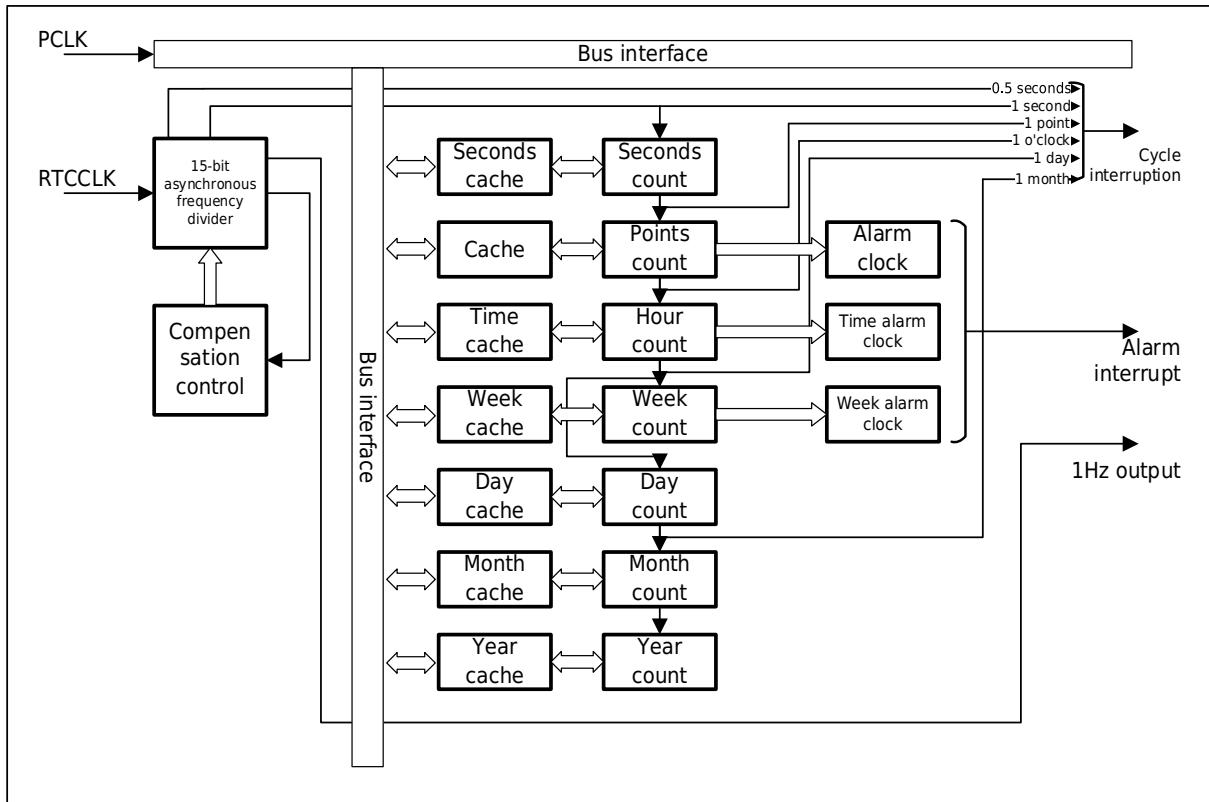


Figure 23-1 Basic block diagram of RTC

23.3 Functional Description

23.3.1 Power-on Setting

The RTC register can be reset by power-on reset or setting RTC_CR0.RESET. After setting the control register, calendar initial value, alarm clock setting, etc., start the RTC. After the RTC is started, other external reset requests cannot reset the RTC, and the RTC will always be in the working state. You can set the START bit of the control register to "0" to stop the RTC from working. When setting the RTC, the clock source must be stable.

23.3.2 RTC Count Start Setting

- 1) After power-on, all registers except RTC_CR0 are reset; you can also set RTC_CR0.RESET=0, and after confirming that the RESET bit is "0", set RTC_CR0.RESET=1 to reset all registers;
- 2) Set RTC_CR1.START=0, stop counting;
- 3) Set the system clock register, turn on the auxiliary oscillator, and then set RTC_CR3 to select the RTC counting clock source;
- 4) Set RTC_CR1, set time, period, 1Hz clock output;
- 5) Set the calendar counting registers for seconds, minutes, hours, days, weeks, months, and years;
- 6) When clock error compensation is required, set the counting clock error compensation registers RTC_ERRCRL, RTC_ERRCRH;
- 7) Clear the flag register bit in the register RTC_CR1 and enable the interrupt;
- 8) Set RTC_CR1.START=1, counting starts.

23.3.3 System Low Power Mode Switching

When the system switches to the low power consumption mode immediately after the RTC counting starts, please perform any of the following confirmations before switching the mode.

- 1) After setting RTC_CR1.START=1, switch modes after more than 2 RTC count clocks.
- 2) After setting RTC_CR1.START=1, 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 modes.

23.3.4 Read Count Register

- 1) After RTC_CR1.START=1 is set, after more than 2 RTC count clocks, set RTC_CR2.RWREQ=1 to make a calendar register read request;
- 2) Query until RTC_CR2.RWEN=1;
- 3) Read all or part of the second, minute, hour, day, week, month, year count register value;
- 4) Set RTC_CR2.RWREQ=0;
- 5) Query until RTC_CR2.RWEN=0.

23.3.5 Write Count Register

- 1) After setting RTC_CR1.START=1, after more than 2 RTC counting 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, day, week, 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.

23.3.6 Alarm Clock Setting

- 1) Set RTC_CR2.ALME=0, the alarm clock is disabled;
- 2) Set RTC_CR2.ALMIE=1, the alarm clock interruption permission;
- 3) Minute alarm clock RTC_ALMMIN, hour alarm clock RTC_ALMHOUR, week alarm clock RTC_ALM WEEK setting;
- 4) Set RTC_CR2.ALME=1, alarm clock permission;
- 5) Wait for an interrupt to occur;
- 6) When RTC_CR2.ALMF=1, enter the alarm interrupt processing.

23.3.7 Clock Error Compensation

Due to the error of the external secondary oscillator working under various temperature conditions, it is necessary to compensate the error when it is necessary to obtain high-precision counting results. Refer to Compensation Method 23.5.15 When Clock Error Compensation, Set The Counting Clock Error Compensation Registers RTC_ERRCRL, RTC_ERRCRH;.

23.3.8 1Hz Output

The RTC can output a 1Hz clock and provide three precision output methods. The first is the ordinary precision 1Hz output without clock error compensation; the second is the distributed compensation 1Hz output with average compensation within every 32 seconds and the third is compensation per second. The uniform compensation 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. Of which,

The 1Hz output of ordinary precision is set as follows:

- 1) Set RTC_CR0.RESET=0, after confirming that the RESET bit is "0", set RTC_CR0.RESET=1, reset the calendar count register;
- 2) Set RTC_CR1.START=0, counting stops;
- 3) 1Hz output pin setting;
- 4) RTC_CR1.ONEHZOE=1, clock output permission;
- 5) Set RTC_CR1.START=1, start counting;

- 6) Wait for more than 2 count cycles;
- 7) 1Hz output starts.

The distributed compensation 1Hz output setting is as follows:

- 1) Set RTC_CR0.RESET=0, after confirming that the RESET bit is "0", set RTC_CR0.RESET=1, reset the calendar count register;
- 2) Set RTC_CR1.START=0, counting stops;
- 3) 1Hz output pin setting;
- 4) RTC_CR1.ONEHZOE=1, clock output permission;
- 5) 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, start counting;
- 8) Wait for more than 2 count cycles;
- 9) 1Hz output starts.

Uniform compensation 1Hz output setting is as follows:

- 1) Set RTC_CR0.RESET=0, after confirming that the RESET bit is "0", set RTC_CR0.RESET=1, reset the calendar count register;
- 2) Set RTC_CR1.START=0, counting stops;
- 3) RTC output pin setting;
- 4) RTC_CR1.ONEHZOE=1, clock output permission;
- 5) RTC_CR1.ONEHZSEL=1, select 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, accuracy compensation is valid;
- 8) Set RTC_CR1.START=1, start counting;
- 9) Wait for more than 2 count cycles;
- 10) 1Hz output starts.

23.4 Interrupt Description

RTC supports 2 interrupt types. Timing alarm interrupt and periodic interrupt.

23.4.1 Alarm Interrupt

Alarm clock interrupt RTC_ALM, when ALMIE=1 of the control register 2 (RTC_CR2) and ALME=1 of the control register 2 (RTC_CR2), if the current calendar time and the minute alarm clock register (RTC_ALMMIN), the hour alarm clock register (RTC_ALMHOUR), and the weekly alarm clock register (RTC_ALMWEEK) equal, trigger the alarm interrupt. The alarm clock configures an independent flag register bit RTC_CR2.ALMF, write "0" to the RTC_CR1.ALMFCLR bit to clear the alarm clock flag. The alarm clock interrupt can only be configured to the corresponding NVIC vector through INTC.INT_SELx (x=0~31, 44~49).

23.4.2 Periodic Interrupt

Periodic interrupt RTC_PRD, when PRDIE=1 of the control register 2 (RTC_CR2), after the selected period occurs, a regular period wake-up interrupt is triggered. The fixed-period interrupt is not configured with an independent flag bit. You can select the RTC_PRD interrupt to the NVIC vector through INTC.INT_SELx (x=0~31, 44~49), and query the corresponding flag bit of the NVIC.

23.5 Register Description

Table 23-2 shows the register list of the RTC module.

Register base address: 0x4004_C000

Table 23-2 Register List

Register name	Symbol	Offset	Bit width	Reset value
Control register 0	RTC_CR0	0x0000	8	不定
Control register 1	RTC_CR1	0x0004	8	0x00
Control register 2	RTC_CR2	0x0008	8	0x00
Control register 3	RTC_CR3	0x000C	8	0x00
Second Count Register	RTC_SEC	0x0010	8	0x00
Minute counter register	RTC_MIN	0x0014	8	0x00
Time counter register	RTC_HOUR	0x0018	8	0x12
Week Count Register	RTC_WEEK	0x001C	8	0x00
Day Count Register	RTC_DAY	0x0020	8	0x00
Month Count Register	RTC_MON	0x0024	8	0x00
Year Count Register	RTC_YEAR	0x0028	8	0x00
Minute alarm register	RTC_ALMMIN	0x002C	8	0x00
Time alarm register	RTC_ALMHOUR	0x0030	8	0x12
Weekly Alarm Register	RTC_ALM WEEK	0x0034	8	0x00
Clock Error Compensation Register	RTC_ERRCRH	0x0038	8	0x00
Clock Error Compensation Register	RTC_ERRCRL	0x003C	8	0x20

23.5.1 Control Register 0 (RTC_CR0)

Reset value: 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																
Read and write																
Bit	Marking	Place name	Function													
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 status or RTC software reset ends 1: RTC is in reset state Note: Please confirm that this bit is "0" before writing "1", otherwise it will fail to initialize.													R/W

23.5.2 Control Register 1 (RTC_CR1)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16																																
Reserved																																															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																
Reserved								START	ONEHZSEL	ONEHZOE	ALMFCLR	AMPM	PRDS[2]	PRDS[1]	PRDS[0]																																
<hr/>																																															
Bit	Marking	Place name	Function													Read and write																															
b31~b8	Reserved	-	Read as "0", write as "0"													R/W																															
b7	START	RTC counting starts	0: RTC count stop 1: RTC counting work													R/W																															
b6	ONEHZSEL	1Hz output selection	0: Distributed compensation 1Hz output 1: Uniform compensation 1Hz output <small>Note: When RTC_ERRCRH.COMPEN=1, this bit setting is valid.</small>													R/W																															
b5	ONEHZOE	1Hz output enable	0: 1Hz output disabled 1: 1Hz output permission													R/W																															
b4	ALMFCLR	ALMF flag clear	0: Clear the RTC_CR2.ALMF flag register 1: Invalid													R/W																															
b3	AMPM	Time selection	0:12 hour clock 1: 24-hour time format													R/W																															
<hr/>																																															
Cycle selection settings:																																															
<table border="1"> <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 cycle</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 cycle</td></tr> <tr> <td>1</td><td>0</td><td>1</td><td>Every 1 day cycle (every day at 00:00:00:00)</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 cycle	0	1	1	Every 1 minute cycle	1	0	0	Every 1 hour cycle	1	0	1	Every 1 day cycle (every day at 00:00:00:00)	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 cycle																																												
0	1	1	Every 1 minute cycle																																												
1	0	0	Every 1 hour cycle																																												
1	0	1	Every 1 day cycle (every day at 00:00:00:00)																																												
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, please disable the cycle interrupt permission to prevent malfunction. And the relevant flag should be cleared after writing.</small>																																															

23.5.3 Control Register 2 (RTC_CR2)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved																
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved								ALME	ALMIE	PRDIE	-	ALMF	-	RWEN	RWREQ	
Bit	Marking	Place name	Function													Read and write
b31~b8	Reserved	-	Read as "0", write as "0"													R/W
b7	ALME	Alarm enabled	0: Alarm clock function disabled 1: Alarm clock function permission													R/W
b6	ALMIE	Alarm clock interrupt enable	0: Alarm interrupt disabled 1: Alarm interrupt permission													R/W
b5	PRDIE	Periodic 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: Alarm does not match 1: Alarm match													R
b2	Reserved	-	Uncertain when reading, write "0" when writing													R/W
b1	RWEN	Read/write enable	0: Read/write disabled 1: Read/write enabled Note: Calendar register read and write permission flag. Please confirm whether this bit is "1" before reading/writing. The calendar registers include seconds, minutes, hours, weeks, days, months, and year counting 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" to request reading and writing. Since the counter is counting continuously, please complete the reading/writing operation within 1 second and clear this bit to "0".													R/W

23.5.4 Control Register 3 (RTC_CR3)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								RCKSEL	-	-	LCREN	-	-	-	-
<hr/>															
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 sub oscillator 1: Select LOCO	R/W											
b6	Reserved	-	Read as "0", write as "0"	R/W											
b5	Reserved	-	Read as "0", write as "0"	R/W											
b4	LCREN	Low speed oscillator enable	0: Low-speed oscillator stops 1: Low speed oscillator works Note: The operation and stop of the low-speed oscillation are determined by the clock control circuit and any setting value of LRCEN. When the low-speed oscillator is used as the RTC clock source, please set the LRCEN bit to enable it.	R/W											
b3~b1	Reserved	-	Read as "0", write as "0"	R/W											
b0	Reserved	-	Read as "0", write as "0"	R/W											

23.5.5 Second Count Register (RTC_SEC)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	SECD[2:0]			SECU[3:0]			
<hr/>															
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 digit count value	R/W											
b3~b0	SECU[3:0]	Seconds	Second count value	R/W											

Indicates 0-59 seconds, using decimal counting. Please write the BCD code of decimal 0-59. When writing an incorrect value, the written value will be ignored.

23.5.6 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]		Tens		Minute and tens count value								R/W		
b3~b0	MINU[3:0]		Minutes		Minute count value								R/W		

Indicates 0-59 points, using decimal counting. Please write the BCD code of decimal 0-59. When writing an incorrect value, the written value will be ignored.

23.5.7 Hour Count Register (RTC_HOUR)

Reset value: 0x12

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

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R/W
b5~b4	HOURD[1:0]	Time ten	Hour tens count value	R/W
b3~b0	HOURU[3:0]	When ones	Hour count value	R/W

In 24-hour format, it means 0~23 hours. In 12-hour format, 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. Writing a value out of range will be ignored.

Please refer to the following table for the specific time:

24 hour clock	AMPM=1	12 hour clock	AMPM=0
time	Register representation	time	Register representation
SYNCMD[2]	00H	AM 12:00	12H
01 o'clock	01H	AM 1:00	01H
02 o'clock	02H	AM 2:00	02H
03 o'clock	03H	AM 3:00	03H
04 o'clock	04H	AM 4:00	04H
05 o'clock	05H	AM 5:00	05H
06 o'clock	06H	AM 6:00	06H
07 o'clock	07H	AM 7:00	07H
08 o'clock	08H	AM 8:00	08H
09 o'clock	09H	AM 9:00	09H
10 o'clock	10H	AM 10:00	10H
11 o'clock	11H	AM 11:00	11H
12 o'clock	12H	PM 12:00	32H
13 o'clock	13H	PM 1:00	21H
14 o'clock	14H	PM 2:00	22H
15 o'clock	15H	PM 3:00	23H
16 o'clock	16H	PM 4:00	24H
17 o'clock	17H	PM 5:00	25H
18 o'clock	18H	PM 6:00	26H
19 o'clock	19H	PM 7:00	27H
20 o'clock	20H	PM 8:00	28H
21 o'clock	21H	PM 9:00	29H
22 o'clock	22H	PM 10:00	30H
23 o'clock	23H	PM 11:00	31H

23.5.8 Day Count Register (RTC_DAY)

Reset value: 0x00

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

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R/W
b5~b4	DAYD[1:0]	day ten	Day tens digit count value	R/W
b3~b0	DAYU[3:0]	day ones	Day count value	R/W

Decimal means 1~31 days, automatically calculate leap year and month. The specific expression is as follows:

month	day count display
February (normal year)	01~28
February (leap year)	01~29
4,6,9,11月	01~30
1,3,5,7,8,10,12月	01~31

23.5.9 Week Count Register (RTC_WEEK)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	-	-	-	-	-	-	WEEK[2:0]
<hr/>															
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	R/W											

Decimal 0~6 means Sunday~Saturday. Please write the BCD code of decimal 0~ 6. writing an incorrect value, the will be ignored. The corresponding relationship between the week count value is as follows:

week	Week count
sunday	00H
on Monday	01H
Tuesday	02H
Wednesday	03H
Thursday	04H
friday	05H
Saturday	06H

23.5.10 Month Count Register (RTC_MON)

Reset value: 0x00

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

Bit	Marking	Place name	Function	Read and write
b31~b5	Reserved	-	Read as "0", write as "0"	R/W
b4~b0	MON[4:0]	January	Month count	R/W

Decimal 1~12 means Sunday~ 12 January. Please write in the correct decimal 1~12 BCD code, other values will be ignored.

23.5.11 Year Count Register (RTC_YEAR)

Reset value: 0x00

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

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 tens digit count value	R/W
b3~b0	YEARU[3:0]	year unit	Year count value	R/W

Decimal 0~99 means 0~99 years. Count according to the monthly round. Automatically calculate leap years such as: 00, 04, 08, ..., 92, 96, etc. Please write correct decimal year count value, wrong value will be ignored.

23.5.12 Minute Alarm Register (RTC_ALMMIN)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	ALMMIND[2:0]			ALMMINU[3:0]			
<hr/>															
Bit	Marking		Place name		Function										Read and write
b31~b7	Reserved		-		Read as "0", write as "0"										R/W
b6~b4	ALMMIND[2:0]		Alarm clock Alarm tens		Alarm clock Alarm tens match Value										R/W
b3~b0	ALMMINU[3:0]		minute alarm clock		Alarm clock Alarm bit match Value										R/W

Please set the BCD code of decimal 0~59. Write other values, no alarm match will occur.

23.5.13 Hour Alarm register (RTC_ALMHOUR)

Reset value: 0x12

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved																
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved								-	-	ALMHOURD[1:0]			ALMHOURU[3:0]			
<hr/>																
Bit	Marking		Place name		Function										Read and write	
b31~b6	Reserved		-		Read as "0", write as "0"										R/W	
b5~b4	ALMHOURD[1:0]		alarm clock tens		Alarm When Alarm tens match Value										R/W	
b3~b0	ALMHOURU[3:0]		alarm clock unit		Alarm When Alarm bit match Value										R/W	

Please set the correct alarm clock matching value according to the time system, otherwise the alarm clock matching will not happen.

23.5.14 Week Alarm Register (RTC_ALMWEEK)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	ALMWEEK[6:0]						
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b7	Reserved	-	Read as "0", write as "0"	R/W											
b6~b0	ALMWEEK[6:0]	Week alarm clock	Week alarm match value. b0~b6 correspond to Sunday~Saturday respectively, and when they are 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 Sunday and Friday alarm settings are valid.	R/W											

Please set the correct alarm clock matching value according to the time system, otherwise the alarm clock matching will not happen.

23.5.15 When Clock Error Compensation, Set The Counting Clock Error Compensation Registers RTC_ERRCRL, RTC_ERRCRH;

Reset value: 0x0000_0020

RTC_ERRCRH

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								COMPEN	-	-	-	-	-	-	COMP[8]

RTC_ERRCRL

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								COMP[7:0]							

RTC_ERRCRH

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 invalid 1: Clock Error Compensation valid	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	<p>By setting the compensation value, an accuracy compensation of +/-0.96ppm can be performed per second. The compensation value is 9-bit 2's complement code with a decimal point, and the last 5 bits are the decimal part. Compensable range -275.5ppm~+212.9ppm. The minimum resolution is 0.96ppm. Please refer to the following table for specific compensation accuracy:</p> <table border="1"> <thead> <tr> <th colspan="10">Compensation value setting</th> <th>Compensation</th> </tr> <tr> <th>COPDEN</th> <th colspan="9">COMP[8:0]</th> <th></th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>-275.5ppm</td> </tr> <tr> <td></td> <td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>-274.6ppm</td> </tr> <tr> <td></td> <td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>-30.5ppm</td> </tr> <tr> <td></td> <td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>-0.96ppm</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0ppm</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>+0.96ppm</td> </tr> <tr> <td></td> <td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>+30.5ppm</td> </tr> <tr> <td></td> <td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td> </tr> <tr> <td></td> <td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>+212.0ppm</td> </tr> <tr> <td></td> <td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>+212.9ppm</td> </tr> <tr> <td></td> <td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>Invalid</td> </tr> </tbody> </table>	Compensation value setting										Compensation	COPDEN	COMP[8:0]										1	1	0	0	0	0	0	0	0	0	-275.5ppm		1	0	0	0	0	0	0	0	1	-274.6ppm		~	~	~	~	~	~	~	~	~	~		0	0	0	0	0	0	0	0	0	-30.5ppm		~	~	~	~	~	~	~	~	~	~		0	0	0	0	1	1	1	1	1	-0.96ppm		0	0	0	1	0	0	0	0	0	0ppm		0	0	0	1	0	0	0	0	1	+0.96ppm		~	~	~	~	~	~	~	~	~	~		0	0	1	0	0	0	0	0	0	+30.5ppm		~	~	~	~	~	~	~	~	~	~		0	1	1	1	1	1	1	1	0	+212.0ppm		0	1	1	1	1	1	1	1	1	+212.9ppm		0	X	X	X	X	X	X	X	X	Invalid	R/W
Compensation value setting										Compensation																																																																																																																																																																										
COPDEN	COMP[8:0]																																																																																																																																																																																			
1	1	0	0	0	0	0	0	0	0	-275.5ppm																																																																																																																																																																										
	1	0	0	0	0	0	0	0	1	-274.6ppm																																																																																																																																																																										
	~	~	~	~	~	~	~	~	~	~																																																																																																																																																																										
	0	0	0	0	0	0	0	0	0	-30.5ppm																																																																																																																																																																										
	~	~	~	~	~	~	~	~	~	~																																																																																																																																																																										
	0	0	0	0	1	1	1	1	1	-0.96ppm																																																																																																																																																																										
	0	0	0	1	0	0	0	0	0	0ppm																																																																																																																																																																										
	0	0	0	1	0	0	0	0	1	+0.96ppm																																																																																																																																																																										
	~	~	~	~	~	~	~	~	~	~																																																																																																																																																																										
	0	0	1	0	0	0	0	0	0	+30.5ppm																																																																																																																																																																										
	~	~	~	~	~	~	~	~	~	~																																																																																																																																																																										
	0	1	1	1	1	1	1	1	0	+212.0ppm																																																																																																																																																																										
	0	1	1	1	1	1	1	1	1	+212.9ppm																																																																																																																																																																										
	0	X	X	X	X	X	X	X	X	Invalid																																																																																																																																																																										

Compensation calculation instructions:

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 that the actual measured value is 0.9999888Hz, then:

$$\text{Actual oscillation frequency} = 32768 \times 0.9999888 \approx 32767.63$$

Compensation target value = (actual oscillation frequency - target frequency)/target frequency $\times 10^6$

$$\begin{aligned}
 &= (32767.96 - 32768) / 32768 \times 10^6 \\
 &= -11.29\text{ppm}
 \end{aligned}$$

Setpoint calculation:

$$\text{COMP[8:0]} = \left(\frac{\text{Compensation target value[ppm]} \times 2^{15}}{10^6} \right) + 0001.00000B$$

Take the complement of 2

If the compensation target value is +20.3ppm, calculate the corresponding register value as follows:

$$\text{COMP[8:0]} = (20.3 \times 215/106) 2\text{'s complement} + 0001.00000B$$

$$\begin{aligned}
 &= (0.6651904) 2\text{'s complement} + 0001.00000B \\
 &= 0000.10101B + 0001.00000B \\
 &= 0001.10101B
 \end{aligned}$$

If the compensation target value is +20.3ppm, calculate the corresponding register value as follows:

$$\begin{aligned}\text{COMP[8:0]} &= (-20.3 \times 215/106) \text{ 2's complement} + 0001.00000\text{B} \\ &= (-0.6651904) \text{ 2's complement} + 0001.00000\text{B} \\ &= 1111.01011\text{B} + 0001.00000\text{B} \\ &= 0000.01011\text{B}\end{aligned}$$

24 Watchdog Counter (WDT/ SWDT)

24.1 Introduction

There are two watchdog counters, one is the dedicated watchdog counter (SWDT) whose counting clock source is dedicated internal RC (SWDTLRC: 10KHz), and the other is the general-purpose watchdog counter (WDT) whose counting clock source is PCLK3 . The dedicated watchdog and general watchdog are 16-bit down counters used to monitor software failures due to external disturbances or unforeseen logic conditions that deviate from the normal operation of the application program.

Both watchdogs support window functions. The window interval can be preset before the count starts, and when the count value is within the window interval, the counter can be refreshed and the count restarted. Basic characteristics such Table 24-1.

Table 24-1 Basic characteristics of watchdog counter

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(max) 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 (SWDT does not support software startup)
Stop condition	1) Reset 2) Underflow, refresh error occurs Restart: Automatically starts after resetting or terminating the request output in hardware startup mode In software startup mode, set refresh register again
Interrupt/Reset Conditions	1) Count underflow 2) Refresh error

24.2 Functional description

24.2.1 Start the watchdog

Watchdog counter can be started in two ways: Hardware and software. (SWDT only supports hardware boot)

The hardware startup mode refers to reading the watchdog counter setting information (ICG0register) from the main flash memory area during startup, and the counter starts counting automatically. Software startup means that after setting control register, write refresh register completes the refresh action, counter starts counting.

24.2.2 Hardware startup mode

When the bit 16 (WDTAUTS) and bit 0 (SWDTAUTS) of the ICG0 register are 0, it is the hardware startup mode. The WDT_CR register setting information is invalid when selecting hardware startup mode.

In hardware startup mode, the WDT/ SWDT related settings (counting clock, window setting, counting period, etc.) in ICG0register are loaded into the WDT/ SWDT module during reset. After reset, the counter starts counting automatically according to the setting. Figure 24-1 Hardware Startup Example is an example of how the hardware is started.

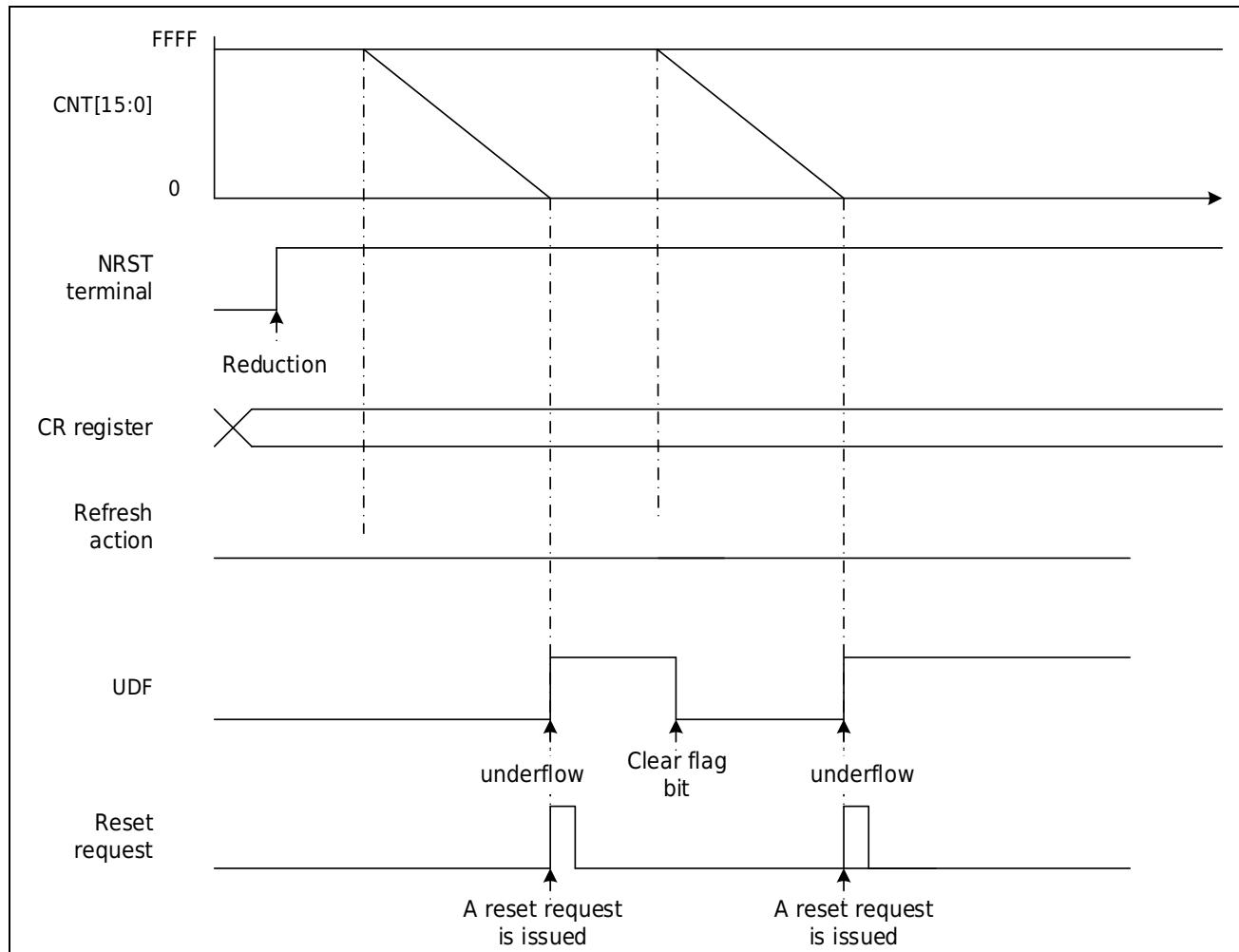


Figure 24-1 Hardware Startup Example

24.2.3 Software Startup Mode

When the bit 16 (WDTAUTS) of ICG0register is 1, the WDT is started by setting the register refresh mode. After the reset, set the counting clock, window setting, counting period and so on in the WDT_CR register, and then execute the refresh action, counter starts counting. WDT_CR can only be set once, and the write value is invalid again. Figure 24-2 Hardware Startup Example is an example of how the software is started.

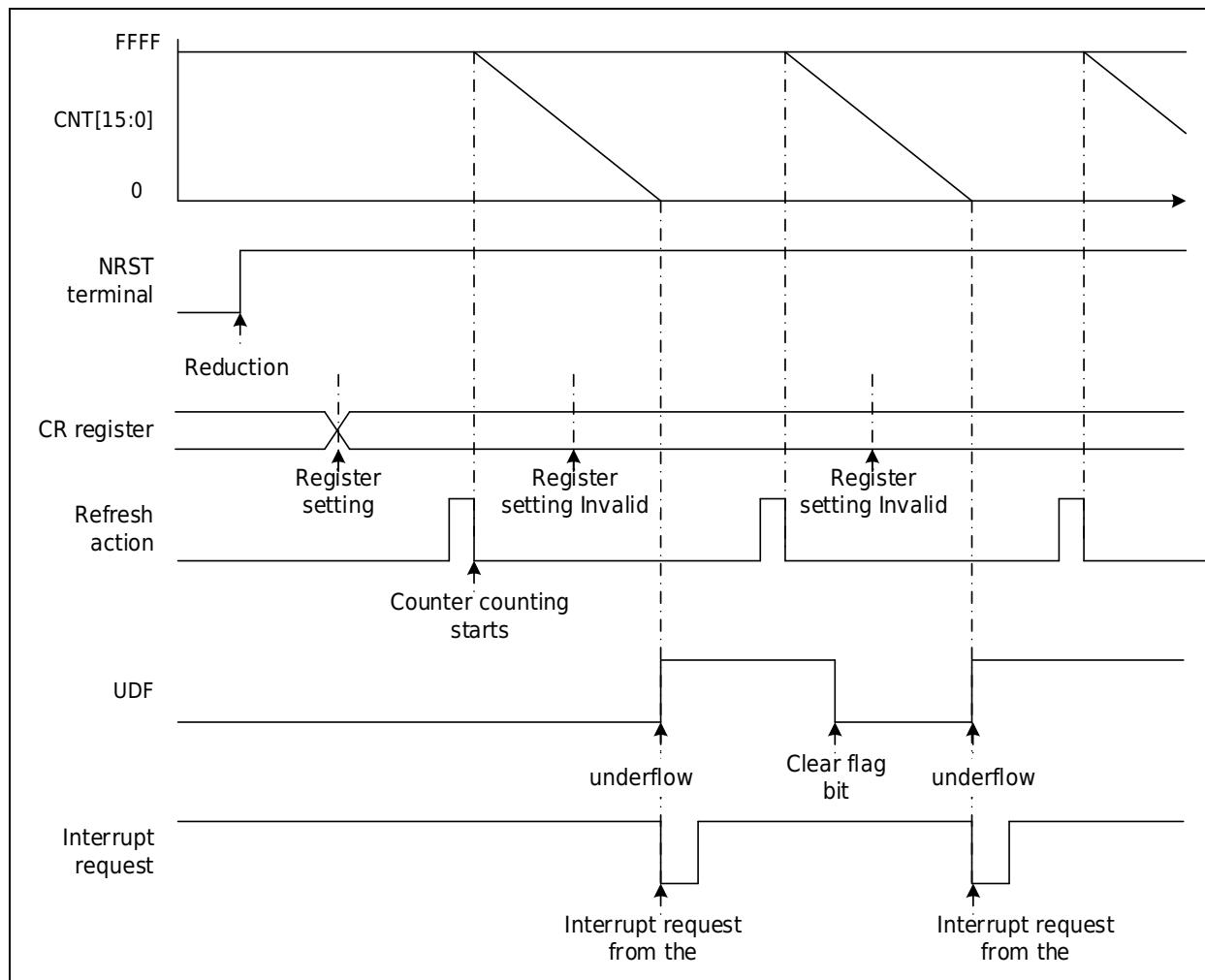


Figure 24-2 Software Startup Example

24.2.4 Refresh Action

(S)WDT_RR register writes 0x0123 first and 0x3210 then completes a refresh action. WDT/SWDT's counter starts counting (software startup) or starts counting again.

(S)WDT_RR register is between write 0x0123 and write 0x3210. If you access or read (S)WDT_RR register to another register, the normal refresh action is not affected.

As Figure 24-3 shown action.

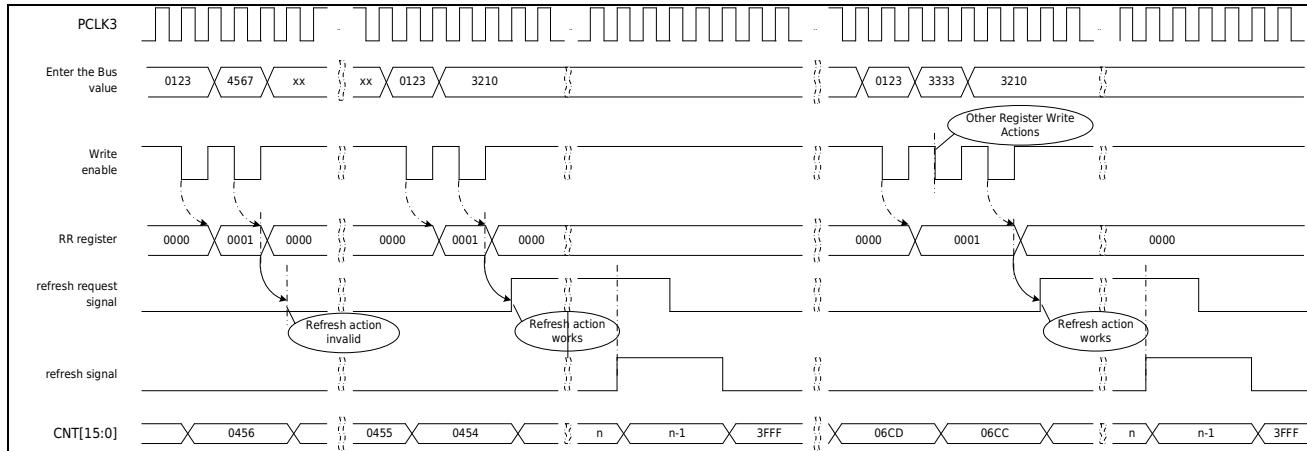


Figure 24-3 Examples of refresh action timing (action confirmation, refresh request signal falling edge, etc.)

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.

24.2.5 Flag Bit

Refresh error flag bits and count underflow flag bits are maintained in both reset and interrupt requests. When the reset is released or interruptd, the reset or interruptd cause can be confirmed by querying the flag bit.

Flag bit zeroing: Read "1" before writing "0".

The hardware boot mode watchdog count does not stop when the refresh error or count underflow flag bit is set. For SWDT, when writing "0" to the flag bit, it takes up to 3 SWDTLRC and 2 PCLK3 times before the register bit can be cleared. For WDT, when writing "0" to the flag bit, it takes at most 5 PCLK3 times 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" and clear the flag bit. This period of time is: 1 counting cycle + 2 SWDTLRC (SWDT module); 1 counting cycle + 2 PCLK3 (WDT module).

24.2.6 Interrupt Reduction

When SWDT overflows or refreshes an error in the counter count, you can choose to generate a interrupt request or a reset request. By setting the SWDTITS bit of ICG0, it is determined whether to generate an interrupt request or a reset request.

When WDT overflows or refreshes an error in the counter count, you can choose to generate a interrupt request or a reset request. When the hardware is started, the WDTITS bit of ICG0 is set, and the ITS bit of the WDT_CR register is set when the software is started, to determine whether to generate an interrupt request or a reset request.

WDT/ There are two conditions for SWDT interrupt reset. One is that counter counts produce underflow; One is to perform a refresh action outside the refresh allowed interval, resulting in a refresh error.

24.2.7 Count Underflow

For Figure 24-4 example, underflow occurs when the decrement counts to zero as shown in Examples of counter underflow.

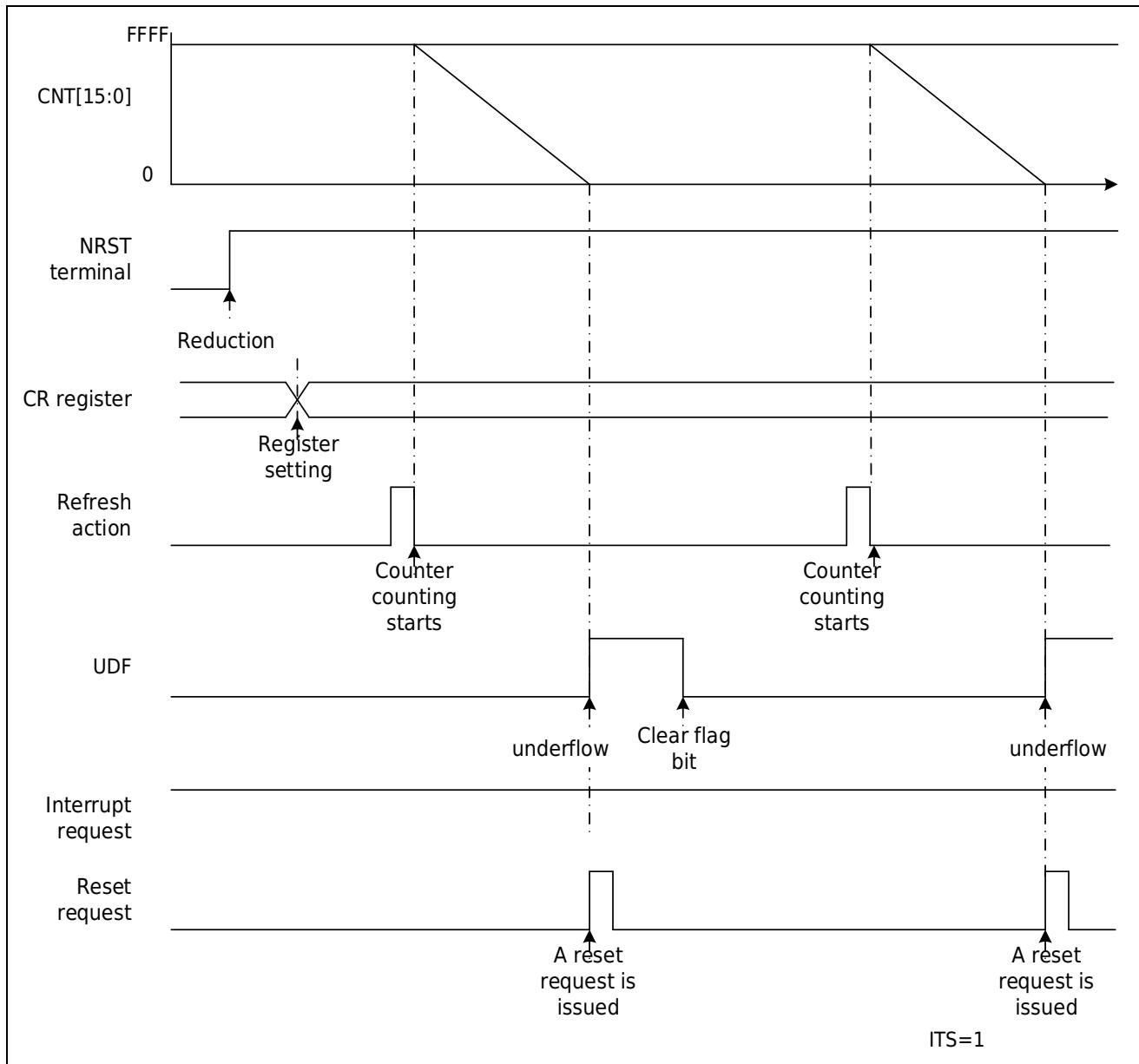


Figure 24-4 Examples Of Counter Underflow

24.2.8 Refresh Error

After the window interval is set, 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 24-5 Refresh Action

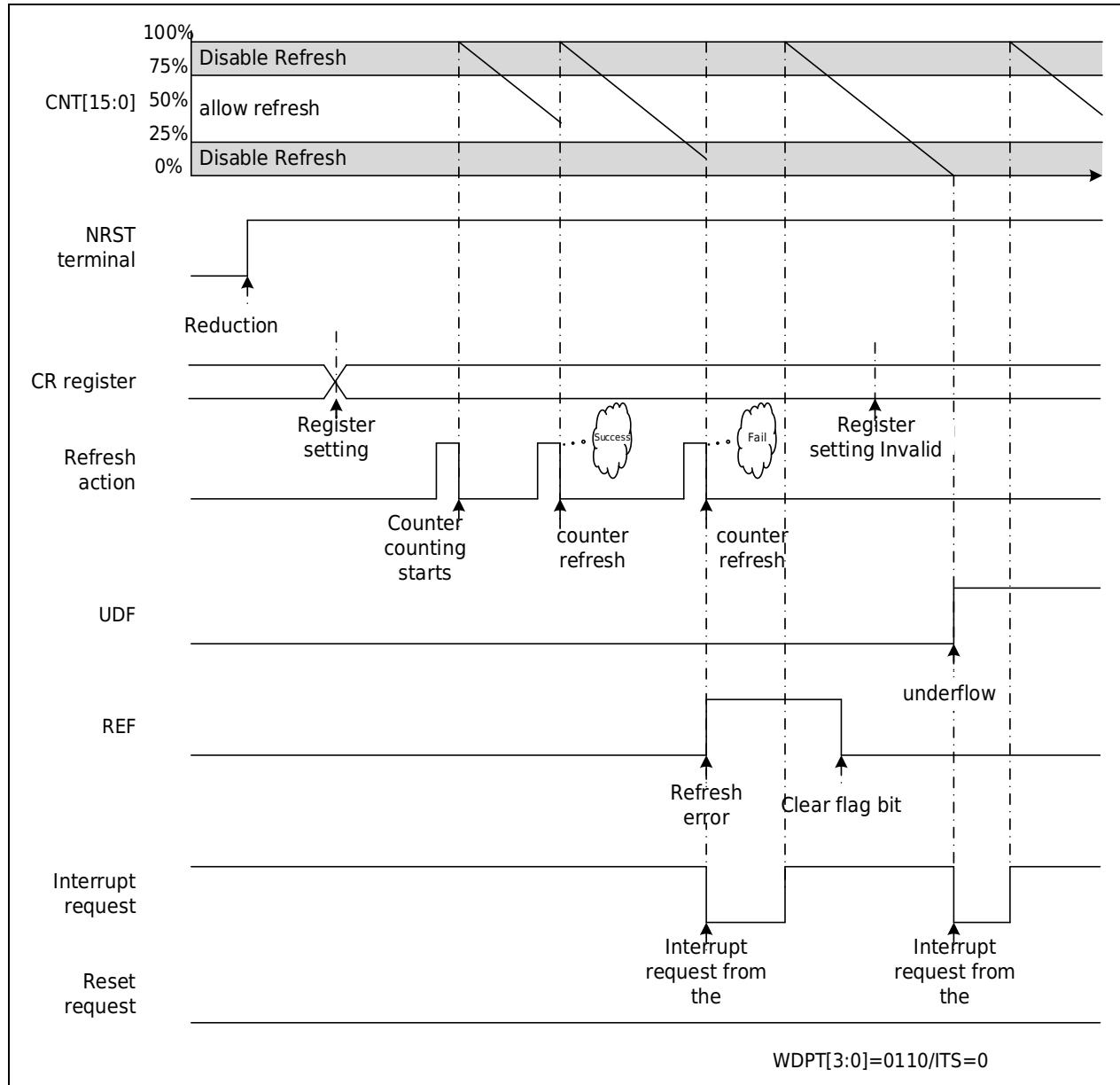


Figure 24-5 Examples of Counter Refresh

24.3 Register Description

Table 24-2 shows the register list of the SWDT and WDT module.

WDT_BASE_ADDR: 0x4004_9000

SWDT_BASE_ADDR: 0x4004_9400

Table 24-2 Register List

Register name	Symbol	Offset address	Bit width	Reset value
SWDT status register	SWDT_SR	0x04	32	0x0000_0000
SWDT Refresh register	SWDT_RR	0x08	32	0x0000_0000
WDT control register	WDT_CR	0x00	32	0x8001_0FF3
WDT status register	WDT_SR	0x04	32	0x0000_0000
WDT Refresh register	WDT_RR	0x08	32	0x0000_0000

24.3.1 Control Register (WDT_CR)

Reset value: 0x8001_0FF3

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

Bit	Marking	Place name	Function	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
b16	SLPOFF	WDT counting Prohibition in Sleep mode	0: WDT count permission in sleep mode 1: WDT Prohibition counting in sleep mode	R/W
b15~b13	Reserved	-	Read as "0", write as "0"	R
b11~b8	WDPT[3:0]	Refresh Allowed Area 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/W
b7~b4	CKS[3:0]	Counting clock	0010: PCLK3/4 0110: PCLK3/64 0111: PCLK3/128 1000: PCLK3/256 1001: PCLK3/512 1010: PCLK3/1024 1011: PCLK3/2048 1101: PCLK3/8192 Remaining values: Reserved function	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R
b1~b0	PERI[1:0]	Counting cycle	00: 256 cycle 01: 4096 cycle 10: 16384 cycle 11: 65536 cycle	R/W

24.3.2 Status Register (SWDT_SR,WDT_SR)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b18	Reserved	-	Read as "0", write as "0"	R
b17	REF	Refresh error flag	0: No refresh error 1: Refresh error occurred to the alignment After reading 1 write 0 and reset.	R/W
b16	UDF	Counting underflow mark	0: No count underflow 1: Occurrence count underflow to the alignment After reading 1 write 0 and reset.	R/W
b15~b0	CNT[15:0]	Count value	Counter current count	R

24.3.3 Refresh Register (SWDT_RR,WDT_RR)

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R
b15~b0	RF[15:0]	Refresh value	Write 0x0123 and 0x3210 to complete the refresh. After the register is written with 0x0123, the read register is 0x0000_0001; the read value of the rest is 0x0000_0000.	R/W

24.4 Precautions for Use

When SWDT operates, the operating frequency of the peripheral clock PCLK3 must be greater than or equal to 4 times the count clock (ICG0.SWDTCKS[3:0] sets the clock after SWDTLRC frequency division), that is, the PCLK3 frequency \geq count clock frequency x4.

25 General Synchronous Asynchronous Transceiver (USART)

25.1 Introduction

The product is equipped with 4 units of General Serial Transceiver Module (USART). General Serial Transceiver Module (USART) can flexibly exchange full-duplex data with external devices. This USART supports UART and smart card interface (ISO/IEC7816-3). Support modem operation (CTS/RTS operation), multiprocessor operation. Cooperate with Timer0 module to support USART receive TIMEOUT function.

Key Features of USART:

- Support full-duplex asynchronous communication, full-duplex clock synchronous communication
- Transmitter and receiver have independent energy position
- Built-in double buffer
- LSB/MSB optional
- Modem operation supported (CTS/RTS)
- Transmission flag: send data register empty, send data complete, receive data register full, receive error flag, USART receive timeout flag

Main Features of UART:

- Data length programmable: 8 bits/9 bits
- Configurable check function: Odd/even/no check
- Stop bit configurable: 1 bit/2 bit
- Timer optional: Internal timer (clock generated by internal baud rate generator)/External timer (clock entered by USARTn _ CK pin)
- Reception error: Check error, frame error, overflow error
- Support for inter-processor communications
- Built-in digital filter
- Support receiving data TIMEOUT function
- Wakeup Unit 1 stop mode Function

Main Features of Clock Synchronization Mode:

- Data length: 8 bits
- Reception error: Overflow error
- Timer: Internal timer (clock generated by internal baud rate generator)/External timer (clock entered by USARTn _ CK pin)

Smart Card Interface Main Features:

- Data length: 8 bits
- Support receiving data TIMEOUT function
- Support data retransmission

25.2 USART System Block Diagram

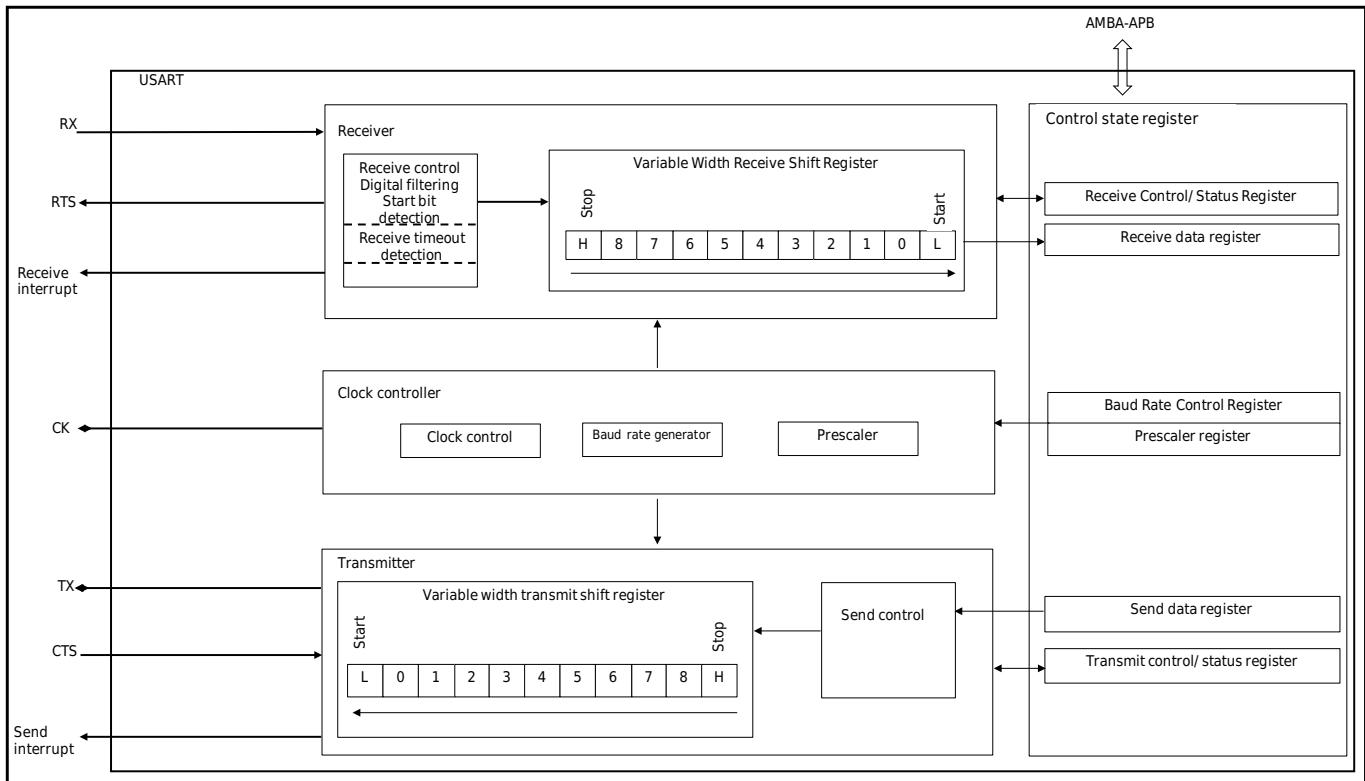


Figure 25-1 USART System Block Diagram

25.3 Pin Description

Table 25-1 USART Pin Descriptions

Pin name	Direction	Functional description
USARTn_CK	Input output	Clock
USARTn_TX	Output	Send data pin
USARTn_RX	Input	Receive data pin
USARTn_CTS	Input	Modem operation pin clear send pin
USARTn_RTS	Output	Modem operation pin Request send pin

n:1~4

25.4 Functional Description

This chapter details the features of UART, Multi-Processor Communication, Smart Card, Clock Synchronization Mode.

25.4.1 UART

25.4.1.1 Clock

UART can select the clock generated by the internal baud rate generator (internal timer) or the clock entered by the USARTn _ CK pin (external timer) as the timer for communication.

Internal timer

USARTn_CR2. When the CLKC [1: 0] bit is set to 00b or 01b, select timer as the clock generated by the internal timer or internal baud rate generator.

USARTn_CR2. The USARTn _ CK pin is not used as a clock pin at CLKC [1: 0] = 00b and can be used as a common IO.

USARTn_CR2.CLKC [1: 0] = 01b outputs the clock at the same frequency as the communication baud rate from the USARTn _ CK pin.

Timer of internal baud rate generator is set to PCLK, PCLK/4, PCLK/16, PCLK/64 by USARTn _ PR. PSC [1: 0] bit.

External timer

USARTn_CR2.CLKC [1: 0] bit is set to 10b or 11b, the timer 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).

Highest baud rate

For internal timer, 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: USARTn _ PR.PSC [1: 0] bit set clock (PCLK,PCLK/4,PCLK/16,PCLK/64) unit: MHz

OVER8: USARTn_CR1.OVER8 setting

DIV _ Integer: USARTn _ BRR.DIV _ Integer setting

The highest baud rate was PCLK/8 (MBps).

When the external clock source is used, the highest 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 highest 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 rate addition to the PCLK-based calculus described above, UART's highest communication baud rate needs to be referred to the highest communication baud rate specified in the Electrical Characteristics chapter.

25.4.1.2 Data Format

A frame of data in UART mode consists of start bits, data bits, check bits, and stop bits.

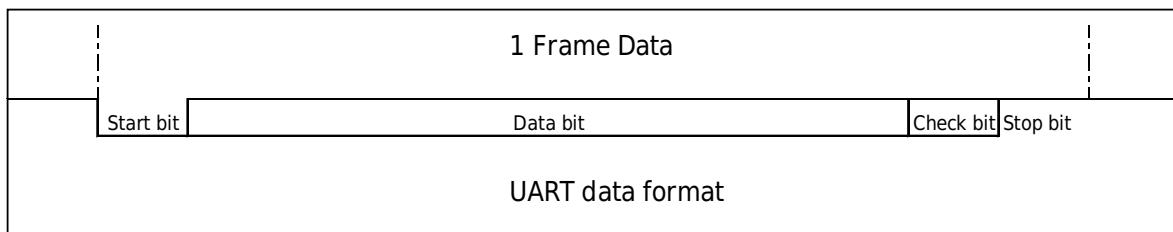


Figure 25-2 UART Data Format

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.

Check bit

The parity bits can be configured as 1-bit parity or 1-bit parity or no parity bits.

Stop bit

Stops Bit a fixed high level and can be configured as 1 or 2 bits.

25.4.1.3 Modem Operation

Modem operations include CTS and RTS functions. CTS and RTS features can only be selected one at a time and cannot be used simultaneously. The RTS function is valid when USARTn_CR3.CTSE=0, and the CTS function is valid when USARTn_CR3.CTSE=1.

CTS function

The CTS function controls the sending of data through the input of the USARTn_CTS pin. The data can be sent only when the USARTn_CTS pin enters the low level. The data being sent is not affected if the USARTn_CTS pin enters the high level.

RTS function

The RTS feature is a low level output from the USARTn_RTS pin to request the other party to send data.

The USARTn_RTS pin output low level needs to meet all of the following conditions:

- Receive enable(USARTn_CR1.RE=1) and not receiving data
- No unread received data in USARTn_DR.RDRregister
- No receive errors, including frame errors, check errors and overflow errors

25.4.1.4 Transmitter

The transmitter can send 8-bit or 9-bit data, depending on the setting value of the USARTn_CR1.M bit.

Transmitter enable bit (USARTn_CR1.TE) 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 sequence of sending data is: Start bit -> Data bit (MSB/LSB) -> Check bit (with or without) -> Stop bit.

The transmission data registerTDR and the internal transmission shift register constitute a dual buffer structure, which can send data continuously.

To ensure the correctness of sending data, a request can only be written once by sending data register empty interrupt or DMA.

Sending data setting procedure

1. Set USARTn_CR1register to reset
2. Set the pin that UART needs to use
3. Through USARTn_CR2.CLKC [1: 0] bit selection timer
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3register
5. Set USARTn_PR to select the pre-frequency division value and USARTn_BRRregister to set the communication baud rate (not required when timer is external timer).
6. Enable the transmitter (USARTn_CR1.TE=1), if you need to use the transmit data register empty interrupt, set USARTn_CR1.TXEIE=1 (write 1 to TE and TXEIE bits at the same time)
7. Wait to send data register empty, write communication data to USARTn_DR.TDR, transfer data to send shift register, send start

(When the CTS feature is active, the USARTn_CTS input is at low level and the data is transmitted to the send shift register, and the send starts)

8. Repeat Step 7 if you need to send data continuously
9. Confirm that the send is complete by confirming the USARTn_SR.TC bit. To continuously send data and use the sending interrupt, the last sent data can be written via Tlinterrupt and USARTn_CR1.TXEIE write 0, USARTn_CR1.TCIE writes 1. After the last frame of data is sent, the transmission is interruptd.

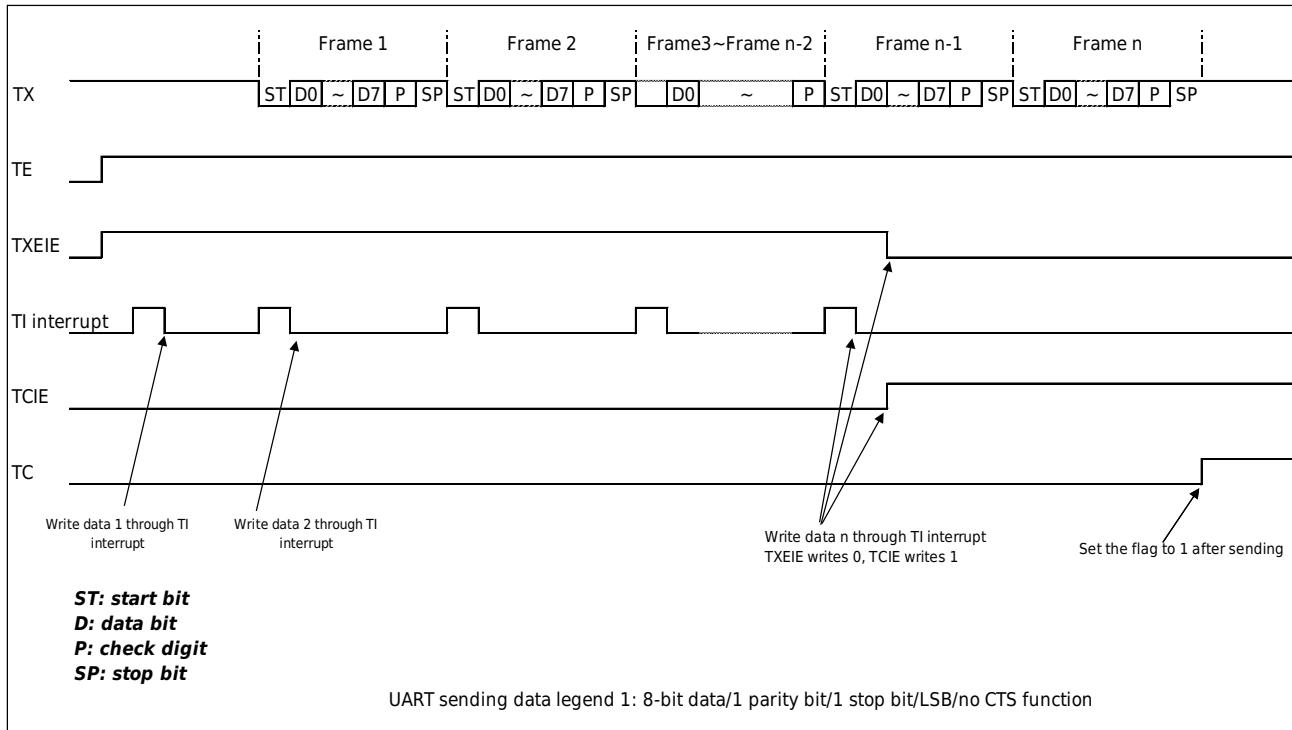


Figure 25-3 UART send data legend 1

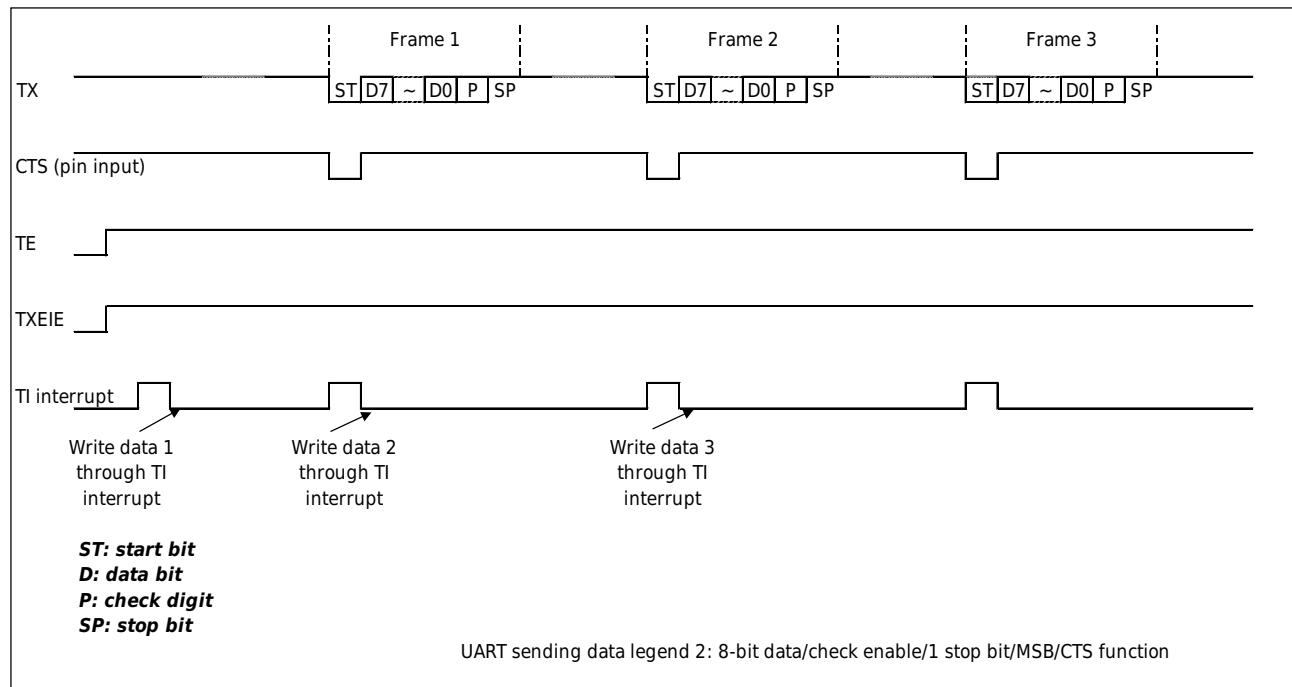


Figure 25-4 UART send data legend 2

Transmitter Interrupt

The UART mode transmitter supports two types of interrupts, sending data register null interruptTI and sending complete interruptTCI.

$\text{TXEIE} = 1$, USARTn_DR.TDR register value sent to send shift register TI interrupt occurs.

$\text{TCIE} = 1$, TCI interrupt occurs when USARTn_DR.TDR register is not updated when the last bit of data is sent.

25.4.1.5 Receiver

Receivers can receive 8 or 9 bits of data depending on USARTn_CR1 . The set value of the M bit. After the receiver sets the energy bit (USARTn_CR.RE) to 1 and detects the start bit, the RX pin receives the received shift register, fills a frame of data, and the data is transferred from the received shift register to the received data register USARTn_DR.RDR .

The sequence of receiving data is: Start bit -> Data bit (MSB/LSB) -> Check bit (with or without) -> Stop bit.

The receiving data register USARTn_DR.RDR register and the internal receiving 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, USARTn_CR1 . Low level detection at $\text{SBS} = 0$, USARTn_CR1 . When $\text{SBS} = 1$, it was the detection of falling edge.

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, $\text{USARTn_CR1.OVER8} = 0$ is sampled at the 8th internal base clock and USARTn_MR . $\text{OVER8} = 1$ is sampled at the 4th internal base clock.

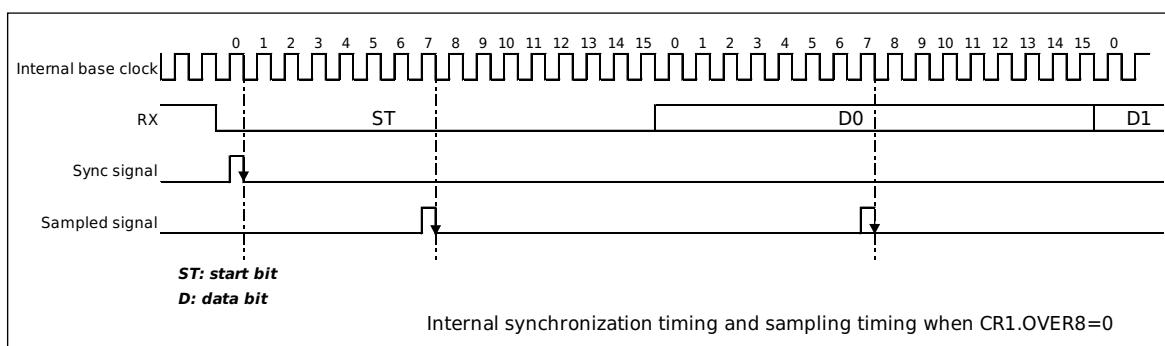


Figure 25-5 UART internal synchronization and sampling timing

The UART asynchronous receiver works only if the total clock system deviation 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
- 8x or 16x oversampling defined by the OVER8 bit in the USART_CR1 register
- Whether to use the fractional baud rate defined by the FBME bit in the USART_CR1 register

Table 25-2 Tolerance of UART receiver when DIV_Fraction is 0

FL	OVER8 bit=0	OVER8 bit=1
10	4.375%	3.75%
11	3.97%	3.41%
12	3.646%	3.125%

Table 25-3 Tolerance of UART receiver when DIV_Fraction is not 0

FL	OVER8 bit=0	OVER8 bit=1
10	3.88%	3%
11	3.53%	2.73%
12	3.23%	2.5%

In special cases, when the received frame contains an idle frame of 10 or 11 or 12 bit times, the data specified in Table25-2 and Table 25-3 may differ slightly.

Receiving data setting procedure

1. Set USARTn _ CR1register to reset
2. Set the pin that UART needs to use
3. Through USARTn _ CR2.CLKC [1: 0] bit selection timer
4. Set USARTn _ CR1, USARTn _ CR2, USARTn _ CR3register
5. Set USARTn _ PR to select the pre-frequency division value and USARTn _ BRRregister to set the communication baud rate (not required when timer is external timer).
6. Enable the 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 receives the received shift register and checks the check bit and stop bit

 - 1) When an error is checked, the received data is transferred to the USARTn_DR.RDRregister 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.RDRregister and the USARTn_SR.FE flag is set.
 - 3) When an overflow 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

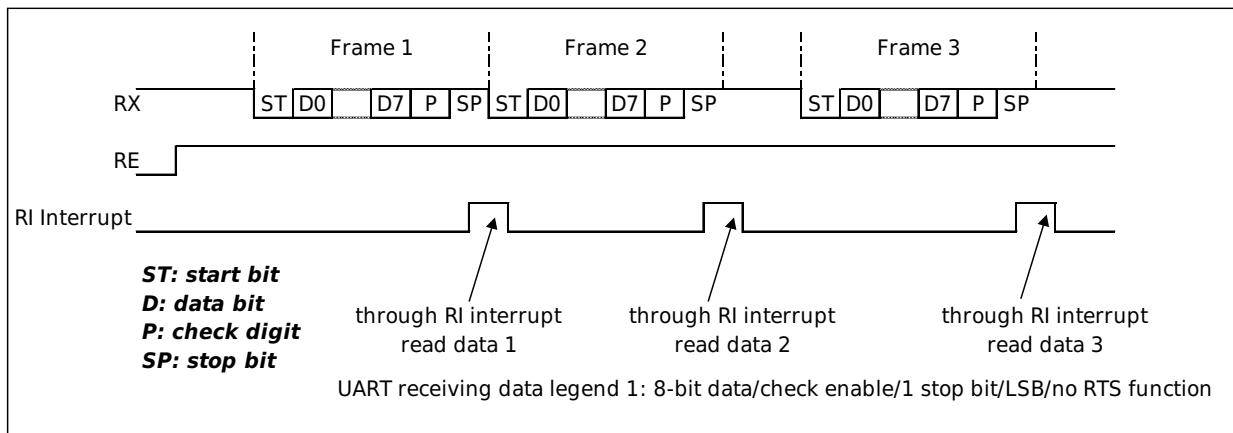


Figure 25-6 UART Received Data Legend 1

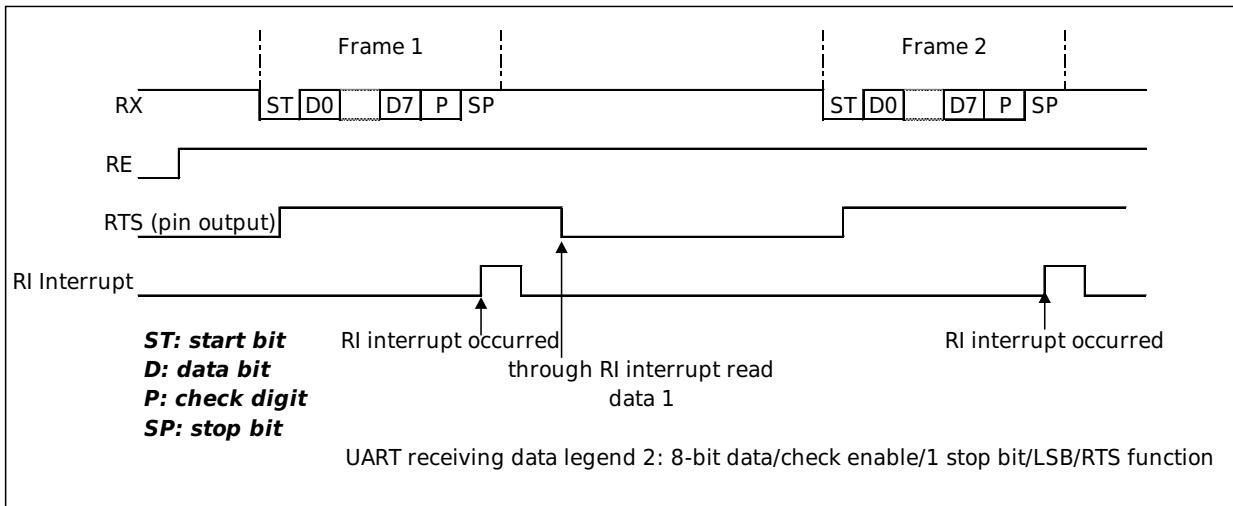


Figure 25-7 UART Received Data Legend 2

Error handling

There are three types of receiving errors at the time of receiving data: Overflow error (USARTn_SR.ORE), verification 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 zeroing register.

The overflow error occurs on condition that a new frame of data is received without the USARTn_DR.RDRRegister 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 overflow error occurs, RI interrupt does not occur.

When a parity error occurs, the received data is sent to USARTn_DR.RDR, and the RI interrupt does not occur.

When a framing error occurs, the received data is sent to USARTn_DR.RDR, and the RI interrupt does not occur.

Receiver interrupt

The UART mode receiver supports two types of interrupts, receive data register full interruptRI and receive error interruptREI.

USARTn_CR.RIE = 1, no receiving error occurred, and RI interrupt occurred when data was transferred from the receiving shift register to the receiving data register.

USARTn_CR.RIE = 1. An overflow error, a check error, or a frame error occurs during reception.

25.4.1.6 UART Receive TIMEOUT Function

When the stop bit of UART received data is detected, the TIMEOUT counter starts. When the next frame of received data is not detected after the set TIMEOUT time (the setting unit is the number of received bits), TIMEOUT occurs. 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 USARTn_CR1.RE=1 and then the TIMEOUT status bit USARTn_SR.RTOF is set.

The TIMEOUT counter uses the counter of the Timer0 module, and the specific correspondence is as follows:

USART1: Timer0 Unit1 A channel

USART2: Timer0 Unit1 B channel

USART3: Timer0 Unit2 A channel

USART4: Timer0 Unit2 B channel

TIMEOUT Function Timer0 Comparison Counter Value Setting

Timer0 is a 16-bit counter, and the counting clock can be divided by up to 1024. The calculation formula for TMR0_CMPAR value setting is as follows:

$$\text{CMPA}_{<\text{B}>} \text{R} = \frac{\text{RTB}}{2^{\text{CKDIVA}_{<\text{B}>}}} - 4 \text{ (Counting clock without frequency division)}$$

$$\text{CMPA}_{<\text{B}>} \text{R} = \frac{\text{RTB}}{2^{\text{CKDIVA}_{<\text{B}>}}} - 2 \text{ (Count clock 2 division)}$$

$$\text{CMPA}_{<\text{B}>} \text{R} = \frac{\text{RTB}}{2^{\text{CKDIVA}_{<\text{B}>}}} - 1 \text{ (Counting clock 4 divisions and above)}$$

CMPAR: TMR0_CMPAR register value

RTB: Receive Timeout Bits

CKDIRA: TMR0.BCONR.CKDIVA bit register value

TIMEOUT Function Setting Procedure

1. Set USARTn _ CR1register to reset
2. Set the pin that UART needs to use
3. Select the clock source by USARTn_CR2.CLK[1:0] bits (if you choose the internal clock source, you need to set CR2.CLKC[0]=1)
4. Set USARTn _ CR1, USARTn _ CR2, USARTn _ CR3register
5. Set USARTn _ PR to select the pre-frequency division value and USARTn _ BRRregister to set the communication baud rate (not required when timer is external timer).
6. Clear the USARTn_SR.RTOF flag, set USARTn_CR1.RTOE=1, if you need to use interrupt, set USARTn_CR1.RTOIE=1
7. Set TMR0.BCONR.CSTA=0
8. Set TMR0_CNTAR to 0, set TMR0_CMPAR register and TMR0.BCONR.CKDIVA register to determine TIMETOU time,
Set TMR0.BCONR.HCLEA= 1, TMR0.BCONR.HSTAA= 1,
TMR0.BCONR.ASYNCLKA= 1, TMR0.BCONR.SYNSA= 1
9. Enable the receiver (USARTn_CR1. RE=1), if you need to use receive interrupt, set USARTn_CR1. RIE=1
10. Set TMR0.BCONR.CSTA=0 after detecting TIMEOUT,
Set TMR0.BCONR.CSTA=0, clear USARTn_SR.RTOF status bit by writing USARTn_CR1.CRTOF.

25.4.1.7 RX Line Stop Mode Wake-Up Enable

When the UART communication is idle, the system can enter the stop mode to save current consumption. Without changing the UART PORT setting, UART unit 1 can wake up the system from the stop mode through the RX line. Specific steps are as follows:

1. When the UART communication is idle, set the USART_1_WUPI interrupt vector and

INT_WUPEN. The RXWUEN bit enables the UART receive signal line to wake up the stop mode function.

2. System Enter stop mode.
3. When the system detects the falling edge of the RX line, it returns from the stop mode and disables this 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 send a frame of wake-up data (recommended to be 0x00), this data will not be received by UART and the relevant flag will not be set. And the communication party needs to pass the time required for the system stop mode to wake up before transmitting the UART communication data.

25.4.1.8 UART Interrupt And Events

Table 25-4 UART interrupt/Event Table

Functional name	Marking	Enable bit (interrupt only)	Flag bit	Can I be the source of an event
Reception error interrupt	REI	RIE	ORE,FE,PE	Yes
Received data register full interrupt	RI	RIE	No	Yes
Send data register empty interrupt	TI	TXEIE	No	Yes
Send complete interrupt	TCI	TCIE	TC	Yes
TIMEOUT interrupt	RTOI	RTOIE	RTOF	Yes
RX stop mode wake-up interrupt	WUPI	INT_WUPEN.RXWUEN	-	No

25.4.2 Multiprocessor Communication

25.4.2.1 Feature Brief

Multi-processor 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.

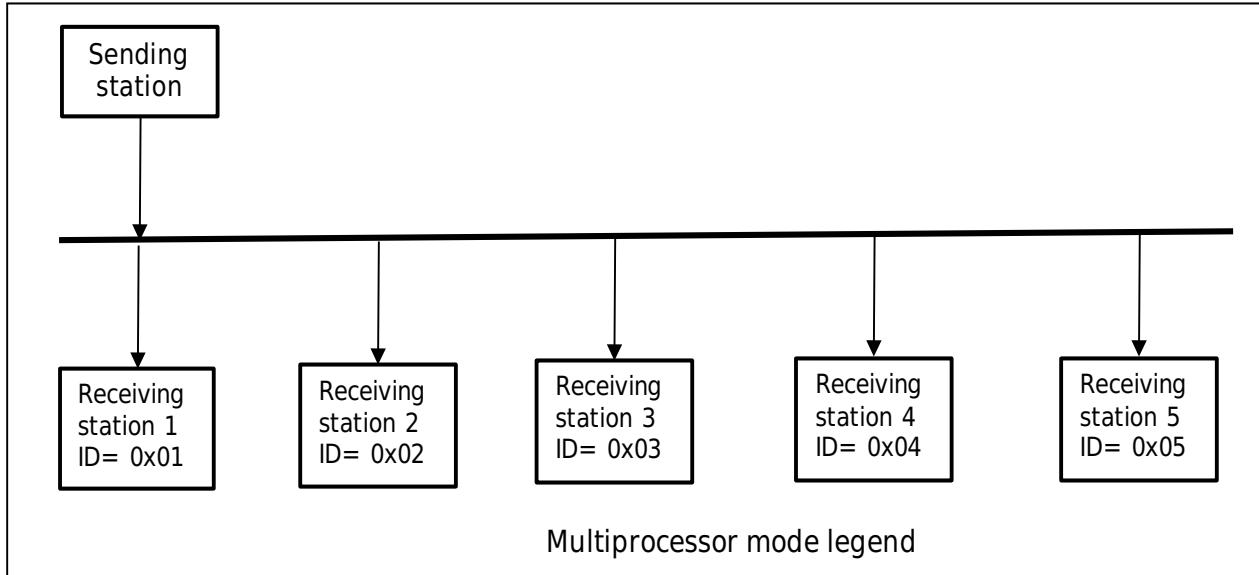


Figure 25-8 Multiprocessor Communication Diagrams

25.4.2.2 Data format

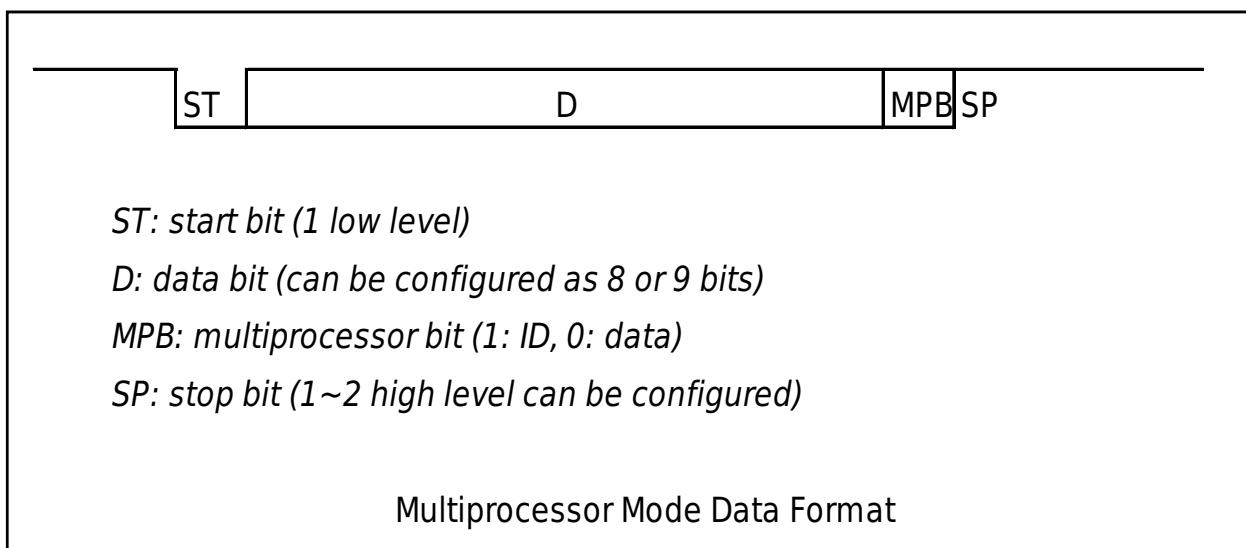


Figure 25-9 Multiprocessor Mode Data Format

25.4.2.3 Action Description

In the multi-processor mode, the check bit function is invalid, and the multi-processor bit function is added. The other functions such as clock and interrupt are the same as UART mode.

Transmitting station action

1. Set USARTn _ CR1register to reset
2. Set the pins to be used
3. Through USARTn _ CR2.CLKC [1: 0] bit selection timer
4. Set USARTn _ CR1, USARTn _ CR2, USARTn _ CR3register
5. Set USARTn _ PR to select the pre-frequency division value and USARTn _ BRRregister to

- set the communication baud rate (not required when timer is external timer).
6. Enable the transmitter (USARTn_CR1.TE=1), if you need to use the transmit data register empty interrupt, set USARTn_CR1.TXEIE=1 (write 1 to TE and TXEIE bits at the same time)
 7. Wait for send data register to be empty, set USARTn_DR.MPID bit to 1 (send ID), write ID to USARTn_DR, send ID
(When the CTS feature is active, the USARTn_CTS input is at low level and the data is transmitted to the send shift register, and the send starts)
 8. Set USARTn_DR.MPID bit to 0 (send data), write data to USARTn_DR, send data
(When the CTS feature is active, the USARTn_CTS input low level data transfer to the send shift register, send start)
 9. If you need to send the data continuously, repeat step 8. If you need to change the ID, repeat steps 7 and 8.
 10. Confirm that the send is complete by confirming the USARTn_SR.TC bit. To continuously send data and use the sending interrupt, the last sent data can be written via 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.

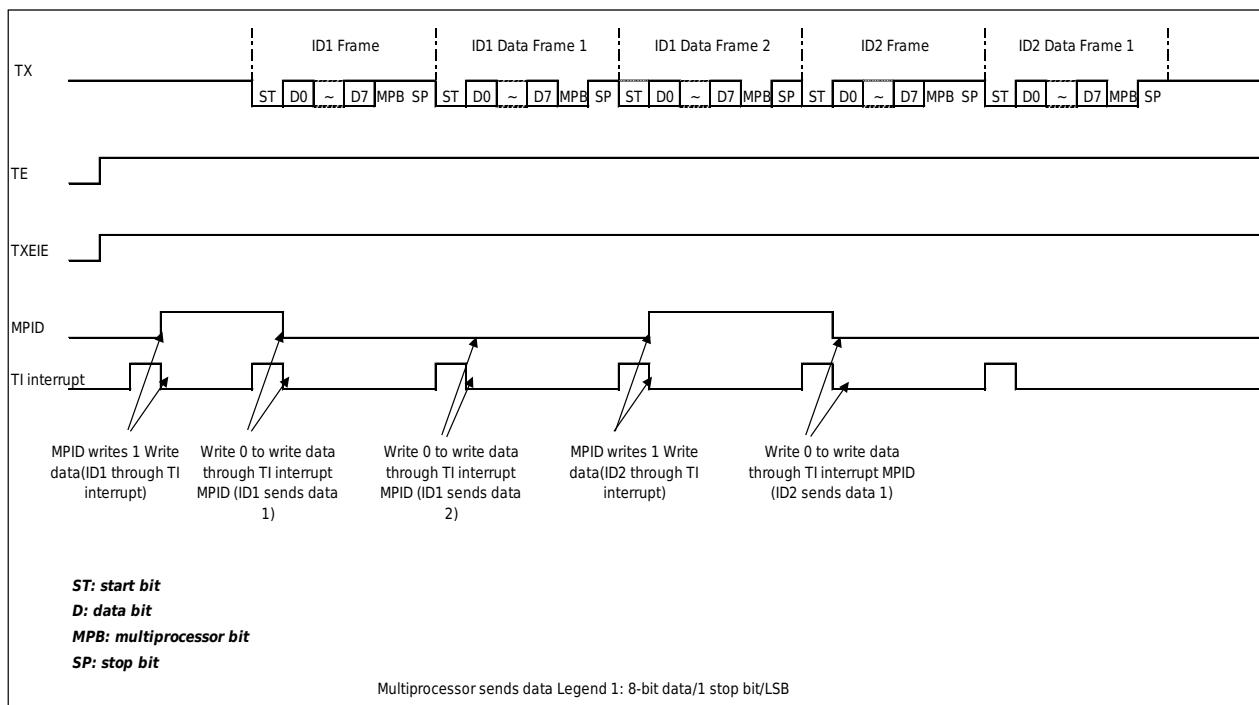


Figure 25-10 Diagram of Multi-Processor Mode Send Data

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 by the USARTn_CR1.SLME bit.

USARTn_CR1. Receive data normally at SLME = 0.

USARTn_CR1. When 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 timer
4. Set USARTn _ CR1, USARTn _ CR2, USARTn _ CR3register
5. Set USARTn _ PR to select the pre-frequency division value and USARTn _ BRRregister to set the communication baud rate (not required when timer is external timer).
6. USARTn_CR1.RE=1, USARTn_CR1.SLME=1 (waiting to receive ID), if using receive interrupt, set USARTn_CR1.RIE=1
7. When the start bit is detected, the receiver receives the received 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 is normally received, interruptd, and the error is detected, the same as the data received by UART
 - 2) If the ID does not match, the software will USARTn _ CR1 again. SLME bit write 1, repeat 8

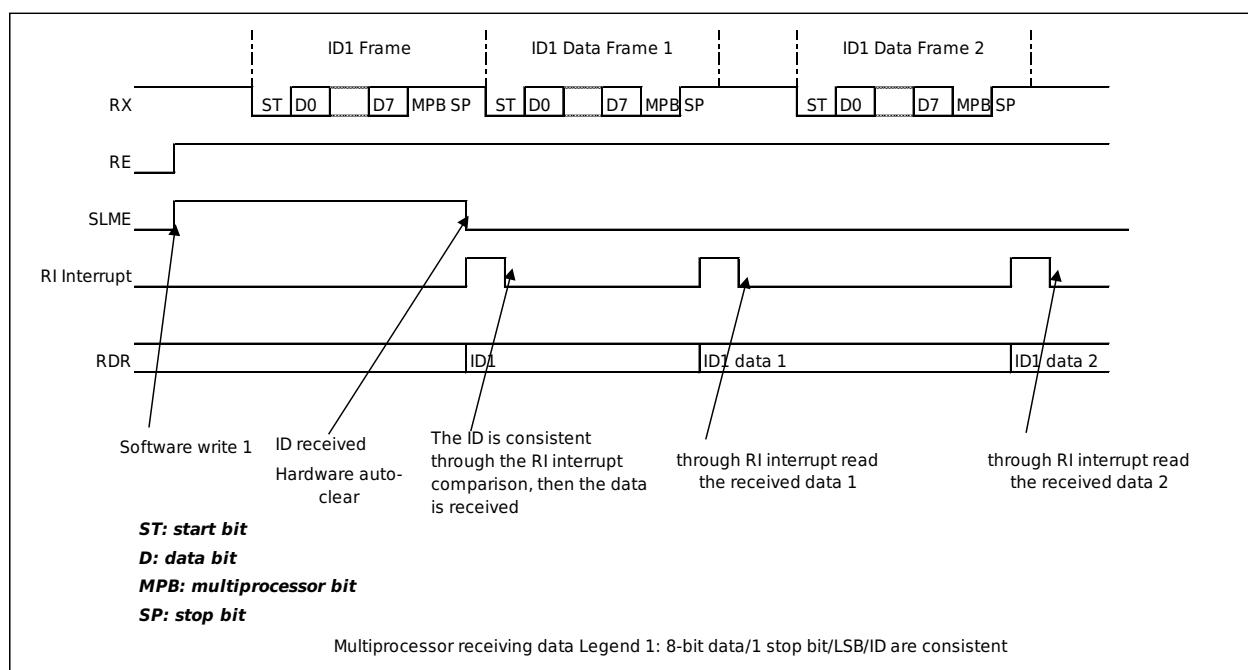


Figure 25-11 Diagram of Multiprocessor Mode Receive Data 1

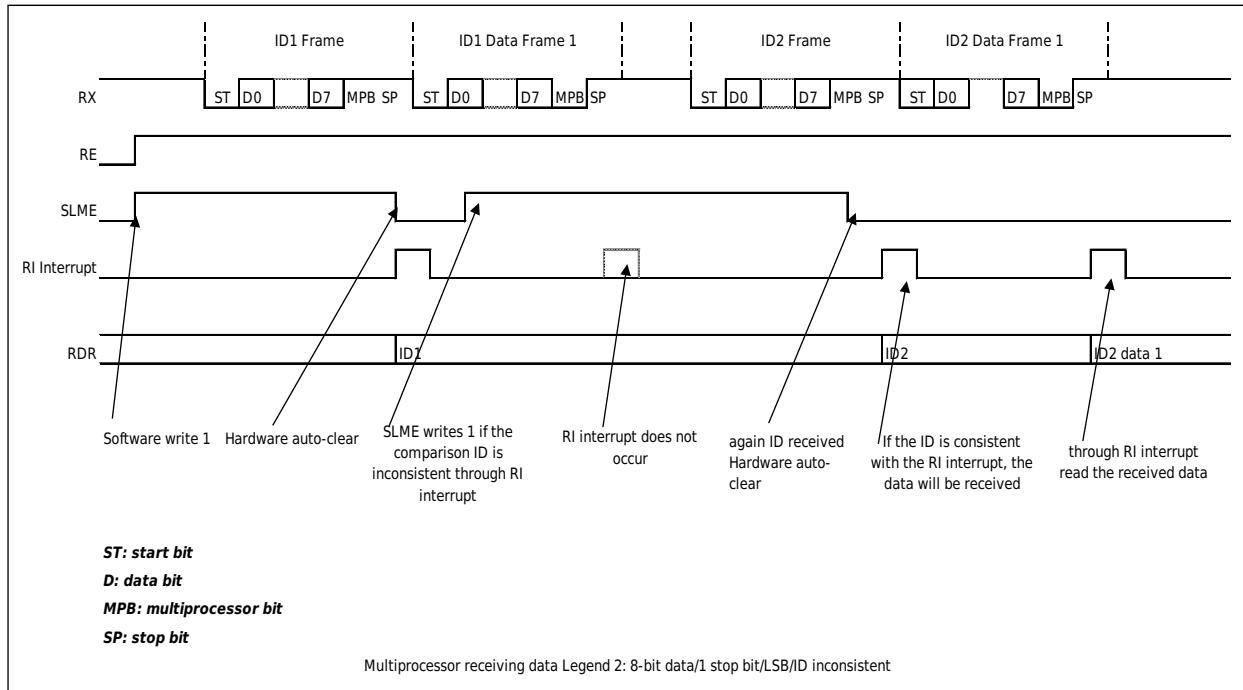


Figure 25-12 Diagram of Multiprocessor Mode Receive Data 2

25.4.2.4 Interrupt and Events

The interrupt mode is the same as the UART mode except for no check error.

Table 25-5 Multi-Processor Mode Interrupt/ Event

Functional name	Marking	Enable bit (interrupt only)	Mark	Can I be the source of an event
Reception error interrupt	REI	RIE	ORE,FE	No
Received data register full interrupt	RI	RIE	RXNE	Yes
Send data register empty interrupt	TI	TXEIE	TXE	Yes
Send complete interrupt	TCI	TCIE	TC	No

25.4.3 Smart Card

25.4.3.1 Connection Diagram

Support the smart card communication protocol specified by ISO/IEC 7816-3. underflow - Smart card Pattern mode with connected legend.

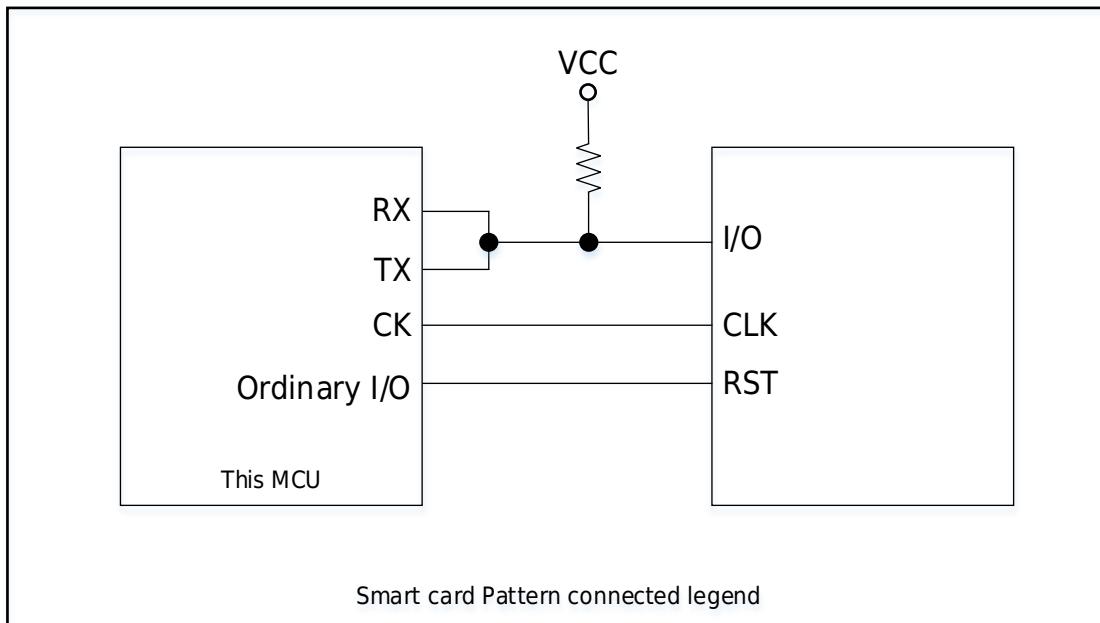


Figure 25-13 Smart Card Pattern Connected Legend

25.4.3.2 Clock

Internal timer

Only the clock generated by the internal baud rate generator can be used as the clock source in smart card mode.

The basic clock number for one-bit data transmission is the setting value of USARTn_CR3.BCN[2:0].

The clock output of smart card mode is controlled by setting register USARTn_CR2.CLKC[1:0].

Highest Baud Rate

For internal timer, the Baud Rate formula generated by the internal Baud Rate Generator is:

$$B = \frac{C}{2 \times BCN \times (DIV_Integer + 1)}$$

B: Baud rate unit: MBps

C: USARTn_PR.PSC [1: 0] bit set clock (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 highest baud rate is PCLK/64(MBps).

Sampling and reception tolerance

After detecting the falling edge of RX, the USART will clock the received data based on the internal basic clock to start data reception. Received data will be sampled at the data center.

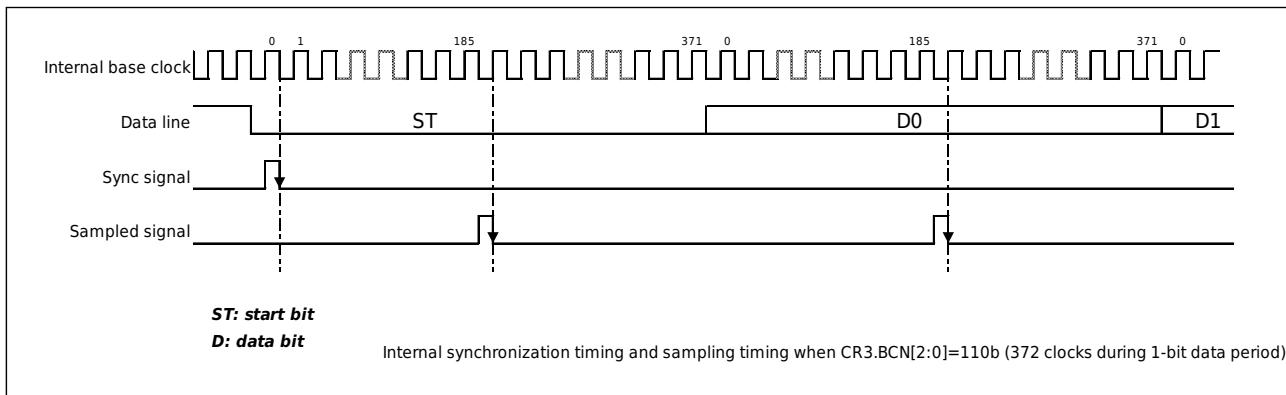


Figure 25-14 Smart Card Pattern synchronization and sampling timing Figure

The following formula is used to calculate the receiving tolerance:

$$RM[\%] = |0.5 \times \left(1 - \frac{1}{BCN}\right) - 9.5CFD| \times 100$$

RM: reception tolerance

BCN: The number of clocks required for one bit data transmission (USARTn.CR3.BCN[2:0] setting value)

CFD: Clock frequency deviation

25.4.3.3 Data format

In smart card mode, a frame of data consists of start bit, data bit and check bit.

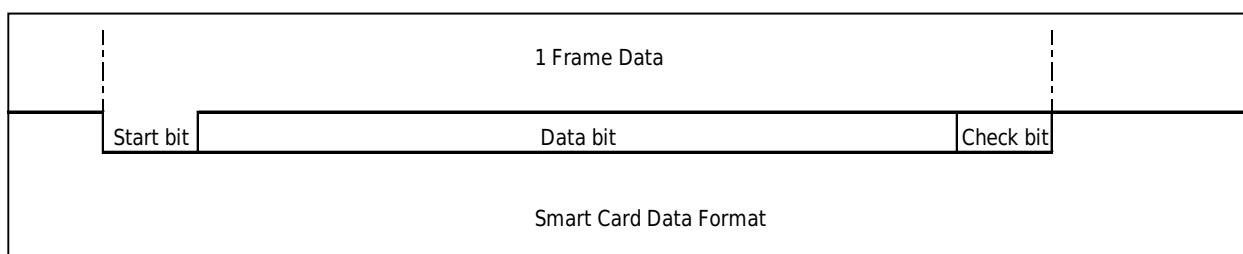


Figure 25-15 Smart Card Pattern synchronization and sampling timing Figure

Start bit

The start bit is fixed with a low level.

Data bit

The data bits are fixed to 8-bit data.

Check bit

The parity bit needs to be configured as 1-bit even parity.

25.4.3.4 Initial Setting Procedure of Smart Card

1. Set USARTn _ CR1register to reset
2. Set the pins to be used
3. Set USARTn _ CR1, Status Register and confirm that SRregister to reset
4. Set USARTn _ CR1, USARTn _ CR2, USARTn _ CR3register
5. Set USARTn _ PR to select the pre-frequency division value and USARTn _ BRRegister to set the communication baud rate
6. USARTn_CR2.CLKC[1:0] bits set clock control
7. USARTn_CR1 register (TE, RE, RIE, TXEIE bit) setting, except for self-test, TE and RE should not be set to 1 at the same time

When switching from sending mode to receiving mode, or switching from receiving mode to sending mode, you need to reset the above steps 1 to 7.

25.4.3.5 Smart Card Pattern Action description

In smart card mode, the flag bit of TI interrupt (send data empty interrupt) is USARTn_SR.TC bit. TI interrupt is generated when USARTn_SR.TC=1 and USARTn_CR1.TXEIE=1.

Functional Overview

When sending data, there is a protection time of more than 2 etu (Elementary Time Unit) between two frames of data (from the end of the parity bit to the start bit of the next frame).

When sending data, if an error signal sent by the receiver is detected, the data will be resent automatically after 2 etu.

When a verification error occurs in the received data, a low level of 1etu is sent, which is an error signal. The timing of sending the error signal is 10.5etu after the start of reception.

send instructions

1. After a frame of data is sent, if an error signal sent by the receiver is detected, USARTn_SR.FE is set to 1 (if USARTn_CR.RIE=1, an error interrupt occurs), the USARTn_SR.TC flag is not set to 1, and the data is automatically retransmitted. The USARTn_SR.FE bit must be cleared before receiving the next frame parity bit.
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. Write data again, then the data can be sent continuously.

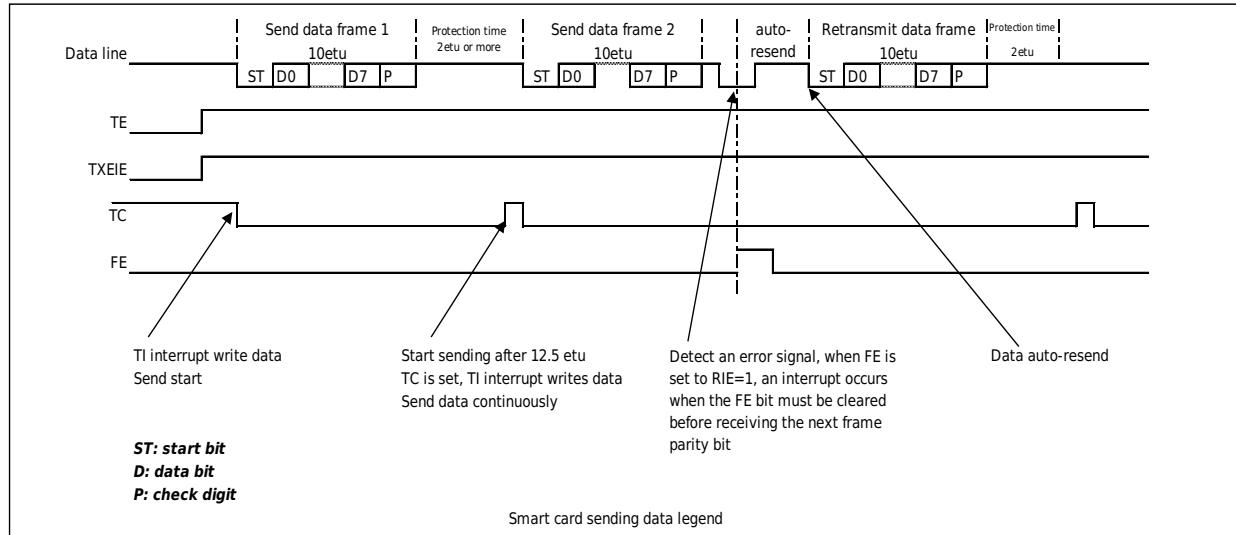


Figure 25-16 Smart Card Pattern Sending Data Legend

Receiving Instructions

- When receiving data, if a parity error is detected, USARTn_SR.PE will be set, and when the interrupt is enabled, a REI interrupt will occur. DR.RDRregister and the USARTn _ SR.PE bit must be cleared before receiving the next frame parity bit.
- When a verification error occurs, a low level of 1etu will be sent, that is, an error signal, requiring the sender to resend the data.
- To receive data normally, you can read the received data through RI interrupt and receive continuously.
- When receiving data, an overflow error is detected.

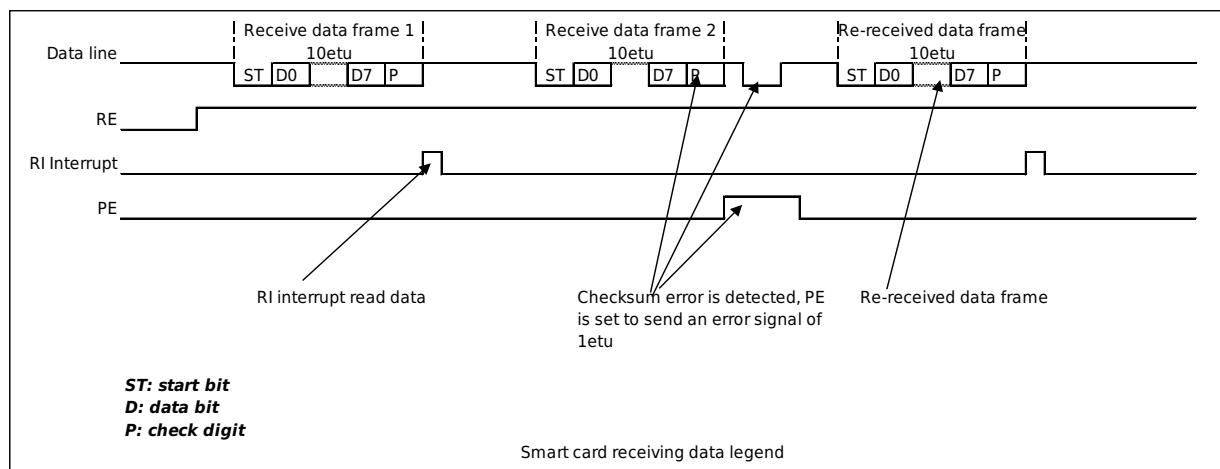


Figure 25-17 Smart card Pattern Receive data legend

25.4.3.6 Interrupt and Events

Table 25-6 Smart Card Mode Interrupt/Event

Functional name	Marking	Enable bit (interrupt only)	Flag bit	Can I be the source of an event
error interrupt	REI	RIE	ORE,PE,FE	Yes
Received data full Interrupt	RI	RIE	RXNE	Yes
Send data null Interrupt	TI	TXEIE	TC	Yes

25.4.4 Clock synchronization mode

25.4.4.1 Clock

The clock synchronization mode can select the clock generated by the internal baud rate generator (internal timer) or the clock input from the USARTn _ CK pin (external timer) as the timer for communication.

Internal timer

The synchronization clock is output from the USARTn _ CK pin, and a frame of data outputs eight clock pulses. When neither the data nor the data is sent, the clock output is fixed at a high level.

External timer

The external timer is the input clock from the USARTn _ CK pin as the communication clock.

Highest baud rate

For internal timer, 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: USARTn _ PR.PSC [1: 0] bit set clock (PCLK,PCLK/4,PCLK/16,PCLK/64) unit: MHz

DIV _ Integer: USARTn _ BRR.DIV _ Integer setting

When C is PCLK and DIV _ Integer = 1, the highest baud rate is PCLK/8 (MBps). Note that the DIV _ Integer disable is set to 0 in synchronous mode.

For external timer, the maximum frequency of external input clock is PCLK (MHz)/6, so the maximum baud rate is PCLK/6 (MBps).

It should be noted that in rate addition to the PCLK-based calculus described above, the Synchronous Mode Maximum Communication Potter also needs to refer to the Maximum Communication Baud Rate specified in the Electrical Characteristics chapter.

25.4.4.2 Data Format

One frame of data in clock synchronization mode is composed of eight bits, and eight synchronization clock pulses are needed to send and receive one frame of data. Data is sent at the falling edge of the synchronization clock when data is sent and sampled at the rising edge of the synchronization clock when data is received.

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.

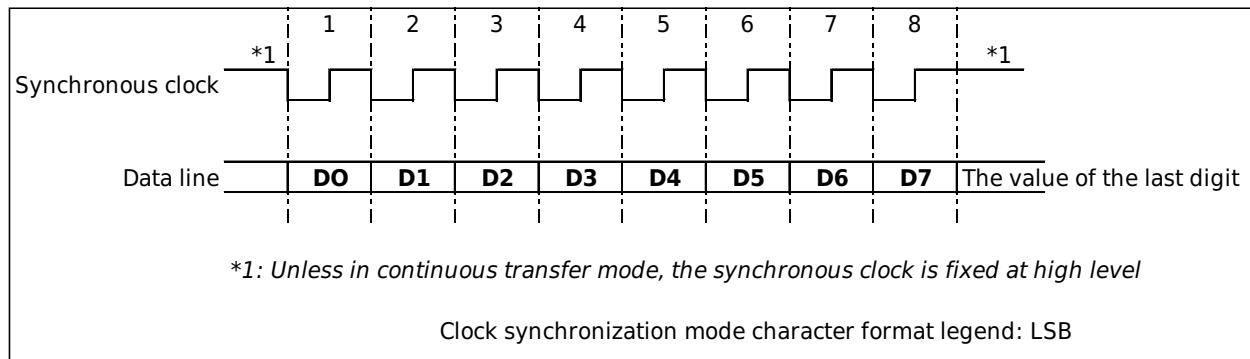


Figure 25-18 Clock Synchronization Mode Data Format

25.4.4.3 Modem Operation

Modem operations include CTS and RTS functions. The CTS function and the RTS function can only be selected from one of the two, and cannot be used at the same time. The RTS function is valid when USARTn_CR3.CTSE=0, and the CTS function is valid when USARTn_CR3.CTSE=1.

CTS Function

The CTS function controls the sending of data through the input of the USARTn_CTS pin. The data can be sent only when the USARTn_CTS pin enters the low level. The data being sent is not affected if the USARTn_CTS pin enters the high level.

RTS Function

The RTS feature is a low level output from the USARTn_RTS pin to request the other party to send data.

The USARTn_RTS pin output low level needs to meet all of the following conditions:

- Receive enable (USARTn_CR1.RE = 1) and not receiving data
- No unread data in USARTn_DR.RDRRegister (USARTn_CR1.RE = 1)
- The update of USARTn_DR.TDR is completed (when USARTn_CR1.TE=1)
- No reception error

If all the above conditions are not met simultaneously, USARTn_RTS outputs a high level.

25.4.4.4 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 registerUSARTn _ DR.TDR and the internal transmission shift register constitute a dual buffer structure, which can send data continuously.

To ensure the correctness of sending data, a request can only be written once by sending data register empty interrupt or DMA.

Sending Data Setting Procedure

1. Set USARTn _ CR1, USARTn _ SR1register to reset
2. Set the pins to be used
3. Through USARTn _ CR2.CLKC [1: 0] bit selection timer
4. Set USARTn _ CR1, USARTn _ CR2, USARTn _ CR3register
5. Set USARTn _ PR to select the pre-frequency division value and USARTn _ BRRregister to set the communication baud rate (not required when timer is external timer).
6. Enable the transmitter (USARTn_CR1. TE=1), if you need to use the transmit data register empty interrupt, set USARTn_CR1. TXEIE=1 (write 1 to TE and TXEIE bits at the same time)
7. Wait to send data register empty, write communication data to USARTn _ DR.TDR, transfer data to send shift register, send start
(When the CTS feature is active, theUSARTn_CTS input is at low level and the data is transmitted to the send shift register, and the send starts)
8. Repeat Step 7 if you need to send data continuously
9. Confirm that the send is complete by confirming the USARTn _ SR.TC bit. To continuously send data and use the sending interrupt, the last sent data can be written via Tlinterrupt and USARTn _ CR1.TXEIE write 0, USARTn _ CR1.TCIE writes 1. After the last data is sent, the transmission is interruptd.

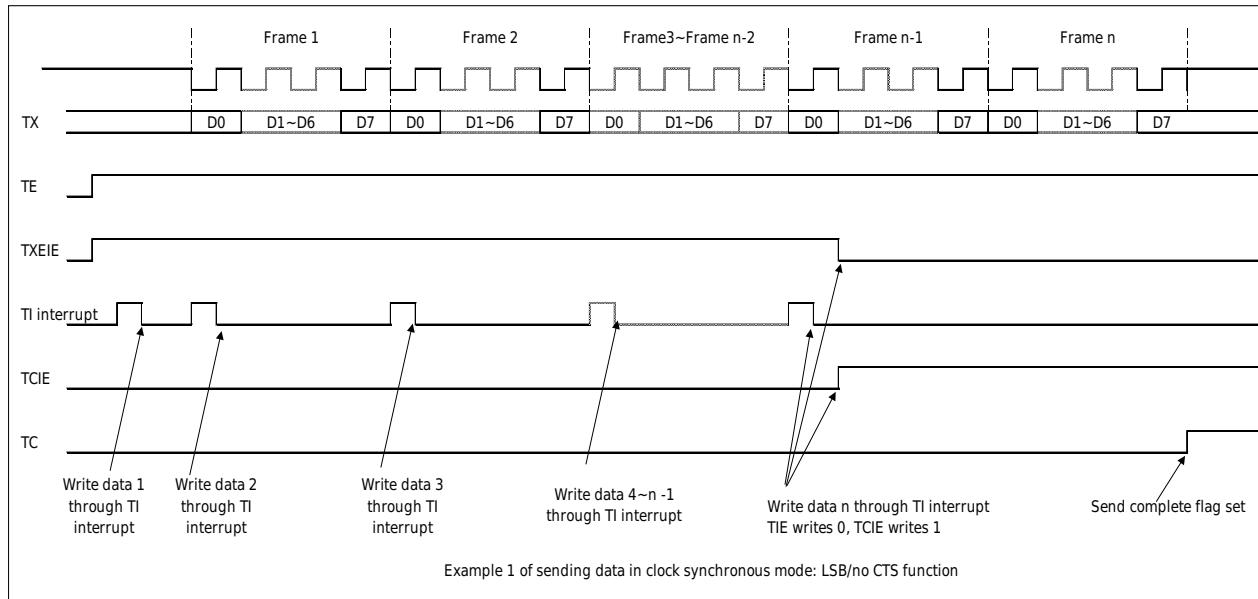


Figure 25-19 Diagram of Clock Synchronization Mode Sending Data 1

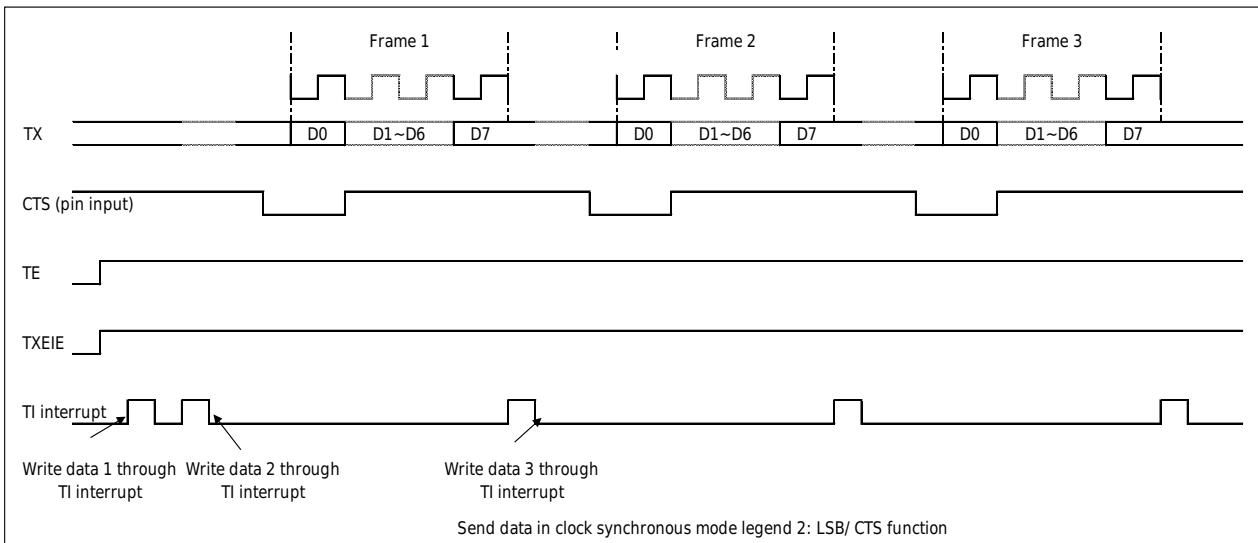


Figure 25-20 Diagram of Clock Synchronization Mode Sending Data 2

Transmitter interrupt

The Clock Synchronization Mode Transmitter supports two types of interrupts, sending data register null interrupt TI and sending complete interrupt TCI.

TXEIE = 1, USARTn_DR.TDR register value sent to send shift register TI interrupt occurs.

TCIE=1, TCI interrupt will occur if the USARTn_DR.TDR register is not updated when the last bit of data is sent.

25.4.4.5 Receiver

Receiving data setting procedure

1. Set USARTn_CR1, USARTn_SRregister to reset
2. Set the pins to be used
3. Through USARTn_CR2.CLKC [1: 0] bit selection timer
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3register
5. Set USARTn_PR to select the pre-frequency division value and USARTn_BRRregister to set the communication baud rate (not required when timer is external timer).
6. Enable the receiver (USARTn_CR1.RE=1), if you need to use receive interrupt, set USARTn_CR1.RIE=1
(When using the RTS feature, RE = 1 rear USARTn_RTS output low level)
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 overflow 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, and then step 7 is repeated to achieve continuous Receive data function

(When using the RTS function, USARTn_RTS outputs a low level after data is read)

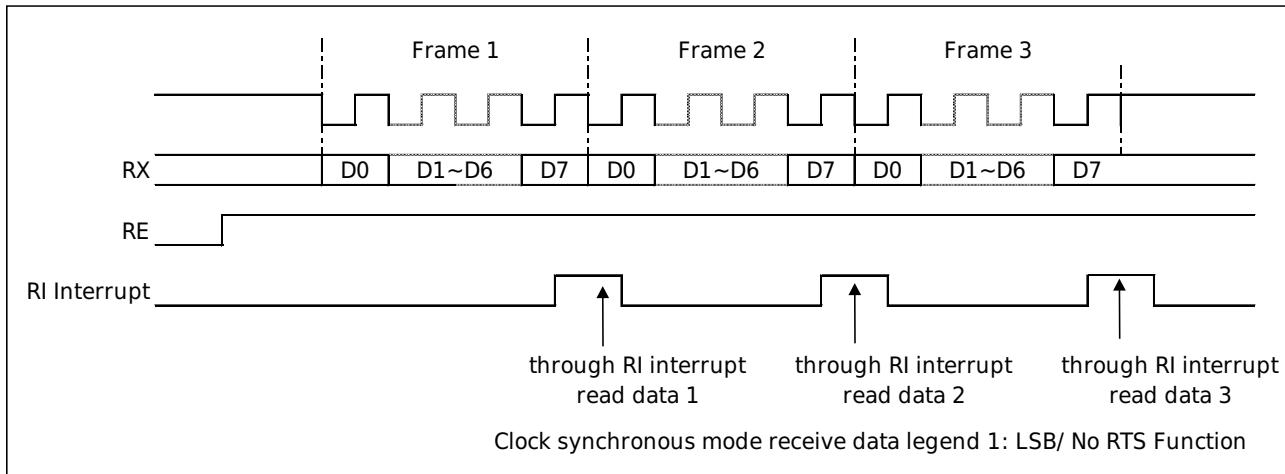


Figure 25-21 Diagram of Clock Synchronization Mode Receive Data 1

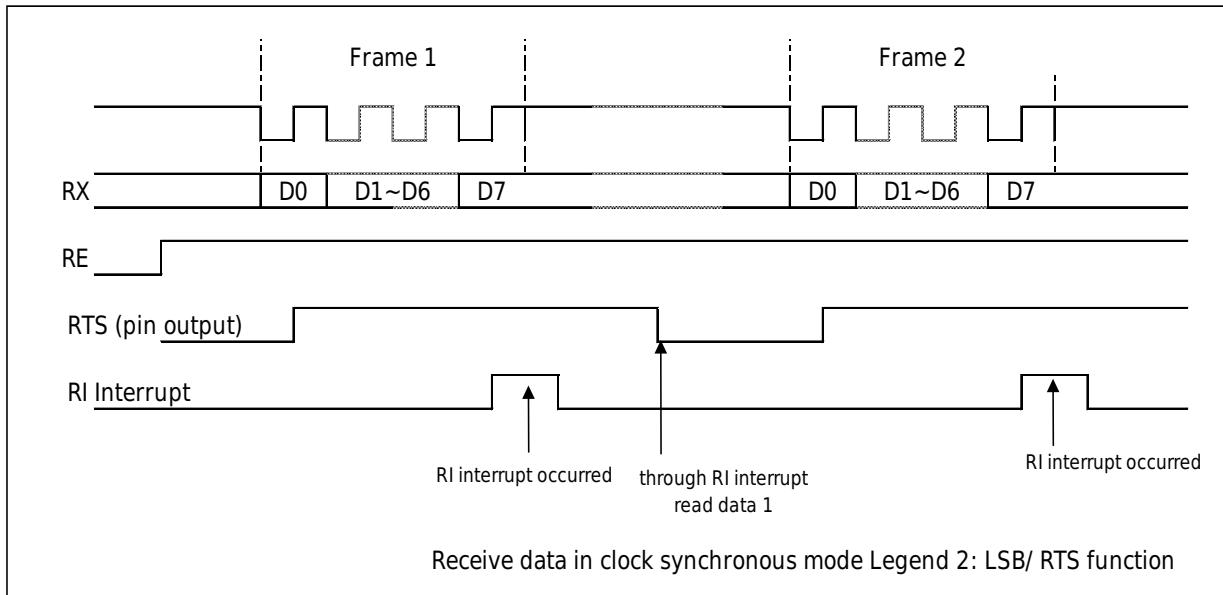


Figure 25-22 Diagram of Clock Synchronization Mode Receive Data 2

Error Handling

An overflow error (USARTn_SR.ORE) is received while receiving data in clock synchronization mode. Receiving errors can no longer be received and sent. You can restart the data transfer by clearing the error flag by writing the corresponding zeroing register.

The overflow 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 overflow error occurs, RI interrupt does not occur.

Receiver Interrupt

The clock synchronization mode receiver supports two types of interrupts, receive data register full interrupt RI and receive error interrupt REI.

RIE = 1, RI interrupt occurs when data is transferred from the receiving shift register to the receiving data register.

RIE = 1, REI interrupt occurs when an error (overflow error) occurs in the received data.

25.4.4.6 Send And Receive Data At The Same Time

The USART clock synchronization mode supports full-duplex operation while sending 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.

25.4.4.7 Clock Synchronization Mode Interrupt and Events

Table 25-7 Clock Synchronization Mode Interrupt/Event Table

Interrupt name	Marking	Enable bit (interrupt only)	Flag bit	Can be used as an event source
Reception error interrupt	REI	RIE	ORE	No
Received data register full interrupt	RI	RIE	RXNE	Yes
Send data register empty interrupt	TI	TXEIE	TXE	Yes
Send complete interrupt	TCI	TCIE	TC	No

25.4.5 Digital Filtering Function

USARTn_CR1.When 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 3/16 (USARTn _ CR1.OVER8 = 0 or 3/8 width (USARTn _ CR1.OVER8 = 1) noise.

If the clock of the digital filter stops before it starts, the digital filter continues to work from the time the clock stops.

USARTn _ CR.TE = 0 and USARTn _ CR.RE = 0 reset the Flip-Flop state inside the digital filter to 1.

25.5 Register Description

USART1_BASE_ADDR:0x4001_D000

USART2_BASE_ADDR:0x4001_D400

USART3_BASE_ADDR:0x4002_1000

USART4_BASE_ADDR:0x4002_1400

Table 25-8 List of USART Registers

Register name	Offset address	Reset value
Status register (USART _ SR)	0x00	0x0000_00C0
Data register (USART _ DR)	0x04	0x0000_01FF
Baud rate register (USART _ BRR)	0x08	0x0000_FFFF
Control register1 (USART _ CR1)	0x0C	0x8000_0000
Control register2 (USART _ CR2)	0x10	0x0000_0000
Control register3 (USART _ CR3)	0x14	0x0000_0000
Pre-divider register (USART _ PR)	0x18	0x0000_0000

25.5.1 USART Status Register (USART_SR)

Offset address: 0x00

Reset value: 0x0000_00C0

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

Bit	Marking	Place name	Function	Read and write
b31~b17	Reserved	-	Read as "0", write as "0"	R
b16	MPB	Multiprocessor position	Multi-processor bit flag 0: Current received data is communication data 1: Current received data is ID Note: MPB bits are only valid in multiprocessor mode	R
b15~b9	Reserved	-	Read as "0", write as "0"	R
b8	RTOF	UART receive TIMEOUT Flag bit	UART receive TIMEOUT Flag bit 0: no UART receive TIMEOUT 1: Occurrence UART receive TIMEOUT RTOF setting condition • No new received data is detected after the set time has elapsed since the STOP bit of the last frame data was detected RTOF zeroing condition • Clear register CR1.CRTOF bit write Note: RTOF is set to 1 by hardware, and it is set to 1 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. RTOF is a valid value only when USART_CR1.RTOE=1. Please ignore this bit in other cases. When using USART to receive Timeout function, before setting RTOE=1, software needs to clear this bit.	R
b7	TXE	Send data register empty	Send data register null flag TXE bit is valid in USART and clock synchronous mode. 0: Data not transmitted to shift register, send data register is not empty 1: Data transfer to shift register, send data register empty Note: TXE bits are set by hardware 1 and clear 0, TXE is cleared by hardware when data is not transmitted to shift register, TXE is cleared by hardware when data is transmitted to shift register 1	R

Bit	Marking	Place name	Function	Read and write
b6	TC	Send completion mark	<p>Send completion flag bit 0: Send data 1: Completion of data transmission UART mode, Clock synchronization mode TC setting condition <ul style="list-style-type: none"> • TE = 0 Send Disabled • When the last bit of a frame of data is sent, the value of the sent data register is not updated TC zeroing condition • When TE = 1, write the sending data to the sending data register </p>	R
b5	RXNE	Received data register is not empty	<p>Smart Card Pattern TC setting condition <ul style="list-style-type: none"> • TE = 0 Send Disabled • After a certain time has elapsed after the last 1 byte of data is sent, FE=0 and the value of the send data register is not updated. The specific timing of TC setting is: 2.5 digits after the check digit is sent TC zeroing condition <ul style="list-style-type: none"> • When TE = 1, write the sending data to the sending data register <p>Note: When the TE bit is changed from 0 to 1, TC is kept to 1.</p> </p>	R
b4	Reserved	-	Received data register is not null 0: No data received 1: Ready to read received data <p>Note: The RXNE bit is set by hardware 1 and cleared by 0, the RXNE bit is set by hardware 1 when ready to read the received data, and the RXNE bit is cleared by hardware 0 after reading the received data.</p>	R
b3	ORE	Reception overflow error	<p>Read as "0", write as "0"</p> <p>Reception overflow error flag bit 0: No Receive Overflow Error 1: Reception overflow error ORE setting condition <ul style="list-style-type: none"> • A new frame of data was received when the received data register was not read ORE zeroing condition <ul style="list-style-type: none"> • Zeroing registerCR1 bit write 1 <p>Note: Re = 0 does not reset ORE 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 send data in clock synchronization mode</p> </p>	R
b2	Reserved	-	Read as "0", write as "0"	R

Bit	Marking	Place name	Function	Read and write
b1	FE	Received frame error	<p>Received frame error flag bit 0: No received frame error 1: Receive frame error UART mode Fe setting 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 zeroing condition <ul style="list-style-type: none"> • Zeroing registerCR1 bit write 1 <p>Note: In UART mode, Re = 0 does not reset the FE position Data received at FE = 1 will be retained but Rlinterrupt will not occur and cannot continue to receive data after FE = 1</p> <p>Smart Card Pattern Fe setting condition <ul style="list-style-type: none"> • Sampled low error signal flag FE zeroing condition <ul style="list-style-type: none"> • Zeroing registerCR1 bit write 1 <p>Note: In smart card mode, RE=0 does not reset the FE bit</p> </p></p>	R
b0	PE	Received data verification	<p>Received data check error flag 0: No received data check error 1: Received data check error PE setting condition <ul style="list-style-type: none"> • When a parity error occurs in the received data PE zeroing condition <ul style="list-style-type: none"> • Zeroing registerCR1.CPE bit write 1 <p>Note: Re = 0 does not reset the PE position Data received at PE = 1 will be retained but Rlinterrupt will not occur and cannot continue to receive data after PE = 1</p> </p>	R

25.5.2 USART Data Register (USART_DR)

Offset address: 0x04

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16						
-	-	-	-	-	-	-	-	RDR[8:0]													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0						
-	-	-	-	-	-	-	MPID	TDR[8:0]													

Bit	Marking	Place name	Function	Read and write
b31~b25	Reserved	-	Read as "0", write as "0"	R
b24~b16	RDR[8:0]	Receive data register	Receive data register Note: The highest 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
b9	MPID	Multiprocessor mode ID bit	Send communication data or select bits for sending IDs in multiprocessor mode 0: Send data 1: Sending ID Note: MPID bits are only valid in multiprocessor mode and other modes must be set to reset	R/W
b8~b0	TDR[8:0]	Send data	Send data Note: The highest TDR [8] is only valid in UART mode and the data length is set to 9 bits	R/W

25.5.3 USART Bit Rate Register (USART_BRR)

Offset address: 0x08

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DIV_Integer [7:0]										-	DIV_Fraction[6:0]				

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R
b15~b8	DIV_Integer[7:0]	Integer frequency division register Note: IV_Integer [7: 0] can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	Integer frequency division register Note: IV_Integer [7: 0] can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b7	Reserved	-	Read as "0", write as "0"	R
b6~b0	DIV_Fraction[6:0]	less division register Note:[6: 0] only be set at TE = 0 & RE = 0 (send/receive disabled), and Only Set value valid at FBME = 1	less division register Note:[6: 0] only be set at TE = 0 & RE = 0 (send/receive disabled), and Only Set value valid at FBME = 1	R/W

Table 25-9 Baud rate calculation formula (invalid fractional baud rate FBME=0)

Pattern	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	$B = \frac{C}{4 \times (DIV_Integer + 1)}$	-
Smart Card Pattern	$B = \frac{C}{2 \times BCN \times (DIV_Integer + 1)}$	$E(%) = \left\{ \frac{C}{2 \times BCN \times (DIV_Integer + 1) \times B} - 1 \right\} \times 100$

B: Baud rate unit: Mbps

C: PR.PSC [1: 0] bit set clock unit: MHz

BCN: CR3.BCN register setting value

Table 25-10 Baud rate calculation formula valid fractional baud rate 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	$B = \frac{C \times (128 + DIV_Fraction)}{4 \times (DIV_Integer + 1) \times 256}$	
Smart Card Pattern	$B = \frac{C \times (128 + DIV_Fraction)}{2 \times BCN \times (DIV_Integer + 1) \times 256}$	$E(%) = \left\{ \frac{C \times (128 + DIV_Fraction)}{2 \times BCN \times (DIV_Integer + 1) \times 256 \times B} - 1 \right\} \times 100$

B: Baud rate unit: Mbps

C: PR.PSC [1: 0] bit set clock unit: MHz

BCN: CR3.BCN register setting value

25.5.4 USART Control Register 1 (USART_CR1)

Offset address: 0x0C

Reset value: 0x8000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SBS	NFE	FBME	ML	-	-	-	MS	-	-	-	CRTOF	CORE	-	CFE	CPE
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OVER8	-	-	M	-	PCE	PS	-	TXEIE	TCIE	RIE	SLME	TE	RE	RTOIE	RTOE

Bit	Marking	Place name	Function	Read and write
b31	SBS	Location of UART Mode Receiving Data Begin Detection	Location of start bit detection mode when USART mode receives data 0: Start bit detection mode is RX pin low level 1: Starting position detection method is RX pin falling edge Note: In non-USART mode, the SBS bit must remain reset SBS bits can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b30	NFE	Digital filter enable bit	Digital filter enable bit 0: Disable digital filtering 1: Enable data filtering Note: NFE bits must remain reset in non-USART mode NFE bits can only be set when TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b29	FBME	Fractional baud rate function enable	Fractional baud rate function enable bit 0: Prohibited 1: Enable Note: FBME bits can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b28	ML	MSB/LSB selection bit	MSB/LSB mode selection bit in USART mode/clock synchronization smart card mode 0: LSB mode 1: MSB mode Note: ML bits can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b27~b25	Reserved	-	Read as "0", write as "0"	R
b24	MS	Communication mode selection bit	Communication mode selection bit 0: USART mode 1: Clock synchronization mode Note: The MS bit can only be set when TE=0&RE=0 (send/receive disabled), the smart card mode MS needs to write the reset value	R/W
b23~b21	Reserved	-	Read as "0", write as "0"	R

Bit	Marking	Place name	Function	Read and write
b20	CRTOF	RTOF zeroing	RTOF zeroing 0: Not Zero RTOF Mark 1: Clear RTOF logo Note: CRTOF bit write 1 clears RTOF flag and returns 0 when read	R/W
b19	CORE	ORE flag zeroing	ORE flag zeroing 0: Not zeroed ORE logo 1: Clear ORE logo Note: The CORE bit write 1 clears the ORE flag and returns 0 when read	R/W
b18	Reserved	-	Read as "0", write as "0"	R
b17	CFE	Fe mark zeroing	Fe mark zeroing 0: Unzeroed FE logo 1: Reset FE logo Note: CFE bit write 1 clears FE flag and returns 0 when read	R/W
b16	CPE	PE mark zeroing	PE mark zeroing 0: Not Zero PE Mark 1: Clear PE logo Note: CPE bit write 1 clears PE flag and returns 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: OVER8 bits must remain reset in non-UART mode OVER8 bits can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b14~b13	Reserved	-	Read as "0", write as "0"	R/W
b12	M	Data length setting	In UART mode, transmit/receive data length setting 0: 8 bits 1: 9 bits Note: M bits must remain reset in non-UART mode M bits can only be set when TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b11	Reserved	-	Read as "0", write as "0"	R
b10	PCE	Check enable bit	Parity enable bit in UART mode 0: No check 1: Verification Note: The PCE bit must be 1 in smart card mode, and the PCE bit must maintain the reset value in clock synchronous mode PCE bits can only be set when TE=0&RE=0 (Send/Receive Disabled)	R/W
b9	PS	Check bit	Parity selection bit in UART mode 0: Even check 1: Parity Note: PS bits can only be set at TE = 0 & RE = 0 (send/receive disabled), and PS bits are valid at PCE = 1	R/W
b8	Reserved	-	Read as "0", write as "0"	R

Bit	Marking	Place name	Function	Read and write
b7	TXEIE	Send data register empty interrupt enable bit	Send data register empty interrupt enable bit 0: Tlinterrupt request is invalid, Tlinterrupt does not occur 1: Tlinterrupt request is valid, Tlinterrupt occurs Note: TXEIE bit and TE bit should be written to 1	R/W
b6	TCIE	Send complete interrupt	Search for energy bit during transmission 0: TCInterrupt request is invalid, TCInterrupt does not occur 1: TCInterrupt request enabled, TCInterrupt occurs	R/W
b5	RIE	Receive interrupt	Receive interrupt 0: Invalid receive interrupt request, RI and RElinterrupt do not occur 1: Receive interrupt request is valid, RI and RElinterrupt 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 whose MPB bit is 0 will not be read from the receiving shift register to the receiving 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 UART multiprocessor mode and must remain reset in other modes.	R/W
b3	TE	Transmitter energy position	Transmitter energy position 0: Transmitter prohibited 1: Transmitter enable Note: TE bits in clock synchronization mode can only be written 1 at TE = 0 & RE = 0 (send/receive disabled).	R/W
b2	RE	Receiver enable position	Receiver enable position 0: Receiver disabled 1: Receiver enable Note: In clock synchronization mode RE bits can only be written at TE = 0 & RE = 0 (send/receive disabled) 1	R/W
b1	RTOIE	UART TIMEOUT Interrupt enable bit	UART TIMEOUT Interrupt enable bit 0: UART TIMEOUT interrupt request is invalid, RTOI interrupt does not occur 1: UART TIMEOUT interrupt request is valid, RTOI interrupt occurs Note: RTOIE needs to be set to 1 at the same time as RTOE, or set to 1 after RTOE=1	R/W
b0	RTOE	UART TIMEOUT Function enable bit	UART TIMEOUT Function enable bit 0: UART TIMEOUT Function Prohibition 1: UART TIMEOUT Function enable Note: Before RTOE=1 is set to 1, it is necessary to clear the USARTn.SR.RTOF flag and stop the Timer0 counting function of the corresponding channel	R/W

25.5.5 USART Control Register 2 (USART_CR2)

Offset address: 0x10

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b14	Reserved	-	Read as "0", write as "0"	R
b13	STOP	Stop positioning	Stop bit length setting in USART mode 0: 1 stop bit 1: 2 stop bit Note: The STOP bit must remain reset in non-USART mode The STOP bit can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b12~b11	CLKC[1:0]	Clock control bit	UART mode 00b: Clock generated by timer for internal baud rate generator, clock not output to USARTn_CK pin, USARTn_CK pin can be used as ordinary IO 01b: Clock generated by timer for internal baud rate generator, clock output to USARTn_CK pin, output clock frequency and baud rate same 10b or 11b: Timer is external input clock, input clock frequency is 16 times (OVER8 = 0) or 8 times (OVER8 = 1) higher than baud rate Clock synchronization mode 00b or 01b: Clock generated by timer for internal baud rate generator, output to USARTn_CK pin 10b or 11b: Timer is an external input clock with the same frequency and baud rate Smart Card Pattern 00b: Clock generated by timer for internal baud rate generator, clock not output to USARTn_CK pin, USARTn_CK pin can be used as ordinary IO 01b: Clock generated by timer for internal baud rate generator, Clock output to USARTn_CK pin 10b or 11b: setting prohibited Note: CLKC [1: 0] bits can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b10~b1	Reserved	-	Read as "0", write as "0"	R
b0	MPE	Multi-processor function	In USART mode, the multiprocessor function enables the bit 0: Prohibited 1: Enable Note: MPE bits must remain reset in non-USART mode MP bits can only be set when TE = 0 & RE = 0 (Send/Receive Disabled)	R/W

25.5.6 USART Control Register 3 (USART_CR3)

Offset address: 0x14

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	BCN[2:0]			-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b15	b14
-	-	-	-	-	-	CTSE	-	-	-	SCE N	-	-	-	-	-

Bit	Marking	Place name	Function	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R
b23~b21	BCN[2:0]	base clock count	In smart card mode, sets the number of base clocks during a one-bit data transfer BCN[2: 0] setting Value the basic number of clocks during one-bit data transmission 000b 32 001b 64 010b Prohibitions 011b 128 100b Prohibitions 101b 256 110b 372 111b Prohibitions Note: The BCN [2: 0] bit must remain reset in non-Smart Card mode BCN [2: 0] bits can only be set at TE=0&RE=0 (Send/Receive Disabled)	R/W
b20~b10	Reserved	-	Read as "0", write as "0"	R
b9	CTSE	CTS function	CTS function 0: RTS function 1: CTS function Note: CTSE bits can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b8~b6	Reserved	-	Read as "0", write as "0"	R
b5	SCEN	Smart Card Pattern Enable bit	Smart Card Pattern Enable bit 0: Prohibited Smart Card Pattern 1: Enable Smart Card Pattern Note: The SCEN bit must remain reset in non-Smart Card mode SCEN bits can only be set when TE = 0 & RE = 0 (Send/Receive Disabled)	R/W
b4~b0	Reserved	-	Read as "0", write as "0"	R

25.5.7 USART Prescaler Register (USART_PR)

Offset address: 0x18

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b2	Reserved	-	Read as "0", write as "0"	R
b1~b0	PSC[1:0]	Pre-divider value	Pre-divider divider selection bit when internal timer 00: PCLK 01: PCLK/4 10: PCLK/16 11: PCLK/64 Note: PSC [1: 0] bits can only be set at TE = 0 & RE = 0 (Send/Receive Disabled)	R/W

25.6 Precautions for Use

25.6.1 UART Considerations

Transmitter

UART mode transmitter send disable (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 sender 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 erasing the error flag, and the receiving error will occur again.

25.6.2 Clock Synchronization Mode Considerations

- 1) When using an external input clock to send 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 is entered.
- 2) When data is sent consecutively, the next frame of data needs to be updated before the last bit of the current frame is sent.

25.6.3 Other Considerations

- 1) In order to prevent the TX communication line Hi-Z from being sent in the Disable state, the following methods can be used:
 - Cable pull-up
 - At the end of sending data, before USARTn_CR1.TE=0, set the TX pin as a normal IO output
 - Before sending data, after USARTn_CR1.TE=1, set IO to TX function

26 Integrated Circuit Bus (I2C)

26.1 Introduction

I2C (integrated circuit bus) used as an interface between the microcontroller and the I2C serial bus. Provide multi-master mode function, which can control the protocol and arbitration of all I2C buses. Standard mode and fast mode are supported. SMBus is also supported.

Main Features of I2C:

- 1) I2C bus mode and SMBUS bus mode are optional. Host mode and slave mode are optional. Automatically ensure various preparation times, hold times, and bus idle times corresponding to the transfer rate.
- 2) The standard mode is up to 100 Kbps, and the fast mode is up to 400 Kbps.
- 3) 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.
- 4) Two slave mode addresses can be set. 7-bit address format and 10-bit address format can be set simultaneously. Broadcast call address, SMBus host address, SMBus device default address, SMBus alarm address can be detected.
- 5) Answer bits can be automatically determined when sent. Answer bits can be sent automatically upon reception.
- 6) Handshake function.
- 7) Arbitration function.
- 8) Timeout function, can detect SCL clock stop for a long time.
- 9) SCL input and SDA input have built-in digital filters that are programmable to filter.
- 10) Communication error, receive data full, send data empty, one frame send end, address match unanimously interrupt.

26.2 I2C System Block Diagram

26.2.1 System Block Diagram

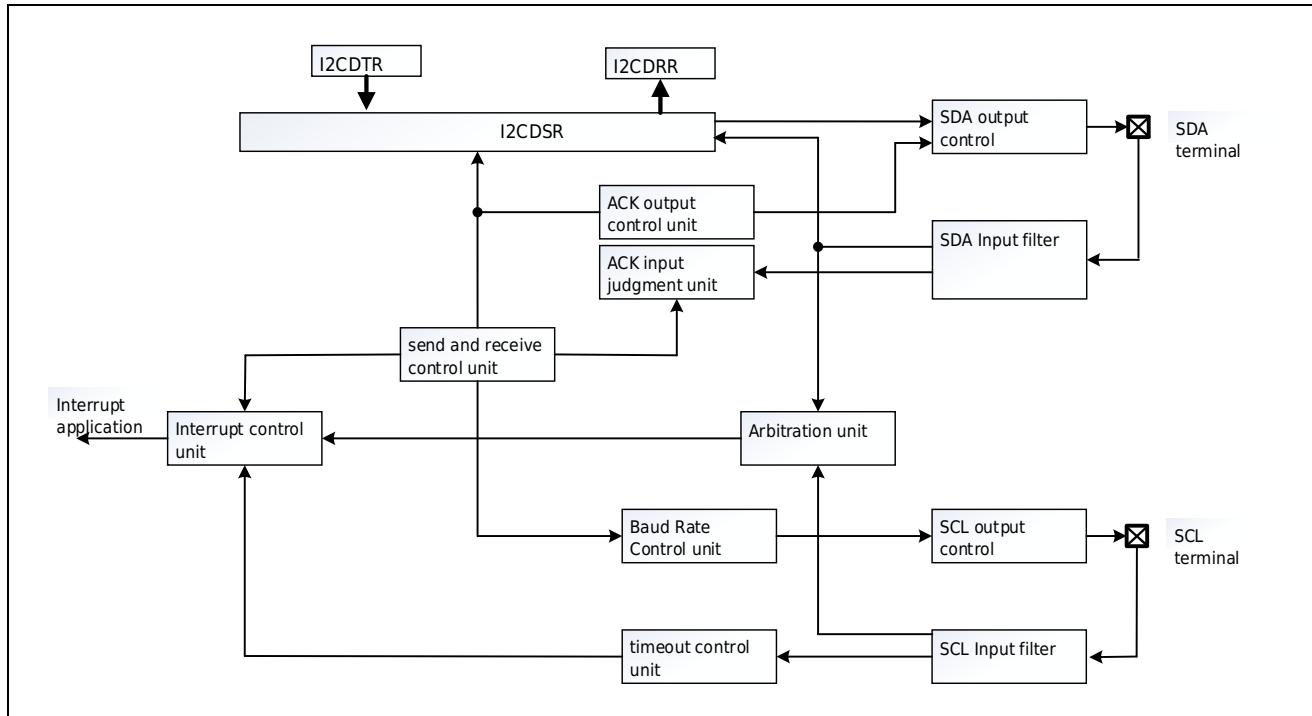


Figure 26-1 I2C System Block Diagram

26.2.2 Structural Diagram

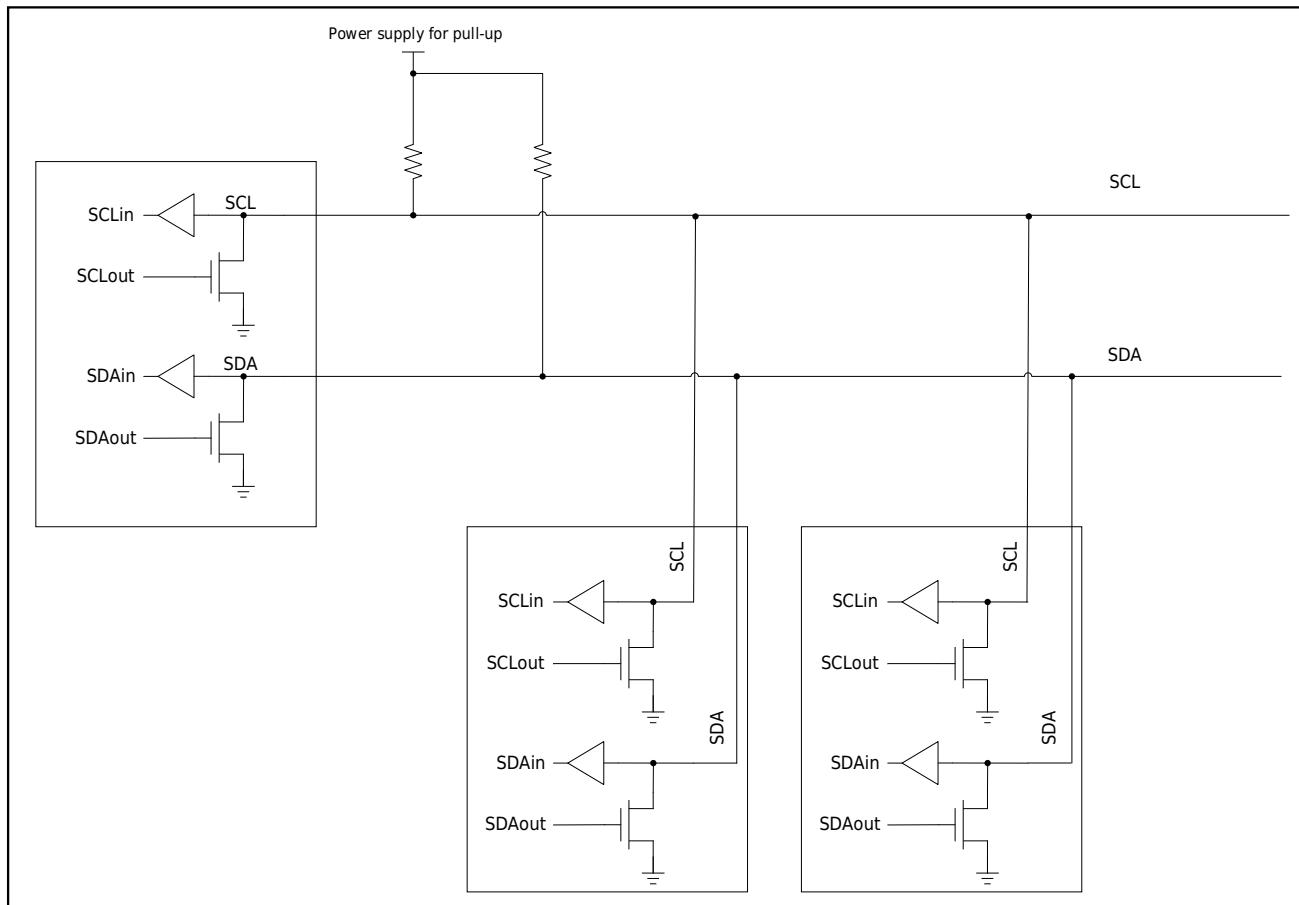


Figure 26-2 Example of I2C Bus Structure

Table 26-1 Input/output Pin

Pin Name	Input/Output	Function
SCL	Input/Output	Input/output pin of serial clock
SDA	Input/Output	Input/output pin of serial clock

SCL/SDA input level is CMOS level when I2C bus is selected. When SMBus is selected, the SCL/SDA input level is TTL.

26.3 Action Description

This section provides a description of the functions of the I²C module.

26.3.1 I²C Protocol

I²C bus consists of a clock line (SCL) and a data line (SDA). All connection devices must be drain open output. SCL, external pull-up resistance of SDA line. Resistance depends on 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 a Sequence Diagram of the I²C bus.

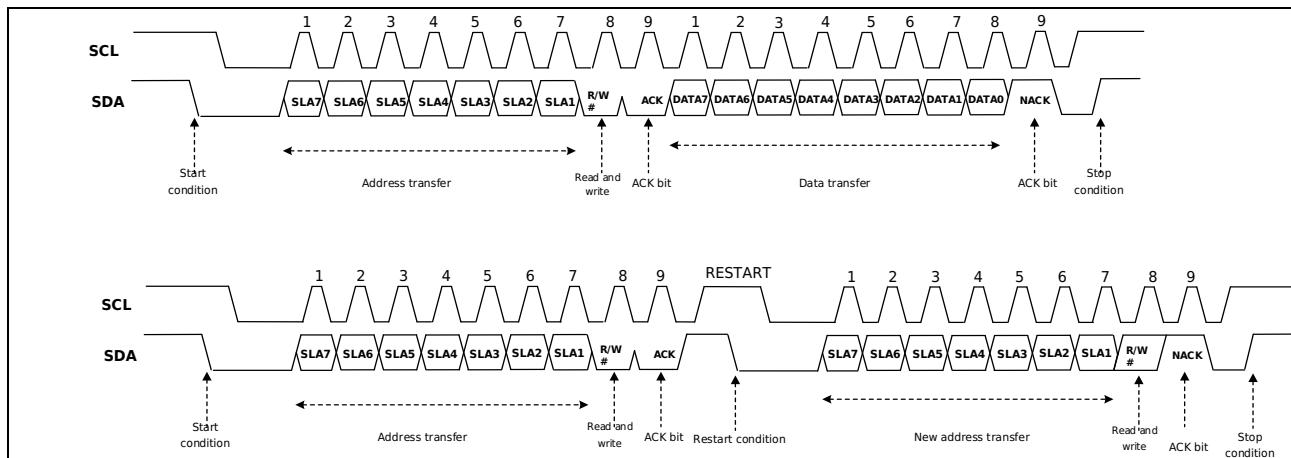


Figure 26-3 Sequence Diagram of I²C bus Timing Diagram.

26.3.1.1 Start Condition

When the host 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.

If the START position is set to "1" in the I²C_SR.BUSY flag of "0" (bus is idle), the launch condition is issued. If a start condition is detected, the I²C_SR.BUSY flag and I²C_SR.STARTF flag are automatically set to "1" and the START position is automatically set to "0". At this point, if the status of the SDA signal sent in the START bit "1" is the same as that of the SDA line, and the start condition is detected, it is considered that the start condition is correctly issued by the START bit, and the I²C_SR.MSL bit and I²C_SR.TRA bit are automatically set to "1" and then changed to the master send mode. In addition, I²C_SR.TEMPTYF automatically changes to '1' because the TRA bit is '1'. Next write the slave address to the I²C_DTR register and send the address.

26.3.1.2 Address Transfer

The frame after the start condition or restart condition is an address frame that specifies the object address for host communication. The specified slave is always valid until the stop condition is sent.

The top 7 bits of the address frame are the slave addresses. The eighth bit of the address frame determines the direction of the data frame.

- 1) The 7-bit addressing pattern is shown in the following figure [7-bit address format]

Host send mode, host send address frame 8th bit is 0

Host receive mode, host send address frame 8th bit is 1

- 2) The 10-bit addressing pattern is shown in the following figure [10-bit address format]

Host send mode, host first frame send header sequence (11110XX0, where XX represents the top two digits of the 10-bit address), then the second frame send the lower octet slave address.

Host accept mode, host first frame send header sequence(11110XX0, where XX represents the top two digits of the 10-bit address), and then the second frame send the lower octet slave address. Next, a restart condition is sent, and then a frame header sequence is sent (11110XX1, where XX represents the top two digits of the 10-bit address).

7-bit address format

S	SLA(7-bit)	R/W #	ACK/ NACK	DATA(8-bit)	ACK		DATA(8-bit)	ACK/ NACK	P
---	------------	----------	--------------	-------------	-----	--	-------------	--------------	---

10-bit address format

S	11110b+SLA (2-bit)	W#	ACK	SLA(8-bit)	ACK	DATA(8-bit)	ACK		DATA(8-bit)	ACK/NACK	P
---	-----------------------	----	-----	------------	-----	-------------	-----	--	-------------	----------	---

S	11110b+SLA (2-bit)	W#	ACK	SLA(8-bit)	ACK	Sr	11110b+SLA (2-bit)	R	ACK	DATA(8-bit)	ACK		DATA(8-bit)	ACK/NACK	P
---	-----------------------	----	-----	------------	-----	----	-----------------------	---	-----	-------------	-----	--	-------------	----------	---

Figure 26-4 I²C Data format of I²C bus

S: Indicates the start condition.

SLA: Indicates the slave address.

R/W #: Indicates the direction of transmission and reception. Send data from slave to host when R/W # is "1"; When R/W # is "0", the data is sent from the host to the slave.

Sr: Restarting condition.

DATA: Data sent and received

P: Stop condition.

26.3.1.3 Data transfer

After address matching is consistent, the host on the bus transmits data frame by frame according to the direction defined by R/W.

The data transmitted after all address frames are treated as data frames. Even a lower 8-bit address in a 10-bit address format is considered a data frame.

The length of the data frame is 8 bits. The low-level SDA of the SCL changes, the high-level SDA of the SCL remains, and a bit of data is sent per clock cycle. The 9th clock after the data frame is an answer bit and is a handshake signal transmitted to the receiver.

If data is received from the slave computer on the bus and does not respond to the host in the ninth clock cycle, the slave computer must send NACK. If the host receives data on the bus, NACK is sent in the 9th cycle, NACK is received from the slave, and data is stopped from the slave.

Whether the host or the slave sends NACK, the data transfer interrupts. The host can do any of the following:

- 1) Transmit stop conditional release bus
- 2) Send the restart condition to start a new communication.

Host sends data

In host transmit mode, the host outputs SCL clock and transmit data, receives data from the host and returns an answer. The sequence of host sending data is shown in the following figure.

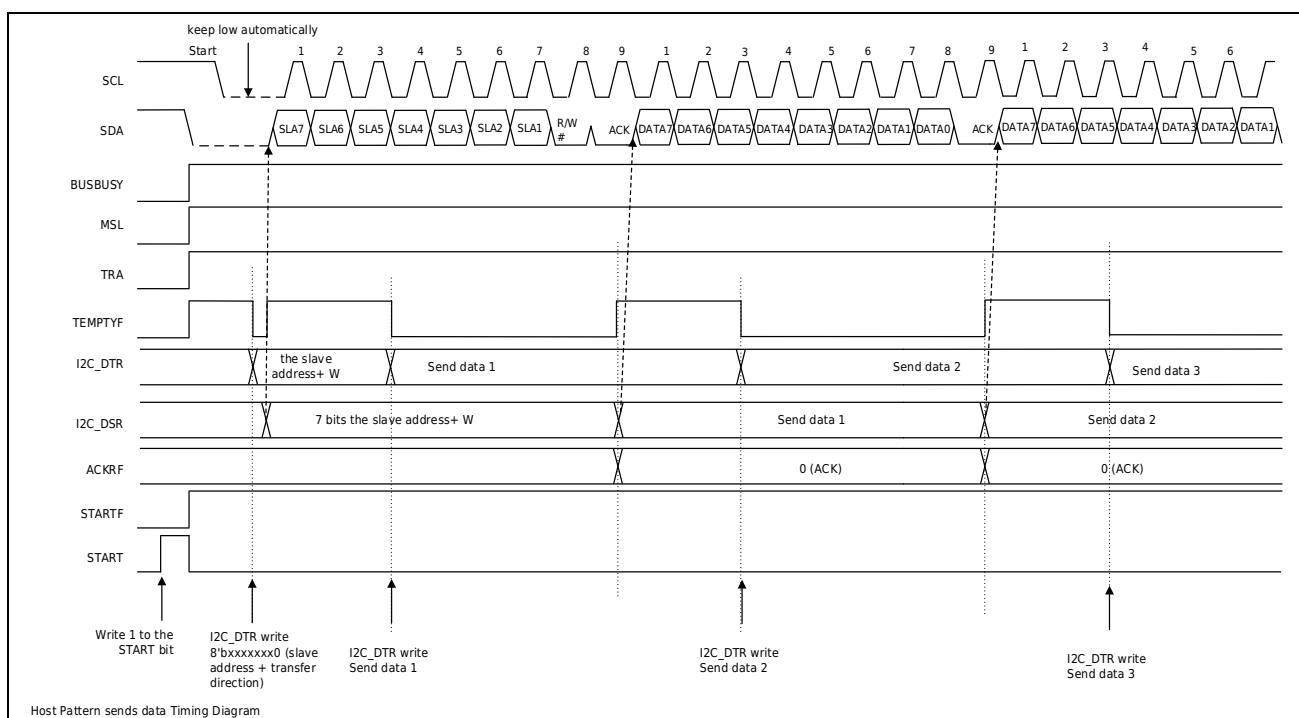


Figure 26-5 Sequence Diagram of Host Sending Data in 7-bit Address Format (Example)

Host Receiving Data

In host receiving mode, the host outputs an SCL clock, receives slave data, and returns an answer. The running sequence of receiving data by the host is shown in the following figure.

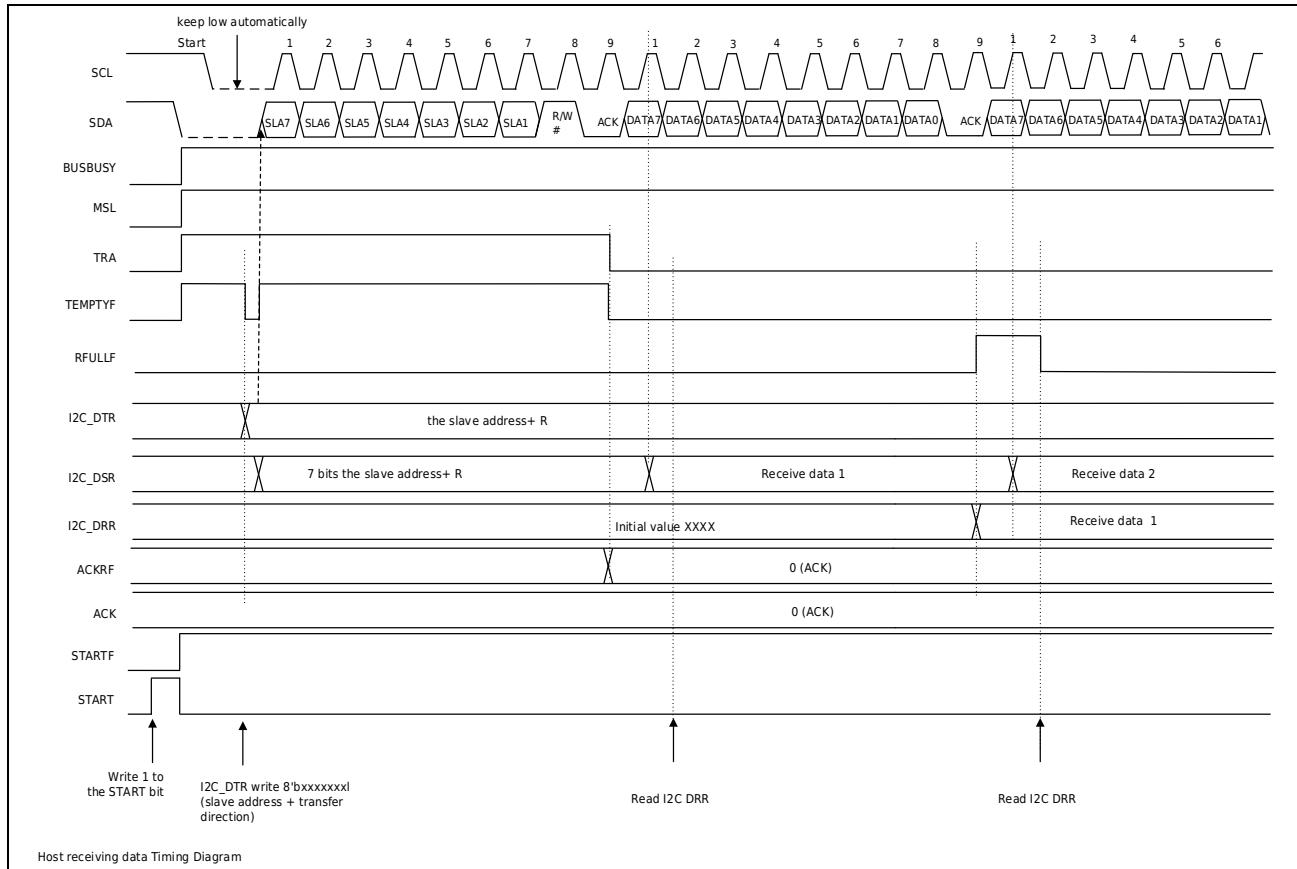


Figure 26-6 Sequence Diagram of Received Data for Host in 7-bit Address Format (Example)

Sending data from a slave computer

In Slave Send mode, the SCL clock from the host is received. This product sends data for the slave and receives a reply from the host. The sequence of data sent from the machine is shown in the following figure.

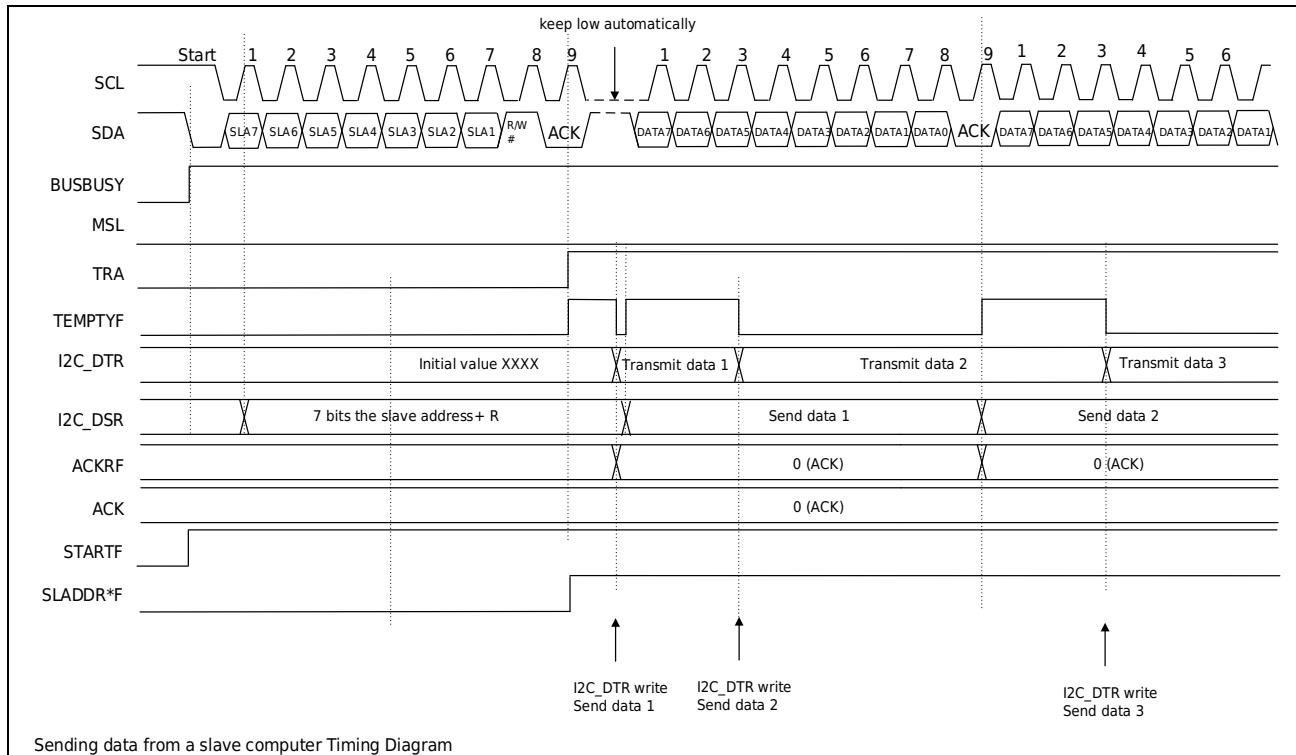


Figure 26-7 Sequence Diagram of Slave Send Mode in 7-bit Address Format (Example)

Receive data from machine

In slave receiving mode, the SCL clock and data from the host are received, and the response is returned after receiving the data. The sequence of data received from the machine is shown in the following figure.

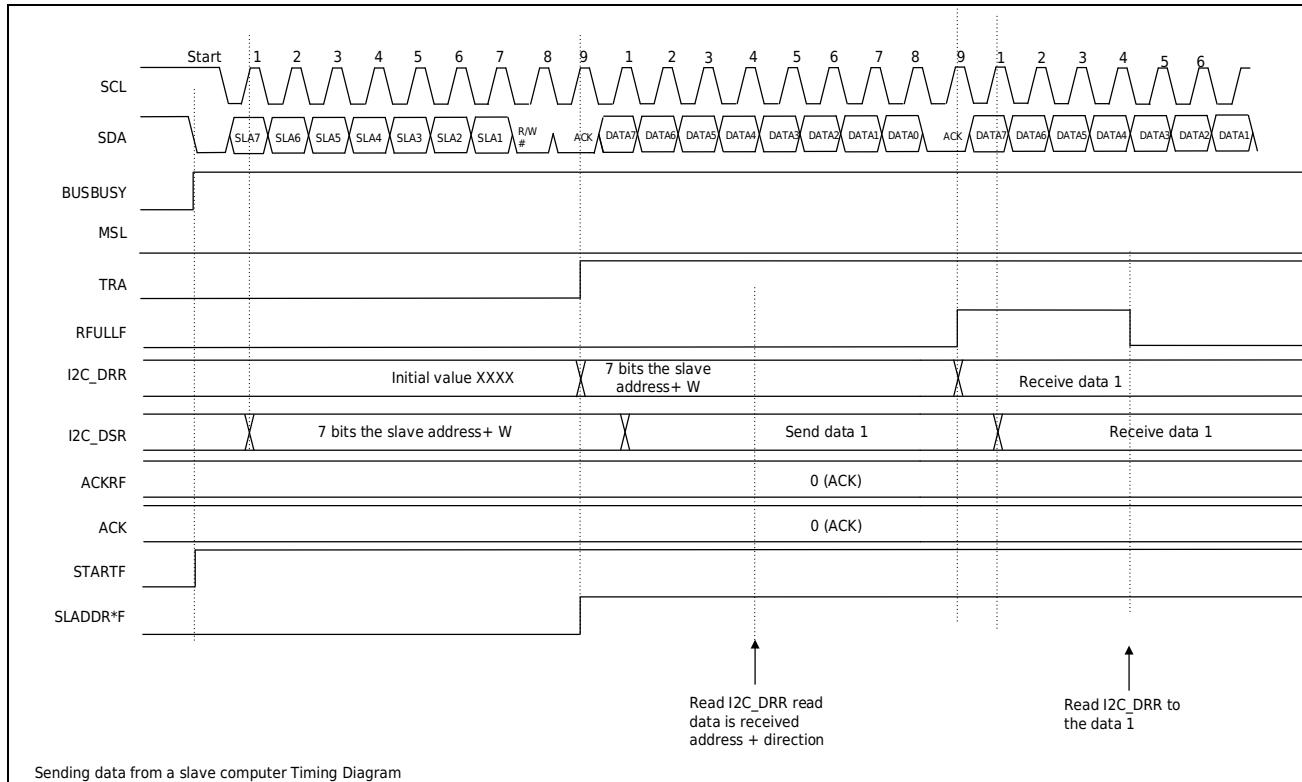


Figure 26-8 Sequence Diagram of 7-bit Address Format Slave Receive Mode (Example)

26.3.1.4 Stop Condition

Through I2C_CR1.The STOP bit release stop condition.

When the I2C_SR.BUSY flag is "1" (bus is busy) and the I2C_SR.MSL bit is "1" (host mode), the STOP bit is set to "1" to issue a stop condition.

26.3.1.5 Restart Condition

Through I2C_CR1.The RSTART bit generation restart condition.

The RSTART position "1" generates a restart condition when the I2C_SR.BUSY flag is "1" (busy bus) and the I2C_SR.MSL bit is "1" (host mode).

By restarting the condition, the host can switch transmit/receive modes without releasing BUS authority. Communication can also be established with another slave without releasing BUS authority.

26.3.1.6 SCL Clock Synchronization

When using the I2C bus in the multi-master mode, the SCL clock and SCL clock may conflict due to competition with other hosts. If the SCL clock conflicts, the host needs to synchronize with the SCL clock and synchronize the SCL clock bit by bit. When the rising edge of the SCL line is detected and the high level set by I2C _ CCR.SHIGHWregister is counted, if the SCL line is lowered due to the SCL clock output of other hosts, the increment count of the high level width is stopped when the falling edge of the SCL line is detected, and the increment count of the low level width set by I2C _ CCR.SLOWW is started while the SCL line is driven to low level, the low level drive of the SCL line is interrupted and the SCL line is released. At this time, if the low level width of the SCL clock of other hosts is greater than the low level width set by SLOWW, the low level width of the SCL clock is prolonged. When other hosts end the low-level output, release the SCL line and the SCL clock rises. Therefore, in case of SCL clock output conflict, the high level width of SCL clock synchronizes with short clock, and the low level width synchronizes with long clock.

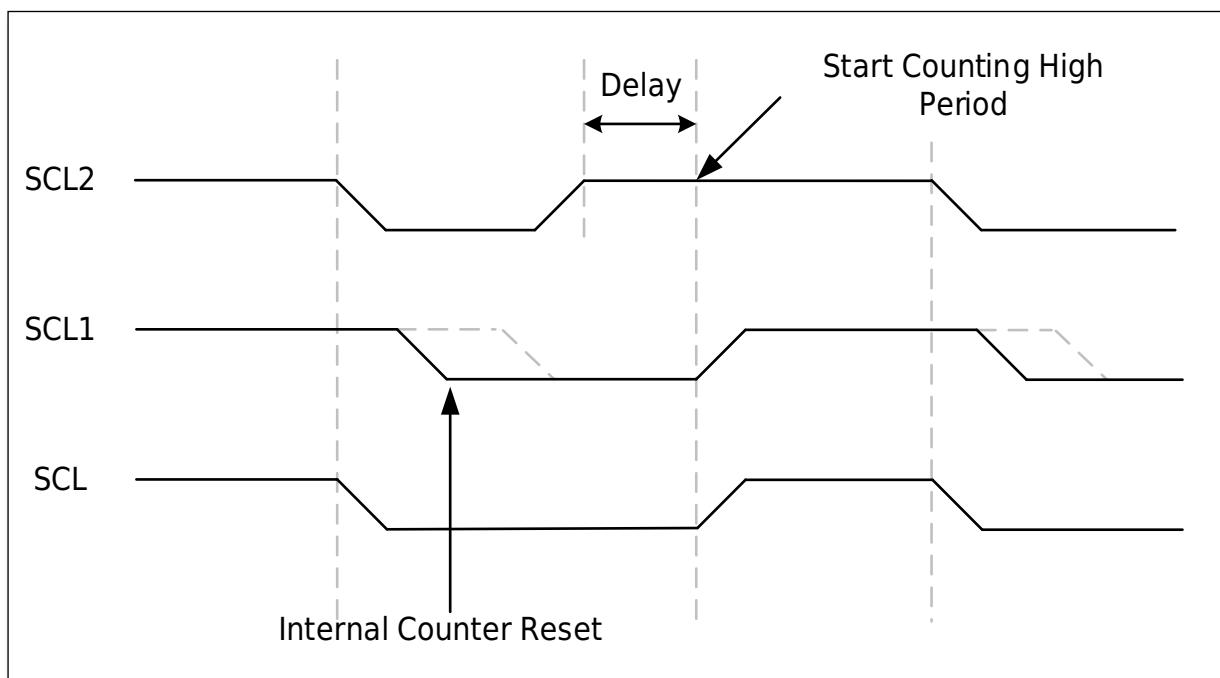


Figure 26-9 SSCL Synchronization Time Series

26.3.1.7 Arbitration

I2C bus is a real multi-host bus that allows multiple hosts to connect.

If two or more hosts try to control the bus at the same time, the SCL clock synchronization process determines the bus clock. The low 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 at high levels. When the SDA output sent is a high level output (the SDA pin is in a high impedance state) and the SDA line is detected to be at a low level, arbitration fails. The I2C _ SR.AROLF bit will be hardware "1". If a host arbitration fails, it is

immediately transferred to the slave receiving mode. At this point, if the slave addresses, including broadcast addresses, match, the slave mode continues to run.

26.3.1.8 Handshake

The handshake is realized by SCL clock synchronization mechanism during data transfer. After transmitting a frame of data (including ACK bits), the slave maintains the SCL clock line at low level. In this case, the low level of the SCL clock causes the host to enter the waiting state until the SCL line is released from the host.

[Slave Send Mode]

- 1) In transmit mode (I2C_SR.TRA bit = 1), if the shift register (I2C_DSRegister) is empty and the transmit data (I2CDTRegister) is not written, the low level of the SCL line is automatically maintained in the low level range of the 9th clock and the next clock, as shown in the following figure.

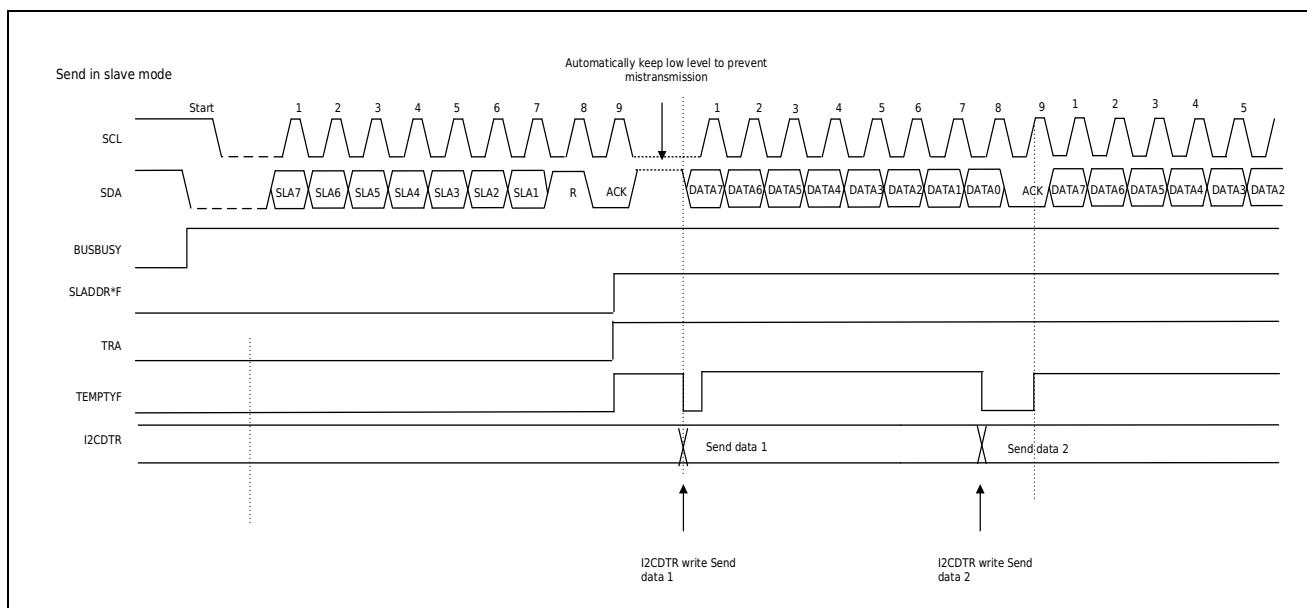
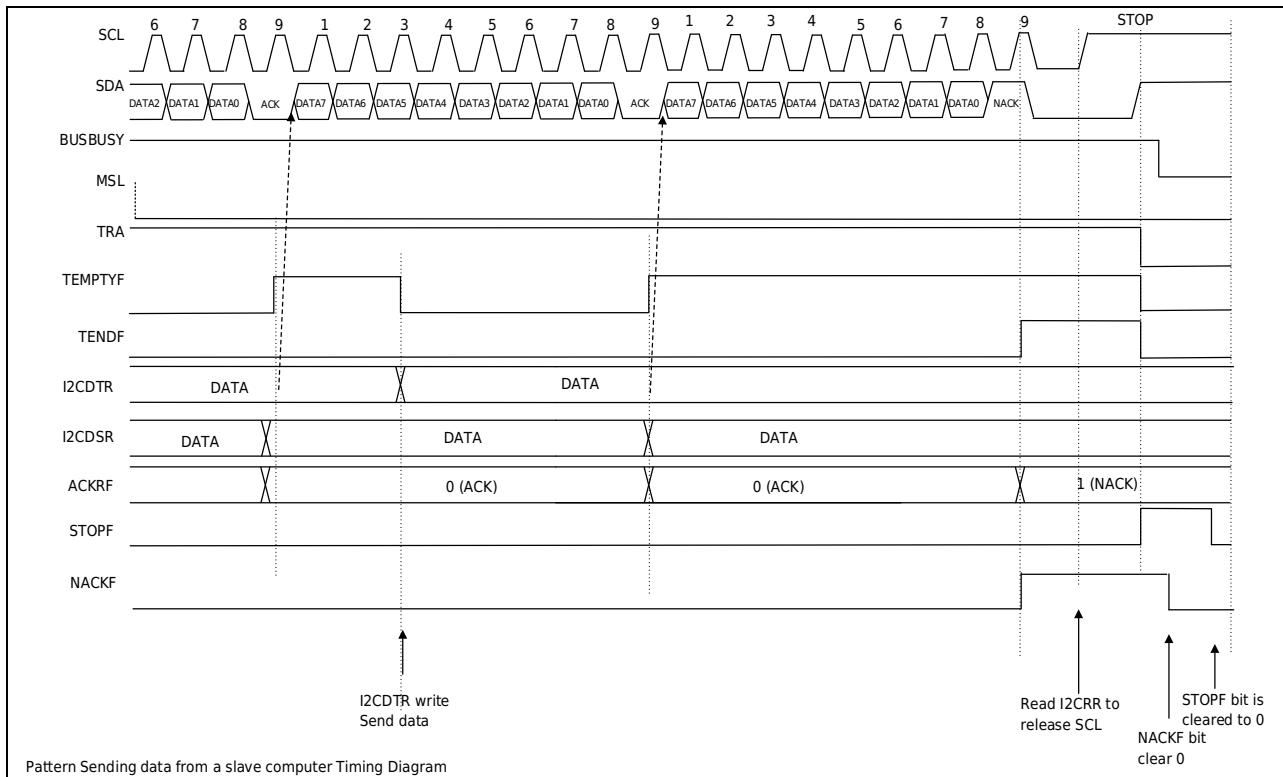


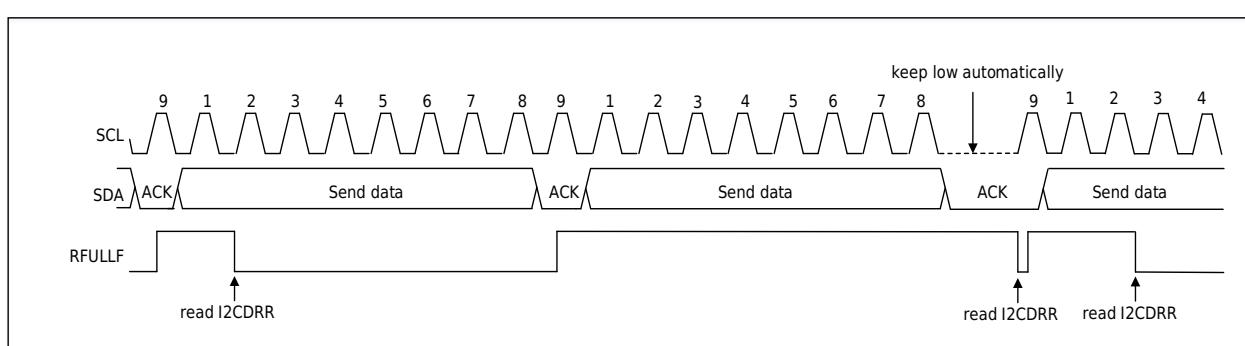
Figure 26-10 Sending Sequence Diagram of Slave (1)

- 2) After the I2C_SR.NACKF flag is changed to "1" or the last sent data is written to the I2C_DTRRegister, the I2C_SR.TEMPTYF flag is changed to "1". When the IC_SR.NACKF flag or TENDF flag is "1", keep the SCL line low after the ninth clock drops. At this point, the communication must be interrupted by reading I2C_DRRegister, thereby releasing the SCL line.



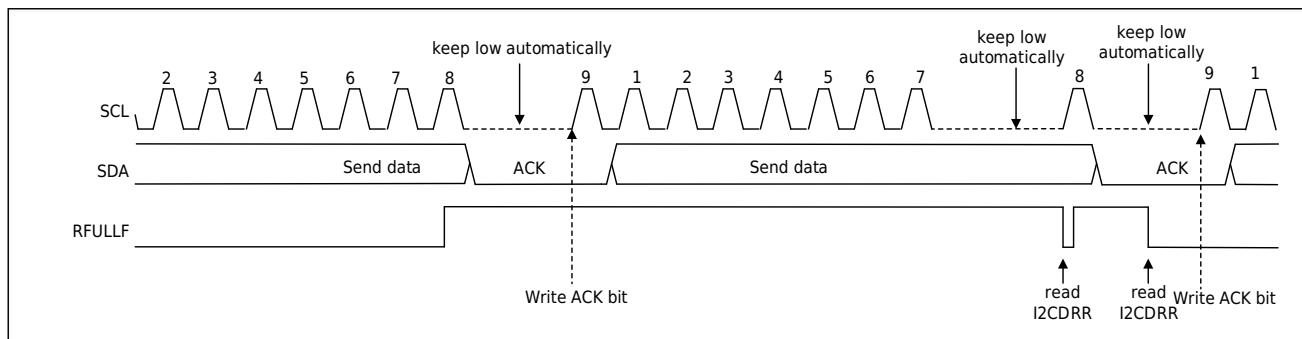
[Slave Receiving Mode]

If a response processing delay occurs in receiving mode (I2C_SR.TRA bit = 0) and when the received data is full (I2C_SR.RFULLF flag = 1) due to the delay of at least one transmission frame read received data (I2C_DR register), etc., the SCL line is automatically maintained between the 8th SCL and the 9th SCL clock, as shown in the following figure.



[Fast ACK/NACK]

In SMBUS communication, the data packet error code (PEC) of SMBUS is calculated or the received data is checked using the built-in CRC operator. During a PEC code check, the last byte sends an ACK or NACK based on matching. This requires the SCL to be kept low at the falling edge of the eighth clock of the last byte of the received SCL. This satisfies the software processing time. The software writes I2C _ CR1.ACK bit to remove SCL low level. Fast ACK/NACK passes through I2C _ CR3.FACKEN bit control, action sequence is shown in the following figure.



26.3.2 Address Matching

As a slave, two kinds of addresses other than broadcast address and host notification address can be set, and 7-bit address or 10-bit address format can be set for slave address.

26.3.2.1 Slave Address Matching

This I₂C bus can set two kinds of slave address, and has corresponding slave address detection function. When SLADDR1EN and SLADDR0EN are "1", I2C _ SLR1 and I2C _ SLR0register subordinate addresses can be detected.

If the set slave addresses match, set the corresponding SLADDR1F, SLADDR0F to "1" at the falling edge of the 9th clock of the SCL clock, and then set the I2C _ SR.RFULLF flag or I2C _ SR.TEMPTYF flag to "1" according to the subsequent R/W # bits. As a result, I₂C _ SR can be confirmed by either receiving full interrupt or sending null interrupt. The SLADDR1F and SLADDR0F flags determine which slave address is specified.

I2C _ SR. The sequence of SLADDR1F and SLADDR0F flag changing to "1" is shown in the following figure.

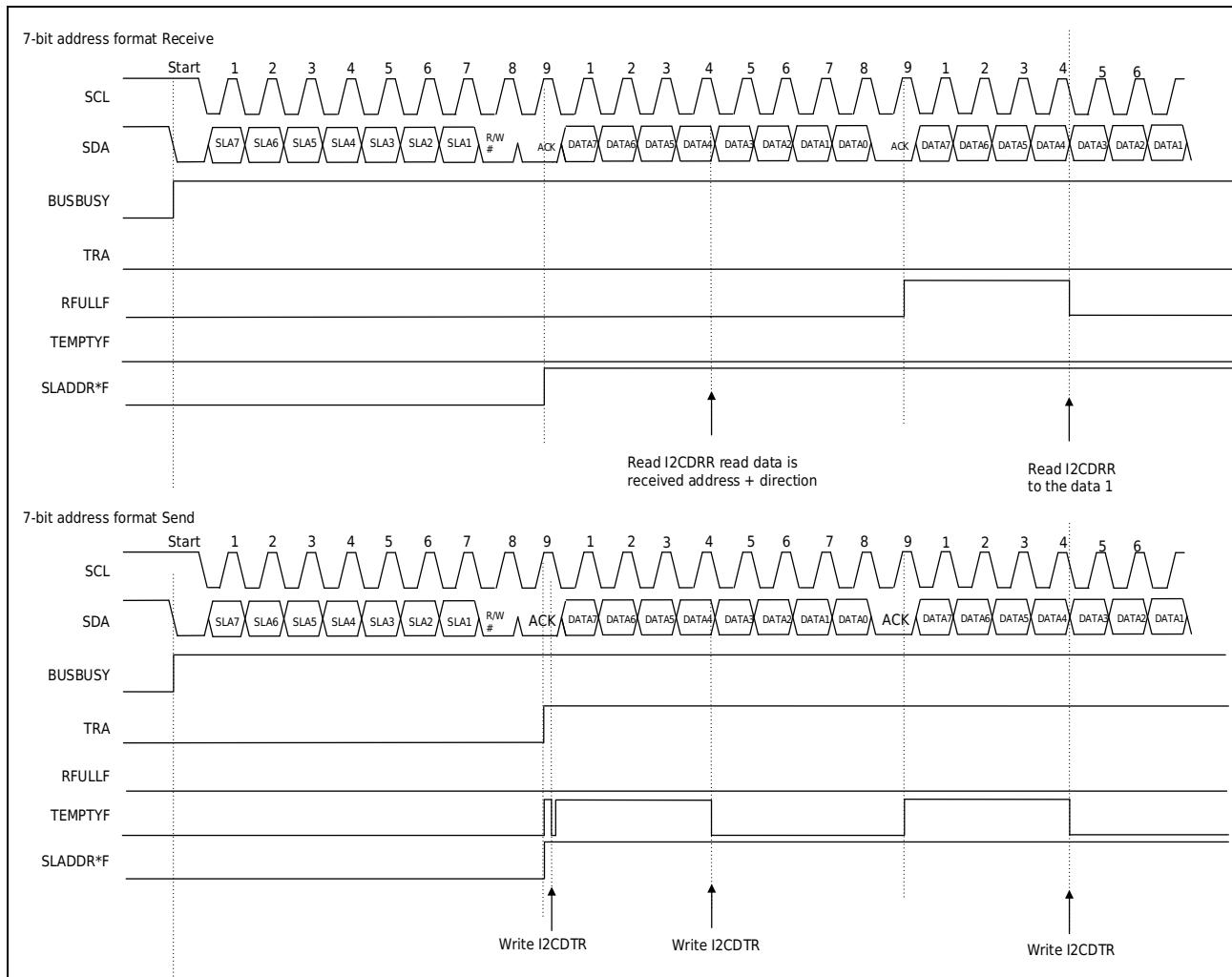


Figure 26-12 Timing When Selecting 7-Bit Address Format

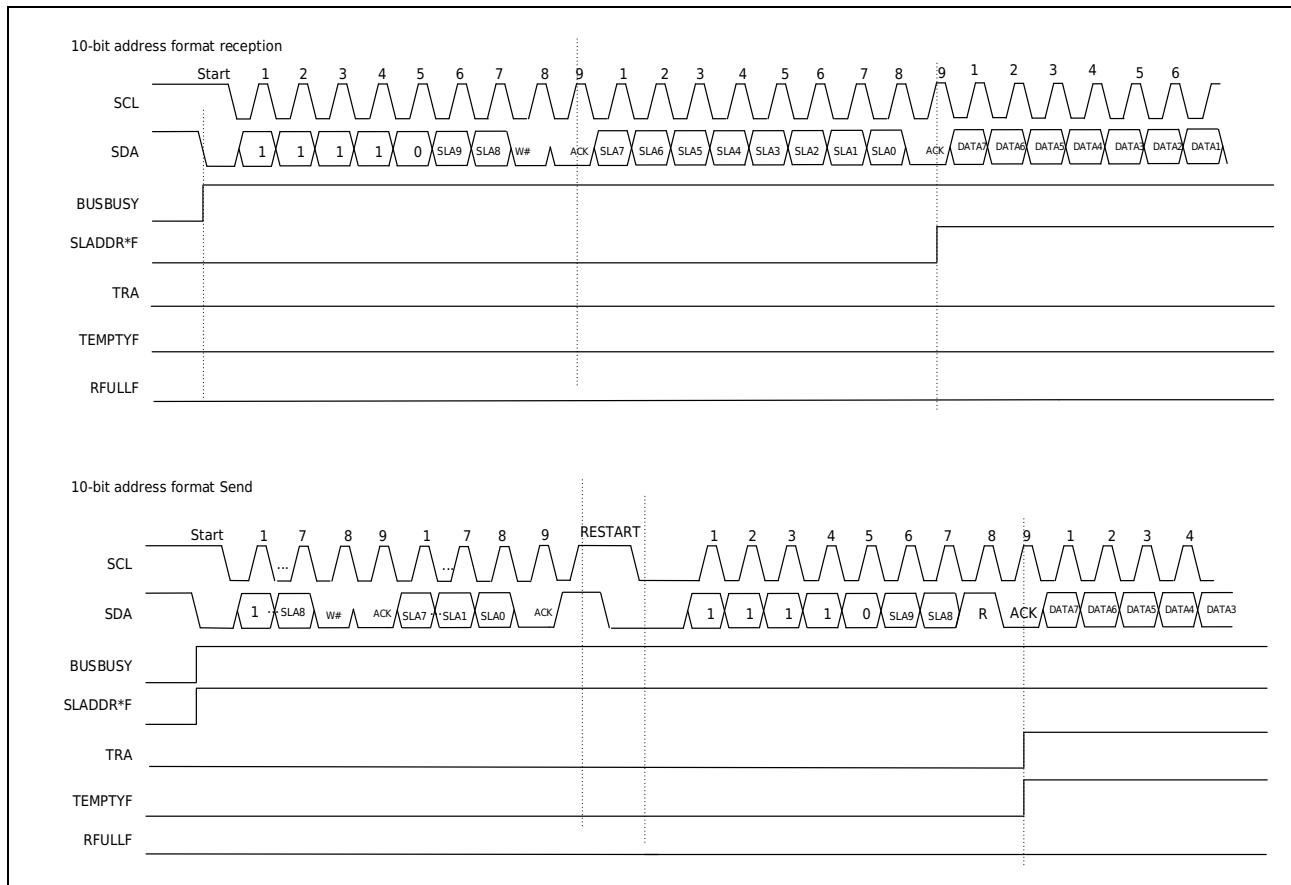


Figure 26-13 Timing When Selecting 10-Bit Address Format

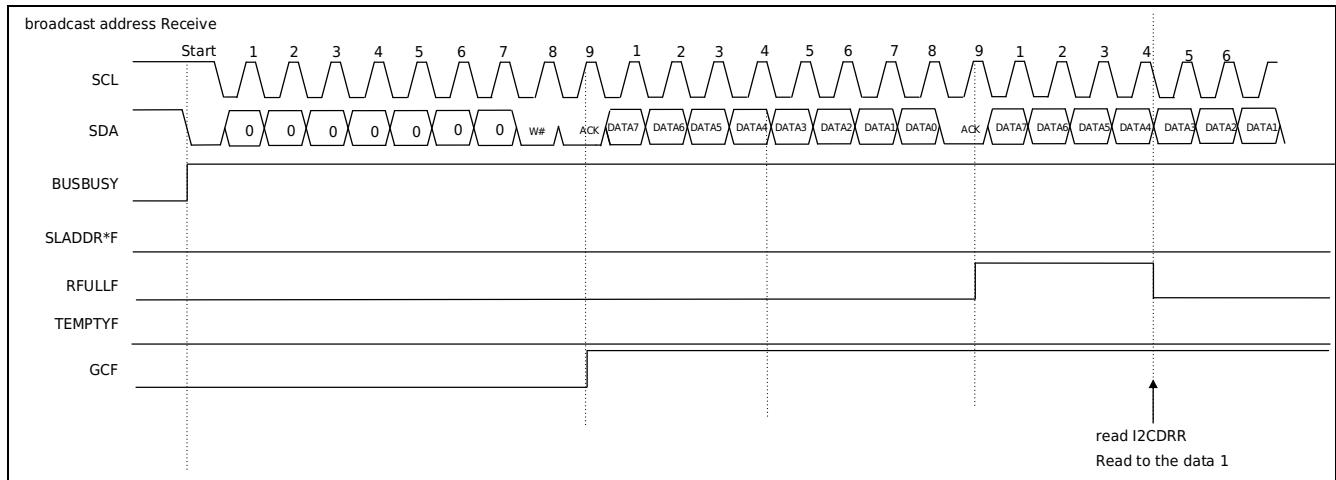
26.3.2.2 Broadcast Address Matching

When I_2C_CR1.GCEN bit is "1", the broadcast address can be detected (0000 000 b + 0 [W]).

But after the start condition or restart condition, the address is 0000 000 b + 1 [R], which is considered the slave address of All "0" and not the broadcast address.

If the broadcast address is matched, set the I2C_SR.GCF flag to "1" at the falling edge of the ninth clock of the SCL clock.

The broadcast address matches the same run as the normal slave receive run.

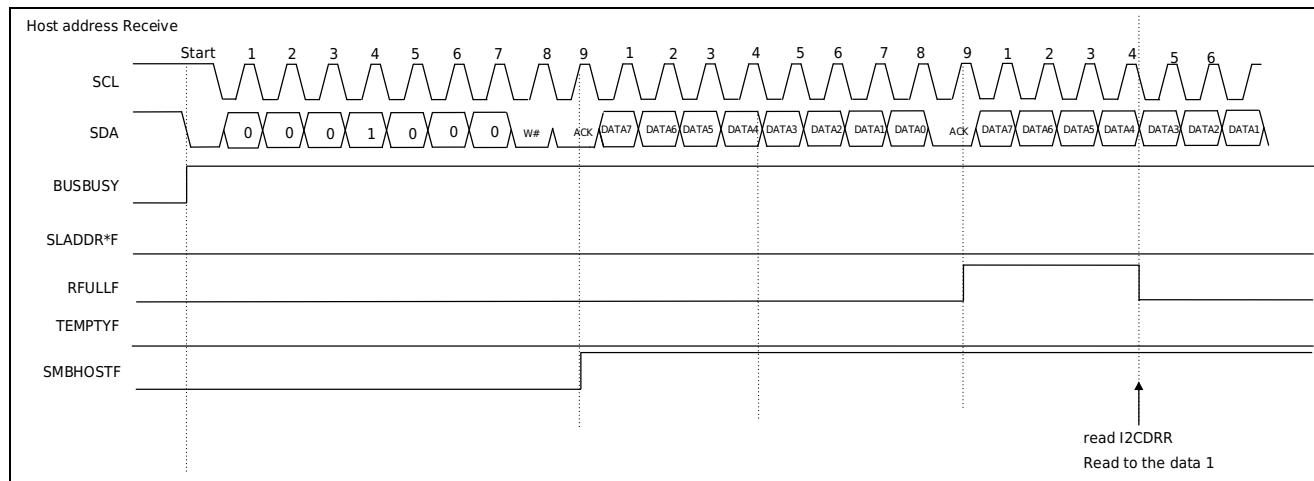


26.3.2.3 SMBus Host Address Matching

This product has the host 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 host address (0001 000b) can be detected in slave receive mode (I2C_CR1.MSL bit TRA bit is "00b").

If the SMBUS host address is detected, set the I2C_SR.SMBHOSTF flag to "1" at the falling edge of the ninth clock of the SCL clock.

Even if the bit following the SMBUS host address (0001000b) is an Rd bit (R/W # bit receives "1"), the SMBUS host address can be detected. The running of SMBUS host address detection is the same as that of normal slave mode.

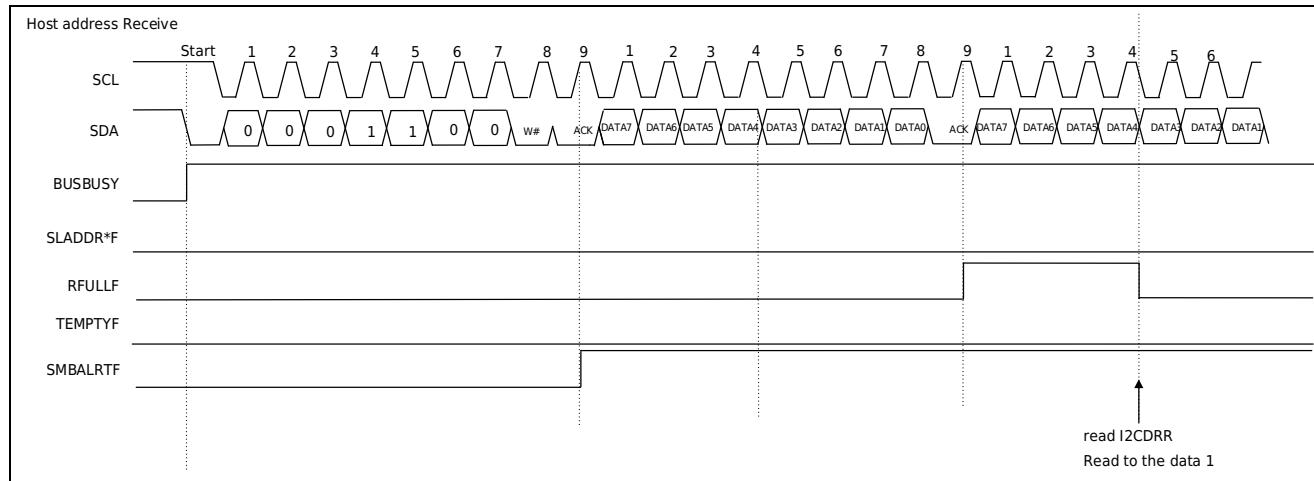


26.3.2.4 SMBus Alarm Response Address Matching

This product has the function of alarm response address detection when SMBus is running. If the I2C_CR1.SMBARLERTEN bit is "1" when the I2C_CR1.SMBUS bit is "1", the SMBUS alarm response address (0001 100b) can be detected in the slave receiver mode (I2C_CR1.MSL bit TRA bit is "00b").

If an SMBUS alarm response address is detected, the I2C_SR.SMBALERTF flag is set to "1" at the falling edge of the 9th clock of the SCL clock.

The operation of SMBUS alarm response address detection is the same as that of normal slave mode.

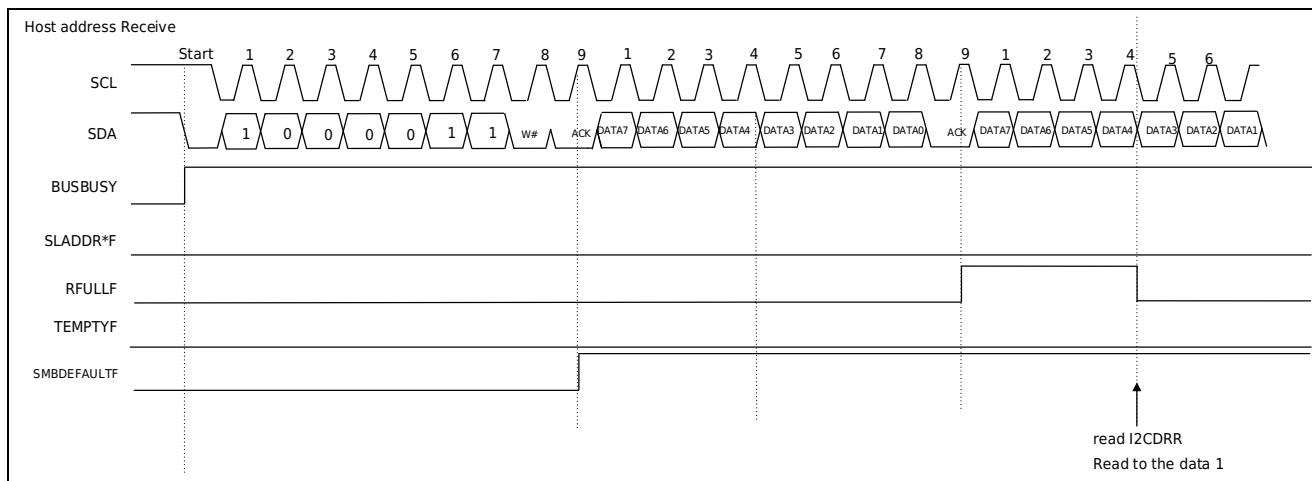


26.3.2.5 SMBus Default Address Match

This product 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 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 ninth clock of the SCL clock.

The SMBUS default address detection runs the same as the normal slave mode.



26.3.3 SMBus Action

This I2C interface can be implemented with SMBus (Ver.2.0) communication as a reference. I_2C_CR1 is required for SMBus communication. SMBUS position "1". The transmission speed must be set within the range of 10kbps to 100kbps in the SMBus specification by setting the I2C_CR1.CKDIV[2:0] bits and the I2CCCR register.

26.3.3.1 SMBus Timeout Measurement

1) SCL level When timeout Measurement

In the busy state of the bus, it can be detected by detecting that the low level or high level of the SCL line has been fixed for a certain period of time, and an abnormal state of the bus can be detected.

The overtime detection function monitors the state of the SCL line, and counts the time of high level or low level through the internal counter. If the SCL line changes (rising/falling), the internal counter is reset, otherwise it continues to count. If the internal counter counts up to the TOUTHIGH/TOUTLOW set value while the SCL line does not change, a time-out is detected and an abnormal state of the bus is notified.

For the counting of the internal counter, you can select whether to count at the low level or high level of the SCL line by setting the HTMOUT and LTMOUT bits, or to count at both low and high levels. If both the HTMOUT and LTMOUT bits are set to "0", no internal counting

will be performed.

2) Timeout measurement of slave machine

The slave device of SMBus communication needs to measure the interval shown below (timeout interval: TLOW: Set).

- Interval from start condition to stop condition

When timeout is measured by the slave device, use the start condition to detect interrupt and stop condition to detect interrupt, and use the chip timer to measure the time from detection to start condition to detection to stop condition. This timeout measurement time must be within the SMBus specification clock low level accumulation time [slave device] TLOW: SEXT: 25ms (max).

If the timer measures longer than the timeout of the SMBus specification clock low level detection TTIMEOUT: 25 ms (min), the slave computer needs to release the bus.

3) Timeout Measurement of Host

The main control unit of SMBus communication needs to measure the interval shown below (timeout interval: TLOW: MEXT).

- Interval from start condition to answer bit
- Interval from Answer to Next Answer
- Interval from answer bit to stop condition

When the host performs timeout measurement, use the start condition to detect interrupt, stop condition to detect interrupt, send end interrupt or receive data full interrupt, and use the chip timer to measure the time between zones. The timeout time must be within 10 ms (max) of the accumulated time [host] TLOW: MEXT: 10 ms (max) of the clock low level in the SMBus specification, and the cumulative time from the start condition to the stop condition must be within 25 ms (max) of TLOW: SEXT: 25 ms (max).

If the time measured by the timer exceeds the accumulated time of the clock low level in the SMBus specification [main control device] TLOW: MEXT: 10ms (max), or the accumulated time exceeds the timeout of the clock low level detection in the SMBus specification (TTIMEOUT: 25ms (min)), the host needs to abort processing. When sending a host, you must immediately abort the sending (write I2C _ DTRregister). Abort host processing through release stop conditions.

26.3.3.2 Packet Error Code (PEC)

In communication, the CPU calculates the CRC, sends the packet error code (PEC) of the SMBus, or checks the received data.

26.3.4 Reduction

It has the function of resetting the communication module. There are two kinds of resets, one for ICCR2. All registers, including the BBSY flag, are initialized and reset. The other is to remove the slave address matching state and to initialize the internal counter while maintaining various set values.

After reset, the I2C_CR1.SWRST bit must be set to "0".

Because either way, the SCL pin/SDA pin output is released to a high impedance state, it can also be used to unplug the bus from unplanned downtime.

The reset in the slave mode causes disparity with the master device, so try to avoid using it. Note: At reset (I_2C_CR1.PE and I2C_CR1.SWRST bit is "01b") The bus status such as start condition cannot be monitored during the process.

26.3.5 Interrupt and Event Signal Output

I2C has four interrupts and event outputs for triggering the activation of other peripheral circuits for user selection. These include: Occurrence of communication error (arbitration failure detection, NACK the timeout detection, start condition detection, stop condition detection), receiving end, sending data null, sending end.

The below is the 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.

Name	Event source	Event conditions
I2C_EEI	Communication error/communication time	ARLOF=1 SLADDR0F=1 SLADDR1F=1 SMBALRTF=1 SMBHOSTF=1 SMBDEFAULTF=1 GENCALLF=1 NACKF=1 TMOUTF=1 STARTF=1 STOPF=1
I2C_RXI	Received data full	RFULLF=1
I2C_TXI	Send data null	TEMPTYF=1
I2C_TEI	End of transmission	TENDF=1

26.3.6 Programmable Digital Filtering

The status of the SCL pin and SDA pin enters the interior via a digital filter. A block diagram of the digital filter circuit is shown below.

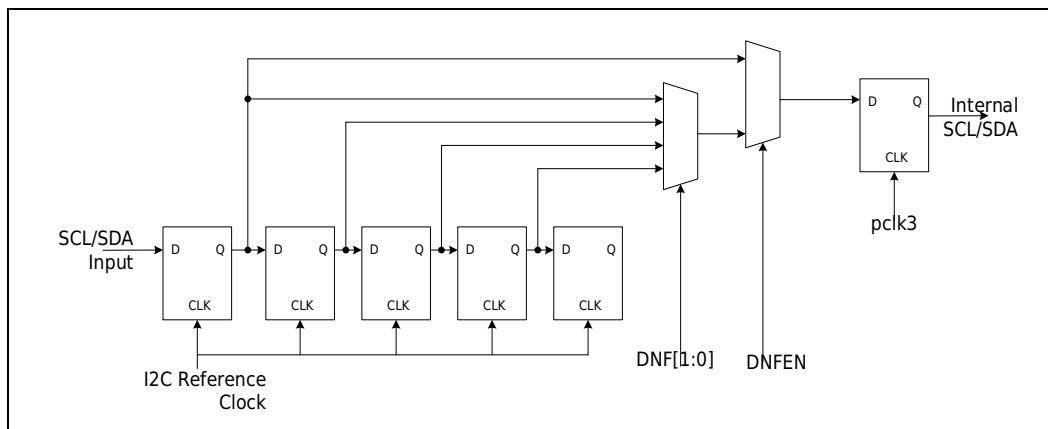


Figure 26-14 Digital Filtering Circuit Block Diagram

The internal digital filter circuit is composed of 4-segment product family-connected trigger circuit and matching detection circuit.

The number of effective segments of the digital filter is selected through the I2C_FLTR.DNF bit. According to the selected effective segment number, the noise elimination capability is 1 to 4 I2C cycles.

The input signal of the SCL pin (or the input signal of the SDA pin) is sampled on the falling edge of the I²C internal clock. If the output of the trigger circuit with the number of effective segments set

by the I2C_FLTR.DNF bit all matches, the level is used as the internal The signal is transmitted, otherwise it keeps the original value.

26.4 Application Software Setting I2C Initialization Process

When starting to send or receive data, you must initialize the steps shown in the following figure.

1. PE bit is set to 0.
2. Set SWRST to 1, communication reset
3. PE bit set 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. The initialization ends. Can send and receive data.

26.5 Register Description

I2C1 base address: 0x4004_E000

I2C2 base address: 0x4004_E400

I2C3 base address: 0x4004_E800

Table 26-2 Register Summary

Register name	Symbol	Offset address	Bit width	Reset value
I2C Control Register 1	I2C_CR1	0x00	32	0x0000_0040
I2C Control Register 2	I2C_CR2	0x04	32	0x0000_0000
I2C Control Register 3	I2C_CR3	0x08	32	0x0000_0006
I2C Control Register 4	I2C_CR4	0x0C	32	0x0030_0307
I2C slave address register0	I2C_SLR0	0x10	32	0x0000_1000
I2C slave address register1	I2C_SLR1	0x14	32	0x0000_0000
I2C state register	I2C_SLTR	0x18	32	0xFFFF_FFFF
I2C state register	I2C_SR	0x1C	32	0x0000_0000
I2C state register	I2C_CLR	0x20	32	0x0000_0000
I2C data sending register	I2C_DTR	0x24	8	0xFF
I2C data receiving register	I2C_DRR	0x28	8	0x00
I2C baud rate control register	I2C_CCR	0x2C	32	0x0000_1F1F
I2C baud rate control register	I2C_FLTR	0x30	32	0x0000_0010

26.5.1 I2C Control Register 1 (I2C_CR1)

Reset value: 0x0000_0040

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SWRST	-	-	-	-	ACK	STOP	START	RESETART	ENGC	-	SMBHOSTEN	SMBDEFALUTEN	SMBALRTEN	SMBUS	PE

Bit	Marking	Place name	Function	Read and write									
b31-16	Reserved	-	Read as "0", write as "0"	R/W									
b15	SWRST	Software reset	0: Decommission 1: Software Reset Combination of native and PE bits, selecting internal state reset or communication reset <table border="1"> <tr> <td>SWRST</td><td>PE</td><td>Reset content</td></tr> <tr> <td>1</td><td>0</td><td>Communication Reset: All registers and internal state resets within I2C.</td></tr> <tr> <td>1</td><td>1</td><td>Internal state reset: I2C_SR, I2C_DSRRegister and internal state machine reset</td></tr> </table>	SWRST	PE	Reset content	1	0	Communication Reset: All registers and internal state resets within I2C.	1	1	Internal state reset: I2C_SR, I2C_DSRRegister and internal state machine reset	R/W
SWRST	PE	Reset content											
1	0	Communication Reset: All registers and internal state resets within I2C.											
1	1	Internal state reset: I2C_SR, I2C_DSRRegister and internal state machine reset											
b14-11	Reserved	-	Read as "0", write as "0"	R/W									
b10	ACK	Send an answer	0: Reply bit sends '0' (send ACK) 1: Reply bit sends "1" (send NACK)	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 zeroing conditions: Stopped condition detected Failure to arbitrate 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 zeroing conditions: Start condition detected Failure of arbitration Communication reset	R/W									
b7	RESTART	Repeating start condition generating bit	0: Do not generate repeat start condition 1: Generate repeat start condition This bit can be set to 1 and cleared to 0. Hardware zeroing conditions: 1) Starting condition detected 2) Failure of arbitration 3) Communications resetting	R/W									
b6	ENGC	Broadcast call is	0: Invalid broadcast address detection 1: Broadcast address detection is valid	R/W									
b5	Reserved	-	Read as "0", write as "0"	R/W									
b4	SMBHOSTEN	Allow matching of SMBUS host address	0: Do not match SMBUS host address 1: Allow match of SMBUS host address	R/W									
b3	SMBDEFALUTEN	Allow matching of SMBUS default address bits	0: Do not match SMBUS default address 1: Allow match of SMBUS default address	R/W									
b2	SMBALRTEN	Allow matching of SMBUS alarm response address bits	0: Disable SMBUS alarm response address 1: Allow SMBUS alarm response address	R/W									
b1	SMBUS	SMBUS/I2C C bus mode selection bit	0: I2C bus mode 1: SMBUS bus mode	R/W									
b0	PE	I2C function enabled	0: I2C function disabled 1: I2C C function permissible Combined with SWRST bits, select internal state reset or communication reset	R/W									

26.5.2 I2C Control Register 1 (I2C_CR2)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	SMB ALRT IE	SMB HOST IE	SMB DEFA ULTI	GEN CALL IE	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	TMO UTIE	-	NAC KIE	-	-	ARL OIE	-	TEMP TYIE	RFUL LIE	-	STO PIE	TEN DIE	SLA DDR1 IE	SLA DDR0 IE	STA RTIE

Bit	Marking	Place name	Function	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	SMBALRTIE	SMBUS Alert Response Address Matching Consistent Interrupt Allow	0: SMBUS Alarm Response Address Matching Consistent Interrupt Disable 1: SMBUS Alarm Response Address Matching Consistent Interrupt Allow	R/W
b22	SMBHOSTIE	SMBUS Host Address Matching Consistent Interrupt Allowed	0: SMBUS Host Address Matching Consistent Interrupt Disable 1: SMBUS Host Address Matching Consistent Interrupt Allowed	R/W
b21	SMBDEFAULTIE	SMBUS Default Address Matching Consistent Interrupt Allowed	0: SMBUS default address match consistent interrupt disable 1: SMBUS Default Address Matching Consistent Interrupt Allowed	R/W
b20	GENCALLIE	Broadcast Call Address Matching Consistent Interrupt Allowed	0: Broadcast Call Address Matching Consistent Interrupt Disable 1: Broadcast Call Address Matching Consistent Interrupt Allowed	R/W
b19~b15	Reserved	-	Read as "0", write as "0"	R/W
b14	TMOUTIE	When timeout interrupt permission	0: When timeout interrupt disabled 1: When timeout interrupt permission	R/W
b13	Reserved	-	Read as "0", write as "0"	R/W
b12	NACKIE	NACKinterrupt Allowed	0: NACKinterrupt disabled received 1: NACKinterrupt Allowed Received	R/W
b11~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	ARLOIE	Arbitration failure interrupt permission	0: Arbitration failure interrupt prohibition 1: Arbitration failure interrupt permission	R/W
b8	Reserved	-	Read as "0", write as "0"	R/W
b7	TEMPTYIE	Send data null interrupt allowed bit	0: Send data null interrupt disable 1: Null interrupt permission for sending data	R/W
b6	RFULLIE	Received data full interrupt allowed bit	0: Full interrupt disablement of received data 1: Receive data full interrupt permission	R/W
b5	Reserved	-	Read as "0", write as "0"	R/W
b4	STOPIE	Stop condition interrupt permission	0: Bus detected stop condition interrupt disable 1: Bus detects stop condition interrupt permit	R/W
b3	TENDIE	Send a frame of data end interrupt allow bit	0: Send a frame of data end interrupt disable 1: Send a frame of data to end interrupt permit	R/W
b2	SLADDR1IE	Match unanimous interrupt permission for slave address 1	0: Match unanimous interrupt disable from machine address 1 1: Match consistent interrupt permission from machine address 1	R/W
b1	SLADDR0IE	Slave Address 0 Matches Uniform Interrupt Allowed	0: Match unanimous interrupt disable from machine address 0 1: Match consistent interrupt permission from machine address 0	R/W
b0	STARTIE	Start condition/restart condition interrupt permission	0: Bus detected start condition interrupt disable 1: Bus detected start condition interrupt permit	R/W

26.5.3 I2C Controlled Register1(I2C_CR3)

Reset value: 0x0000_0006

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

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7	FACKEN	Time point selection of RFULLF flag position	0: This bit is "1" when the 9th clock of the SCL clock rises. (At the falling edge of the 8th clock, the SCL line does not keep low level) 1: This bit is "1" when the 8th clock of the SCL clock rises. (On the falling edge of the 8th clock, the SCL line remains low. Unhold low by writing ACK bits.)	R/W
b6~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	HTMOUT	Low high level When timeout detection permissible	0: Disable time-out detection when SCL line is high. 1: Time-out detection is enabled when the SCL line is high.	R/W
b1	LTMOUT	Low level timeout detection allows	0: In SCL is low level Disable timeout. 1: Time-out detection is enabled when the SCL line is low.	R/W
b0	TMOUTEN	When timeout Feature Allowed Bit	0: SCL low level When timeout detected clock function disabled 1: The timeout function of detecting SCL level is enabled	R/W

26.5.4 I2C Control Register 1 (I2C_CR4)

Reset value: 0x0030_0307

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	BUS WAIT	-	-	-	-	-	-	-	-	-	-

Bit	Marking	Place name	Function	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 is full and I2C_DSR is empty, it will not keep low level between the 9th clock and the 1st clock of the next transmission, and continue to receive the next data. 1: When I2C_DRR is full and I2C_DSR is empty, keep the low level between the 9th clock and the 1st clock of the next transmission, and release the low level by reading the I2C_DRR register.	R/W
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

26.5.5 I2C Slave Address Register0 (I2C _ SLR0)

Reset value: 0x0000_1000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADD RMO D0	-	-	SLA DDRO EN	-	-										SLADDR0[9:0]

Bit	Marking	Place name	Function	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 address 0 significant bit	0: Invalid set value from machine address register0 1: Valid from machine address register0	R/W
b11~b10	Reserved	-	Read as "0", write as "0"	R/W
b9~b8	SLADDR0[9:8]	The high bit of a 10-bit slave address.	Set the slave address. This bit setting is invalid when ADDRMODE0 bit is "0". When ADDRMODE0 bit is "1", this bit is two digits higher than the 10-bit slave address.	R/W
b7~b0	SLADDR0[7:0]	Lower bits of 7-bit address/10-bit address	Set 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

26.5.6 I2C Slave Address Register 1 (I2C_SLR 1)

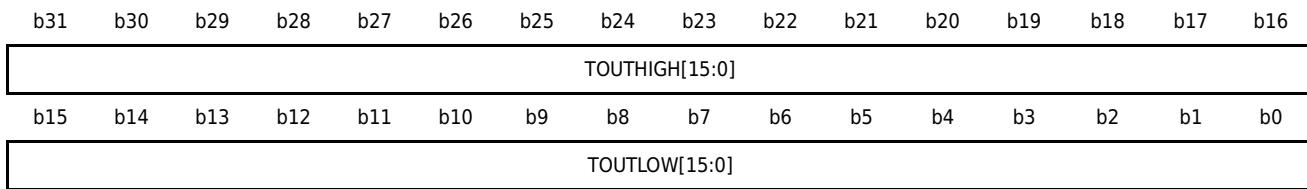
Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADD RMO D1	-	-	SLA DDR1 EN	-	-										SLADDR1[9:0]

Bit	Marking	Place name	Function	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 address 1 significant bit	0: Invalid set value from machine address register1 1: Valid from machine address register1	R/W
b11~b10	Reserved	-	Read as "0", write as "0"	R/W
b9~b8	SLADDR1[9:8]	The high bit of a 10-bit slave address.	Set the slave address. This bit setting is invalid when the ADDRMOD1 bit is "0". When ADDRMOD1 is "1", this bit is two digits higher than the 10-bit slave address.	R/W
b7~b0	SLADDR1[7:0]	Lower bits of 7-bit address/10-bit address	Set the slave address. When ADDRMOD1 bit is "0", SLADDR1 [7: 1] is the 7-bit slave address. SLADDR1 [0] bit is invalid. When ADDRMOD1 is "1", SLADDR1 [7: 0] is the lowest 8-bit address of the 10-bit slave address.	R/W

26.5.7 I2C SCL Level When Timeout Control Register (I2C_SLTR)

Reset value: 0xFFFF_FFFF



Bit	Marking	Place name	Function	Read and write
b31~b16	TOUTHIGH	SCL High Level When timeout	TOUTHIGH Set SCL High Level When timeout. SCL high level timeout time = TOUTHIGH × I2C reference clock period	R/W
b15~b0	TOUTLOW	SCL low Level When timeout	TOUTLOW Set SCL low Level When timeout. SCL low timeout time = TOUTHIGH × I2C reference clock period	R/W

26.5.8 I2C State Register (I2C_SR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	SMB ALR TF	SMB HOS TF	SMBD EFAU LTF	GENC ALLF	-	TRA	BUSY	MSL
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	TMO UTF	-	NAC KF	-	ACK RF	ARL OF	-	TEM PTYF	RFU LLF	-	STO PF	TEN DF	SLAD DR1F	SLAD DR0F	STA RTF

Bit	Marking	Place name	Function	Read and write
b31~b21	Reserved	-	Read as "0", write as "0"	R
b23	SMBALRTF	SMBUS alarm response address match consistent flag bit	0: SMBUS alarm response address not matched 1: Detected host address Set the "1" condition: Received address matches 0001 100b Clear "1" condition: SMBALRTFCLR writes "1" Stopped condition detected Communication reset	R
b22	SMBHOSTF	SMBUS Host Address Matching Consistent Flag Bit	0: Not matched to SMBUS host address 1: Match to SMBUS host address The consistent conditions for address matching are as follows: Set the "1" condition: Received address matches 0001 000b Clear "1" condition: SMBHOSTFCLR writes "1" Stopped condition detected Communication reset	R
b21	SMBDEFAULTF	SMBUS Default Address Matching Consistent Flag Bit	0: SMBUS default address not matched 1: Match to SMBUS default address Set the "1" condition: Received addresses match 1100,001b Clear "1" condition: SMBDEFAULTFCLR Write "1" Stopped condition detected Communication reset	R
b20	GENCALLF	Broadcast call address matching flag	0: Broadcast call address not matched 1: Match to broadcast call address Set the "1" condition: When the received slave address matches the broadcast call address (All "0") Clear "1" condition: GENCALLFCLR Write "1" Stopped condition detected 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 This bit can be set to 1 and cleared to 0. Hardware setting "1" condition Start condition detected In host mode, the sent R/W bit is 0 In slave mode, the address matches and the received R/W bit is 1 "0" Condition of Hardware Cleaning Stopped condition detected In host mode, the sent R/W bit is 1 In slave mode, the address matches and the received R/W bit is 0 Communication reset	R/W
b17	BUSY	Bus busy flag bit	0: Idle state, no communication on bus 1: In possession, the bus is communicating Set the "1" condition: Start condition detected on bus Clear "1" condition:	R

			Bus stop condition detected Communication reset																
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" data-bbox="759 354 1303 550"> <tr> <th>MSL</th><th>TRA</th><th>I2C operating mode</th></tr> <tr> <td>0</td><td>0</td><td>Slave receiving mode</td></tr> <tr> <td>0</td><td>1</td><td>Slave mode</td></tr> <tr> <td>1</td><td>0</td><td>Host receiving mode</td></tr> <tr> <td>1</td><td>1</td><td>Host send mode</td></tr> </table>	MSL	TRA	I2C operating mode	0	0	Slave receiving mode	0	1	Slave mode	1	0	Host receiving mode	1	1	Host send mode	R/W
MSL	TRA	I2C operating mode																	
0	0	Slave receiving mode																	
0	1	Slave mode																	
1	0	Host receiving mode																	
1	1	Host send mode																	
b15	Reserved	-	<p>This bit can be set to 1 and cleared to 0. Hardware setting "1" condition Start condition detected when START bit is 1 "0" Condition of Hardware Cleaning 1) Stop condition detected 2) Failure of arbitration 3) Communications resetting</p>	R															
14	TMOUTF	When timeout flag bit	<p>0: No SCL low level When timeout detected 1: SCL level When timeout Set the "1" condition: I2C_SCL not flipped within the cycle set by SLTR Clear "1" condition: 1: SCL level When timeout writes "1" 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 Set the "1" condition: Receive NACK in send mode Clear "0" condition: NACKFCLR writes "1" Communication reset</p>	R															
b11	Reserved	-	Read as "0", write as "0"	R															
b10	ACKRF	Receiving bit	<p>0: Received ACK bit is "0" (ACK received) 1: Received reply bit is' 1 '(receive NACK) Set the "1" condition: Receive NACK in send mode Clear "0" condition: Receive ACK in transmit mode Communication reset</p>	R															
b9	ARLOF	Arbitration failure flag bit	<p>0: No arbitration failure 1: Arbitration failure Set the "1" condition: Failure to arbitrate Clear "0" condition: ARLOFCLR Write "1" Communication reset</p>	R															
b8	Reserved	-	Read as "0", write as "0"	R															
b7	TEMPTYF	Send data null flag bit	<p>0: I2C_DTRregister full 1: I2C_DTRregister empty Set the "1" condition: I2C_DTR Data Transfer to I2C_DSR TRA position 1 Clear "0" condition: Write I2C_DTR TRA bit clear 0 Communication reset</p>	R															
b6	RFULLF	Received data full flag bit	<p>0: I2C_DRregister empty 1: I2C_DRregister full Set the "1" condition: Received data transferred from I2C DSR to I2C DRR Clear "0" condition: Read I2C DRR RFULLFCLR Write "1" Communication reset</p>	R															
b5	Reserved	-	Read as "0", write as "0"	R															

b4	STOPF	Stop condition flag bit	<p>0: No stop condition detected on bus 1: Bus detected stop condition Set the "1" condition: Stopped condition detected Clear "0" condition: STOPFCLR Write "1" Communication reset</p>	R
b3	TENDF	Send data end flag bit	<p>0: I2C_DSRRegister sending 1: End of I2C_DSRRegister Set the "1" condition: Under the condition of TEMPTYF = 1, the 9th rising edge of SCL is "1" Clear "0" condition: Stopped condition detected Write I2C_DTR TENDFCLR writes "1" Communication reset</p>	R
b2	SLADDR1F	Register 1 Matching Consistency Mark	<p>0: No consistent address detected for slave address register 1 1: Consistent address detected for slave address register 1 Set the "1" condition: When the I2C_SLR1.ADDRMOD1 bit is "0", the received slave address matches I2C_SLR1.SLADDR1[7:1]. When I2C_SLR1.ADDRMOD1 bit is "1", the first byte address of the 10-bit slave address is received and 11110b + I2C_SLR1.SLADDR1 [9: 8] matches and the second byte address is I2C_SLR1.SLADDR1 [7: 0] matches consistently. Clear "0" condition: Stopped condition detected SLADDR1FCLR Write "1" Communication reset</p>	R
b1	SLADDR0F	Match-up flag for slave address register0	<p>0: No consistent address detected for slave address register 0 1: Detected slave address register 0 consistent address Set the "1" condition: When I2C_SLR0.ADDRMOD0 is "0", the received slave address and I2C_SLR0. When SLADDR0 [7: 1] matches. When I2C_SLR0.ADDRMOD0 bit is "1", receive the first byte address of 10-bit slave address and 11110b + I2C_SLR0.SLADDR0 [9: 8] matches and the second byte address is I2C_SLR0.SLADDR0 [7: 0] matches consistently. Clear "0" condition: Stopped condition detected SLADDR0FCLR Write "1" Communication reset</p>	R
b0	STARTF	Start condition/restart condition flag bit	<p>0: Bus not detected starting condition 1: Start condition detected by bus Set "1" condition 1) Starting condition detected "0" Condition in Qing Dynasty 1) Stop condition detected 2) STARTFCLR writes "1" 3) Communications resetting</p>	R

26.5.9 I2C State Reset Register (I2C _ CLR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	SMB ALRT FCLR	SMB HOST FCLR	SMB DEFA ULTF CLR	GEN CALL FCLR	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	TMO UTFCL LR	-	NAC KFCLR	-	-	ARLO FCLR	-	TEM PTYF CLR	RFU LLFC LR	-	STO PFC LR	TEN DFC LR	SLA DDR1 FCLR	SLAD DRDF CLR	STA RTFC LR

Bit	Marking	Place name	Function	Read and write
b31~b24	Reserved	-	Write "0" on write	W
b23	SMBALRTFCLR	SMBUS Alarm Response Address Matching Consistency Mark Zero bit	Write "1" to clear the SMBALRTF flag bit	W
b22	SMBHOSTFCLR	SMBUS Host Address Matching Consistency Mark Zero bit	Write "1" to clear the SMBHOSTF flag bit	W
b21	SMBDEFAULTFCLR	SMBUS Default Address Matching Consistency Mark Zero bit	Write "1" to clear SMBDEFAULTF flag bit	W
b20	GENCALLFCLR	Broadcast call address matching flag Bit	Write "1" to clear GENCALLF flag bit	W
b19~b15	Reserved	-	Write "0" on write	W
b14	TMOUTFCLR	When timeout flag bit	Write '1' to clear the TMOUTF flag bit	W
b13	Reserved	-	Write "0" on write	W
b12	NACKFCLR	Nack flag bit	Write "1" to clear the NACKF flag bit	W
b11~b10	Reserved	-	Write "0" on write	W
b9	ARLOFCLR	Arbitration failure flag bit	Write '1' to clear the ARLOF flag bit	W
b8	Reserved	-	Write "0" on write	W
b7	TEMPTYFCLR	Send data null flag bit	Write "1" to clear TEMPTYF flag bit	W
b6	RFULLFCLR	Received data full flag bit	Write "1" to clear RFULLF flag bit	W
b5	Reserved	-	Write "0" on write	W
b4	STOPFCLR	Stop condition flag bit	Write '1' to clear the STOPF flag bit	W
b3	TENDFCLR	Send data end flag bit	Write '1' to clear the TENDF flag bit	W
b2	SLADDR1FCLR	Zero bit of matching consistency flag from machine address register1	Write "1" to clear the SLADDR1F flag bit	W
b1	SLADDR0FCLR	Zero bit of matching consistency flag from machine address register1	Write "1" to clear the SLADDR0F flag bit	W
b0	STARTFCLR	Start condition/restart condition flag zeroing	Write '1' to clear the STARTF flag bit	W

26.5.10 I2C Data Sending Register (I2C _ DTR)

Reset value: 0xFF

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

If the I2C _ DSRregister is empty, the sending data written to the I2C _ DSRregister is transmitted to the I2C _ DSRregister, and the sending mode starts sending data to the SDA.

The I2C _ DSRregister and I2C _ DTRregister are double buffers. In the course of I2C _ DSRregister data sending, if the data of I2C _ DTRregister is written in advance, the data can be sent continuously.

The I2C_DTR register is readable and writable. Write I2C _ DTRregister only once when the send data is empty interrupt request occurs.

26.5.11 I2C Data Receiving Register (I2C _ DRR)

Reset value: 0x00

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

If a frame of data is received, the received data can be transferred from the shift register (I _ 2C _ DSR) to the I _ 2C _ DRRregister, and then into the next data receiving state.

I2C _ DSRregister and I2C _ DRRregister are double buffers. In the process of receiving I2C _ DSRregister data, if the data of I2C _ DRRregister is read, the data can be received continuously.

Do not write to I2C _ DRRregister. Read I2C _ DRRregister only once when the receive data full interrupt request occurs.

In the state where the I2C _ SR.RFULLF flag bit is "1", if the next data is received immediately without reading the I2C _ DRRregister data, the SCL clock automatically remains low at the previous SCL clock when the next RFULLF flag bit is changed to "1".

26.5.12 I2C Data Shift Register (I2C _ DSR)

b7	b6	b5	b4	b3	b2	b1	b0
DSR							

I2C _ DSRregister is used to send and receive shift registers of data. I2C _ DSRregister is neither readable nor writable.

At the time of data sending, send data from I2C _ DTRRegister to I2C _ DSRRegister and send data from SDA pin. At the time of data reception, data is transferred from the I2C _ DSRRegister to the I2C _ DRRRegister once a frame of data is received.

26.5.13 I2C Clock Control Register (I2C _ CCR)

Reset value: 0x0000_1F1F

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16		
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FREQ[2:0]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
-	-	-	SHIGHW[4:0]				-	-	-	-	SLOWW[4:0]						
<hr/>																	
Bit	Marking	Place name	Function										Read and write				
b31~b23	Reserved	-	Read as "0", write as "0"										R/W				
b22~b19	Reserved	-	Read as "1", write as "1"										R/W				
b18-b16	FREQ[2:0]	Setting of I2C reference clock frequency	0 0 0: I2C reference clock frequency = PCLK3/1 0 0 1: I2C reference clock frequency = PCLK3/2 0 1 0: I2C reference clock frequency = PCLK3/4 0 1 1: I2C reference clock frequency = PCLK3/8 1 0 0: I2C reference clock frequency = PCLK3/16 1 0 1: I2C reference clock frequency = PCLK3/32 1 1 0: I2C reference clock frequency = PCLK3/64 1 1 1: I2C reference clock frequency = PCLK3/128										R/W				
			Read as "1", write as "1"														
			Set SCL high level width bit														
			Setting the High Level Width of SCL Clock														
			Read as "1", write as "1"														
			Set SCL low level width bit														
			Setting the Low Level Width of SCL Clock														

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) 250 ns (~ 100 kbps: Standard mode) 100 ns (~ 400 kbps: Fast mode)

Baud rate:

DNFE=0,FREQ=000

Baud rate = $1/\{[(SHIGHW + 3) + (SLOWW + 3)]/\Phi_{I2C} + \text{SCL rise time} + \text{SCL fall time}\}$

DNFE=1,FREQ=000

Baud rate = $1/\{[(SHIGHW + 3 + \text{filtering capability}) + (SLOWW + 3 + \text{filtering capability})]/\Phi_{I2C} + \text{SCL rise time} + \text{SCL fall time}\}$

DNFE=0,FREQ!=000

Baud rate = $1/\{[(SHIGHW + 2 + \text{filtering capability}) + (SLOWW + 2 + \text{filtering capability})]/\Phi_{I2C} + \text{SCL rise time} + \text{SCL fall time}\}$

DNFE=1,FREQ!=000

Baud rate = $1/\{[(SHIGHW + 2 + \text{filtering capability}) + (SLOWW + 2 + \text{filtering capability})]/\Phi_{I2C} + \text{SCL rise time} + \text{SCL fall time}\}$

SCL rise time + SCL fall time}

Note:

- 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 datasheet for details.

26.5.14 I2C Filtering Control Register (I2C_FLTR)

Reset value: 0x0000_0010

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	-	-	-	-	-	-	ANF EN	DNF EN	-	-	DNF[1:0]	

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R/W
b5	ANFEN	Analog filtering allowed bit	0: Analog filtering function disabled 1: Analog Filtering Allowed	R/W
b4	DNFEN	Digital filtering allowed bit	0: Digital filtering disabled 1: Digital filtering is allowed	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R/W
b1-b0	DNF[1:0]	Selection of Filtering Capabilities for Digital Filters	00: Filter capability 1 I2C reference clock cycle 01: 2 I2C reference clock cycles with filtering capability 10: Filter capability 3 I2C reference clock cycle 11: Filtering capability 4 I2C reference clock cycles	R/W

27 Serial Peripheral Interface (SPI)

27.1 Introduction

This product is equipped with 4-channel serial peripheral interface SPI, supports high-speed full-duplex serial synchronous transmission, and facilitates data exchange with peripheral devices. Users can set the range of 3/4-wire, host/slave and baud rate as needed.

Main Features of SPI:

Table 27-1 Characteristics of SPI

Essentials	Description
Number of channels	4 channel
Serial communication function	<ul style="list-style-type: none">Supporting 4-wire SPI mode and 3-wire clock synchronization modeTwo communication modes, full duplex and only transmission, are supported.Polarity and phase of adjustable communication clock SCK
Data format	<ul style="list-style-type: none">Selectable data shift order: MSB start/LSB startSelectable 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 single frame
Baud rate	<ul style="list-style-type: none">The baud rate can be adjusted by the built-in special baud rate generator in host mode. The baud rate ranges from 2 to 256 frequency division of PCLK.Maximum Baud Rate Allowed in Slave Mode is 6 Frequency Division of PCLK
data buffer	<ul style="list-style-type: none">Data buffer area with 16 bytesSupports double buffering
Error monitoring	<ul style="list-style-type: none">Mode fault error monitoringData overload error monitoringData underload error monitoringParity error monitoring
Chip selection signal control	<ul style="list-style-type: none">Each channel is configured with excluding chip-selected signal lineThe relative timing relationship between the chip select signal and the communication clock can be adjustedThe invalid time of the chip select signal between two consecutive communications can be adjustedPolarity adjustable
Transmission Control in Host Mode	<ul style="list-style-type: none">Start transmission by writing data to the data registerCommunication auto suspend function
Interrupt/AOS Source	<ul style="list-style-type: none">Received data area fullSending data area is emptySPI error (mode/overload/underload/parity)SPI IdleTransmission completion
Low power control	Configurable module stop
Other functions	<ul style="list-style-type: none">SPI initialization function

Note:

- When the communication auto-suspend feature is used in master receive mode, overload errors will not occur because the communication clock stops. Please refer to Table 22 Overload Error for details.

27.2 SPI System Block Diagram

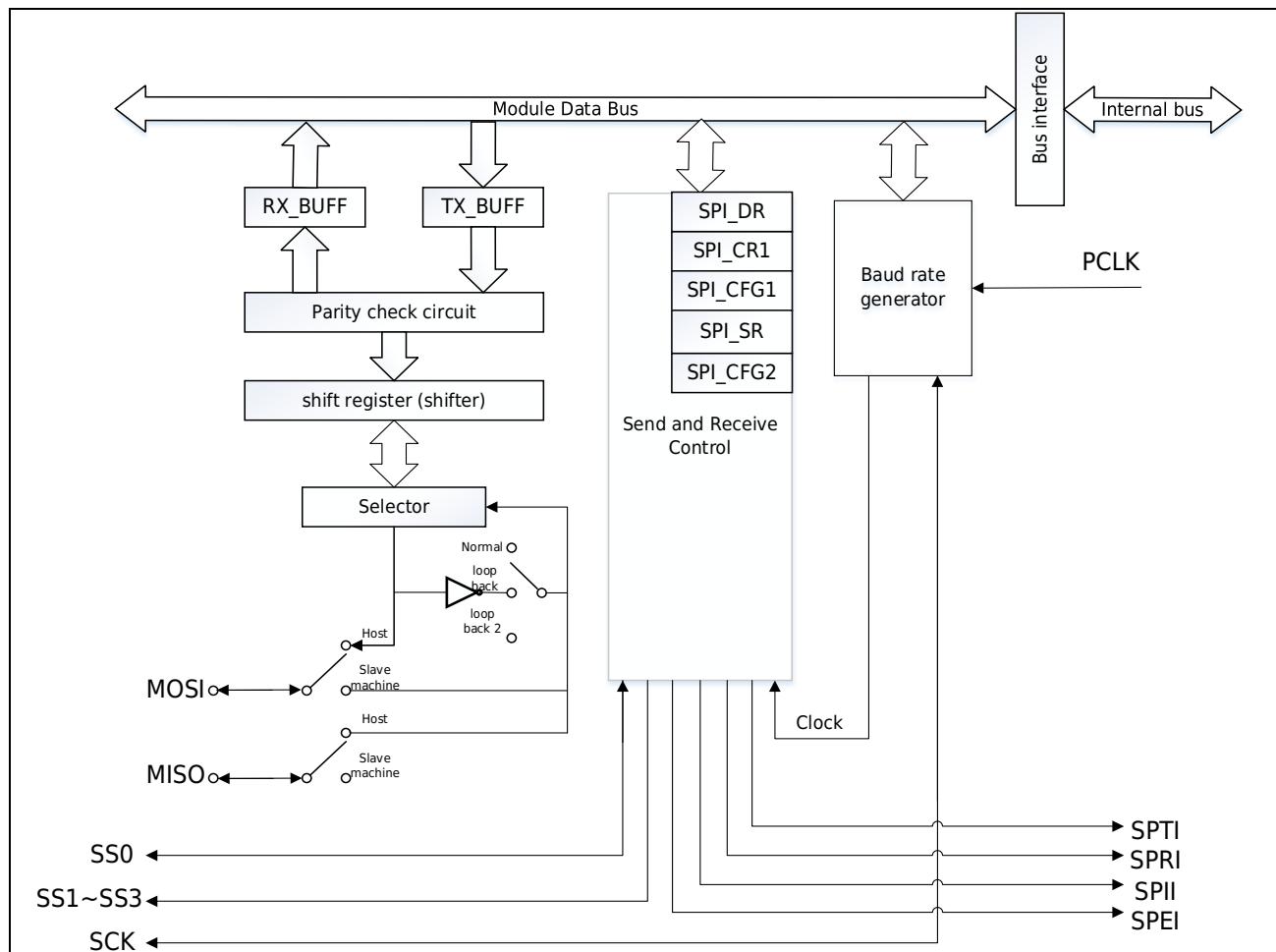


Figure 27-1 System Block Diagram

27.3 Pin Description

Table 27-2 Pin Description

Pin name	Port direction	Function
SCK	Input/Output	Communication clock pin
MOSI	Input/Output	Host data transfer pin
MISO	Input/Output	Slave data transfer pin
SS0	Input/Output	Select input/output pin from machine
SS1	Output	Select output pin from machine
SS2	Output	Select output pin from machine
SS3	Output	Select output pin from machine

27.4 SPI Action System Description

27.4.1 Pin Status in Host Mode

When SPI works in host mode, the status of each pin is shown below.

Table 27-3 SPI Pin Status Description in Host Mode

Pattern		Pin name	Pin Status (PFS.ODS = 0)	Pin status (PFS.ODS = 1)
SPI action (SPIMDS=0)	Host 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)	Host mode (MSTR=1)	SCK	CMOS Output	OD Output
		SS0~SS3(not used)	Hi-Z (available as general I/O)	Hi-Z (available as general I/O)
		MOSI	CMOS Output	OD Output
		MISO	Input	Input

Note:

- When SS0 input valid level, SPI gives out bus control, pin status is Hi-Z.

27.4.2 Pin State in Slave Mode

When SPI works in Slave machine mode, the status of each pin is shown below.

Table 27-4 SPI Pin Status Description in Slave machine 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 general I/O)	Hi-Z (available as general I/O)
		MOSI	Input	Input
		MISO (Note 2)	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 general I/O)	Hi-Z (available as general I/O)
		MOSI	Input	Input
		MISO	CMOS Output	OD Output

27.4.3 SPI System Connection Examples

Host Mode

In the host-multi-slave mode SPI system architecture, the host drives SCK, MOSI, and SS0 ~ SS3.

In SPI slave equipment 0-3, the slave equipment drives the MISO when the SS input of a slave computer is a valid level.

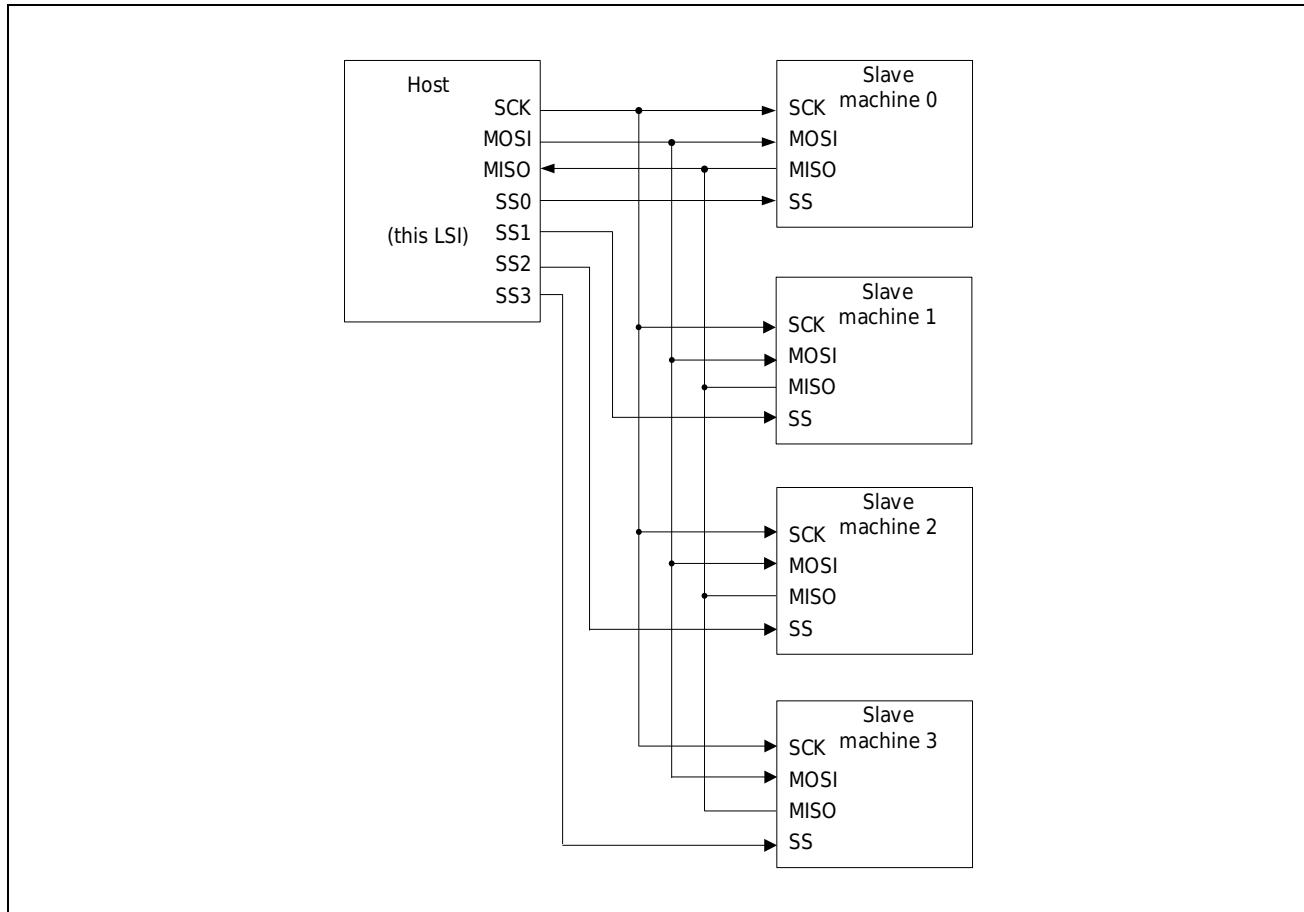


Figure 27-2 Host mode structure

Clock synchronization operation

In the SPI system structure used for clock synchronous operation, the master device drives SCK and MOSI, and the slave device drives MISO. SS pins are not used.

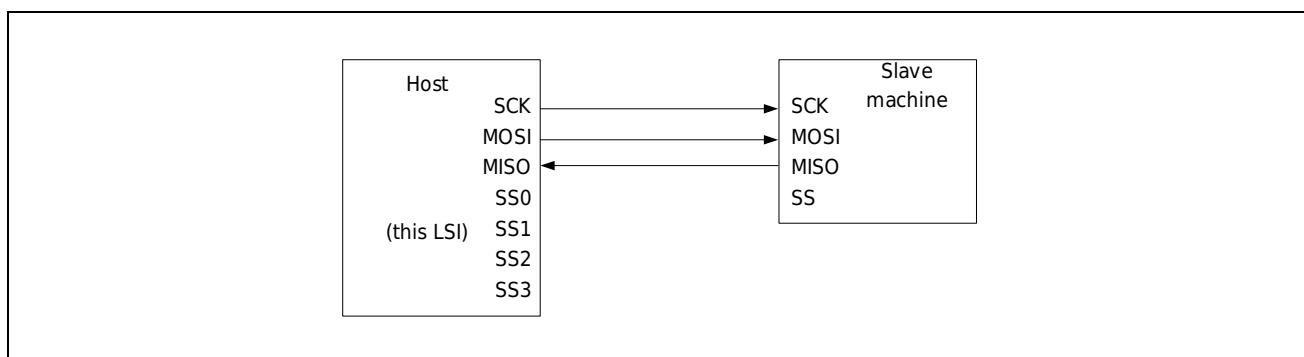


Figure 27-3 Three-Wire Clock Synchronization Operation

27.5 Data Communication Description

27.5.1 Baud Rate

In host mode, SPI clock is provided by internal baud rate generator. In slave mode, the clock is input to the SCK pin.

Baud rate depends on SPI _ CFG2.MBR [2: 0] bit setting. The calculation method is shown in the following formula, where N is the set value of MBR [2: 0] bits, ranging from 0 to 7.

$$\text{Bandrate} = \frac{f_{pck}}{2^{N+1}}$$

Table 27-5 Lower speed of partial setting

Set value of MBR [2: 0] bit	Frequency division ratio	Baud rate			
		PCLK=5MHz	PCLK=10MHz	PCLK=20MHz	PCLK=40MHz
0	2	2.50Mbps	5.00Mbps	10.0Mbps	20.0Mbps
1	4	1.25Mbps	2.50Mbps	5.00Mbps	10.0Mbps
2	8	625kbps	1.25Mbps	2.50Mbps	5.00Mbps
3	16	313kbps	625kbps	1.25Mbps	2.50Mbps
4	32	156kbps	313kbps	625kbps	1.25Mbps
5	64	78kbps	156kbps	313kbps	625kbps
6	128	39kbps	78kbps	156kbps	313kbps
7	256	20kbps	39kbps	78kbps	156kbps

27.5.2 Data Format

The SPI data format depends on the SPI command register SPI _ CFG2 and SPI control the set value of the parity permission bit PAE in register SPI _ CR1. SPI treats the data of a certain length starting from the LSB bit in the data register SPI_DR (the data length is set by the DSIZE[3:0] bits in the register SPI_CFG2) as the transfer object, regardless of the MSB/LSB shift order.

SPI_CFG2.DSIZE[3:0] determines the bit width of the data. The bit width ranges from 4 to 32 bits. SPI_CR1.PAE determines the last bit of the data. When PAE is 1, the last bit is used as the parity bit, and when it is 0, it is The lowest bit of the data itself. As shown in Figure 28-4.

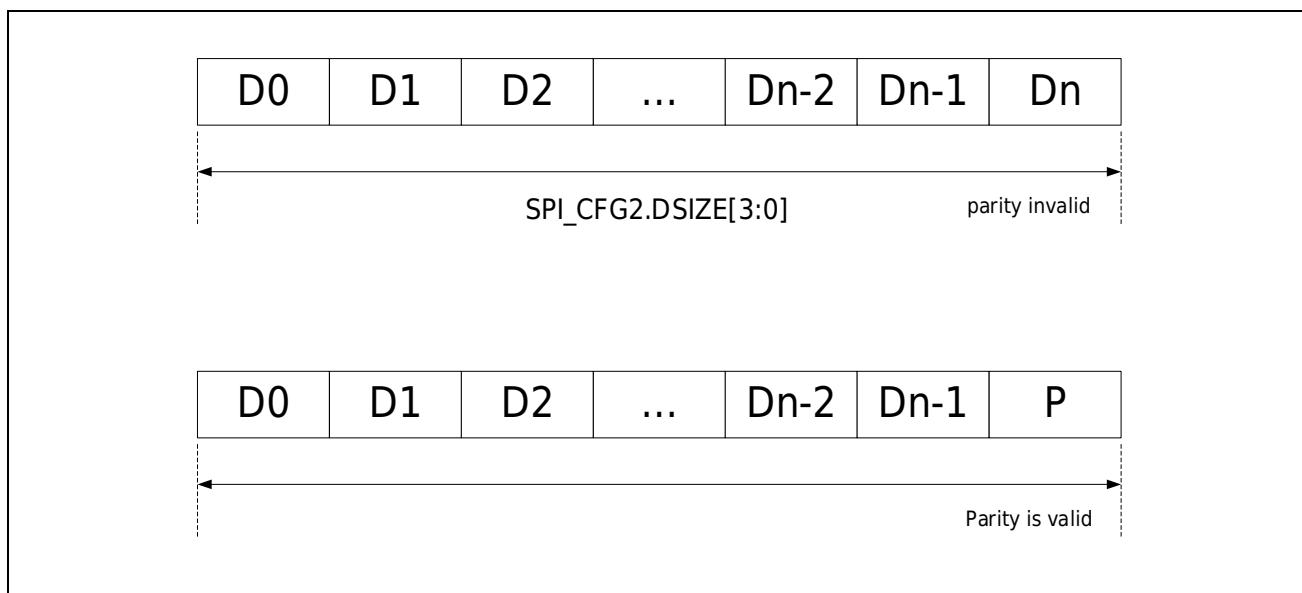


Figure 27-4 Data Format

When SPI data is sent, the transmitted data first enters the transmitting buffer (TX _ BUFF), and then copies the TX _ BUFF data to the shifter register (shifter). When SPI data is received, the data is moved in from shifter in turn, and then the shifter data is copied to the receiving buffer (RX _ BUFF).

In data transmission, the bits SPI _ CFG2 are controlled according to the shift sequence.LSBF and parity control bits SPI _ CR1.The setting of PAE is divided into four cases:

- 1) MSB first pass, parity invalid

Data d31 ~ d0 are copied from TX _ BUFF to shifter in order, and moved out of shifter in order of d31 ~ d0.

Upon receiving, data d31 ~ d0 is moved in from the lowest bit of shifter and copied to RX _ BUFF after all data is moved in.

- 2) LSB first pass, parity invalid

During transmission, d31 ~ d0 is copied from TXBUFF to shifter in the order of d0 ~ d31, and moved out of shifter in the order of d0 ~ d31.

Upon receiving, data d0 ~ d31 is moved in from the lowest bit of shifter, and then copied

from shifter to RX _ BUFF in the order d31 ~ d0 after all data is moved in.

3) MSB transmitted first, parity is valid

When transmitting, the value of parity bit P is calculated according to the value of d31 ~ d1, then P is replaced by d0, copied to shifter according to the order of d31 ~ d1, P, and moved out of shifter according to the order of d31 ~ P.

Upon receiving, the data d31 ~ P moves in from the lowest bit of shifter, and parity is performed when the data is copied to shifter. Finally, copy the data to RX _ BUFF.

4) LSB first pass, parity is valid

When transmitting, the value of parity bit P is calculated according to the value of d30 ~ d0, and then P is used instead of d31, which is copied from TXBUFF to shifter in the order of d0 ~ P and moved out of shifter in the order of d0 ~ P.

Upon receiving, the data d0 ~ P moves in from the lowest bit of shifter, and parity is performed when the data is copied to shifter. The data d0 ~ P is rearranged at the time of replication and copied to RX _ BUFF in the order of Pd0.

27.5.3 Transfer Format

1) CPHA = 0

When SPI _ CFG2. When the CPHA bit is "0", SPI samples the data at the odd edge of SCK and updates the data at the even edge. Figure 27-5 Is the transport Sequence Diagram of SPI when CPHA = 0. When the input level of the SS signal becomes active, MOSI/MISO begins to update the transmitted data. After the SS signal becomes valid for the first time on the edge of the first SCK signal, the data is sampled once per SCK cycle. MOSI/MISO signals are updated at 1/2 SCK cycles after each sampling. The set value of CPOL bit does not affect the running time of SCK signal, but only affects the polarity of signal.

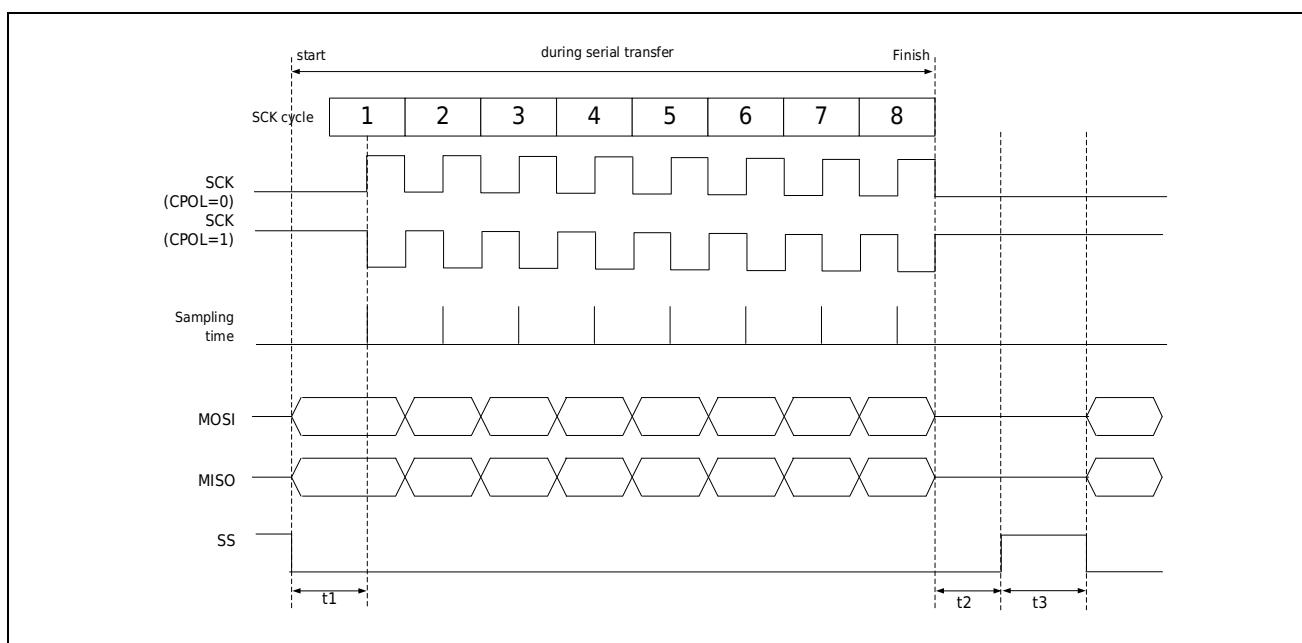


Figure 27-5 Data Transfer Format Diagram (CPHA = 0)

In the above figure, t1 represents the interval time from SS signal valid to SCK oscillation (SCK delay time), t2 represents the interval period from SCK oscillation stop to SS signal becoming invalid (SS invalid delay time), t3 represents the end of serial transmission. The minimum waiting time until the next transmission starts (next access delay). t1, t2 and t3 are controlled by the host device on the SPI system. For details, please refer to 27.6.2 the chapter.

2) CPHA = 1

When SPI _ CFG2.CPOL=0. When the CPHA bit is "1", SPI updates the data at the odd edge of SCK and samples the data at the even edge. Figure 27-6 Is the transport Sequence Diagram of SPI when CPHA = 1. MOSI/MISO starts to transmit data updates at the edge of the first SCK signal after the SS signal becomes active. After this, data is updated every SCK cycle. Data is sampled at 1/2 SCK cycles after each update. SPI_CFG2.CPOL The set value of CPOL bit does not affect the running time of SCK signal, but only affects the polarity of signal.

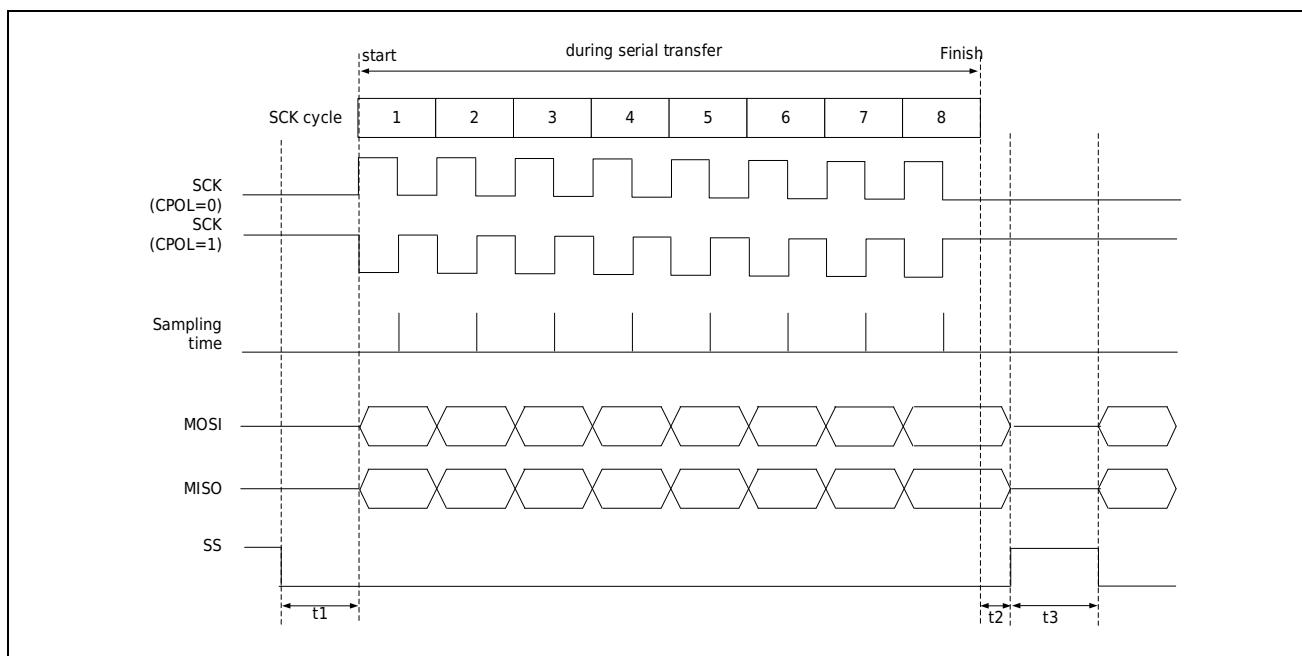


Figure 27-6 Data Transfer Format (CPHA = 1)

The same is true for t1, t2, t3 and CPHA bits.

27.5.4 Communication Mode

This SPI has two communication modes: Full-duplex synchronous serial communication and send-only serial communication, which can be selected through the TXMDS bit of SPI control register (SPI_CR1).

1) Full-duplex synchronous serial communication mode

When SPI _ CR1.WHEN TXMDS bit is "0", SPI runs in full-duplex synchronous serial communication mode. As Figure 27-7 shown, the SPI_CFG1.FTHLV[1:0] bits are "00b", the SPI_CFG2.CPHA bits are "1" and the SPI_CFG2.CPOL bits are "0", and the SPI performs 8-bit serial transfers.

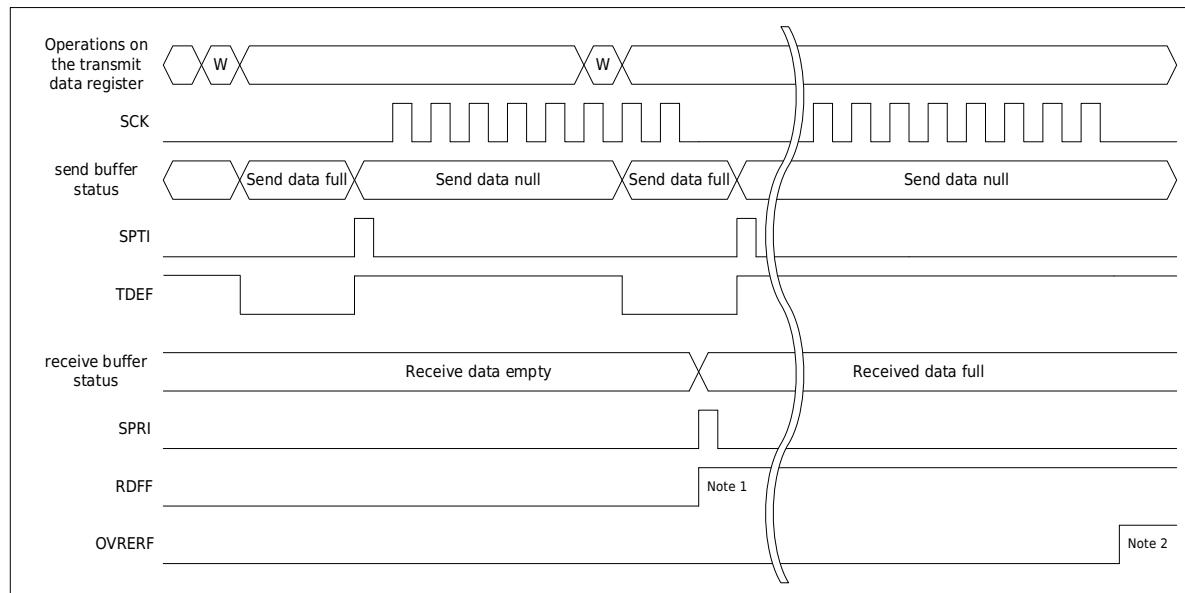


Figure 27-7 Full Duplex Synchronous Serial Communication

Note:

1. At the end of this serial transmission, if the receive data buffer register is empty, the SPI will copy the received data from the shift register to the receive data buffer register, the flag bit full of the receive data buffer register is set to 1 (RDFF), and a interrupt request (SPRI) full of the receive data is generated.
2. At the end of this serial transmission, if the received data buffer register keeps the last received data and is not read by the system, SPI will set the data overload flag to 1. The received data is invalid and the data in the received shift register will be discarded.

2) Send-only communication

When SPI_CR1.TXMDS bit is "1", SPI runs in send-only communication mode. As Figure 27-8 shown, the SPI_CFG1.FTHLV[1:0] bits are "00b", the SPI_CFG2.CPHA bits are "1" and the SPI_CFG2.CPOL bits are "0", and the SPI performs 8-bit serial transfers.

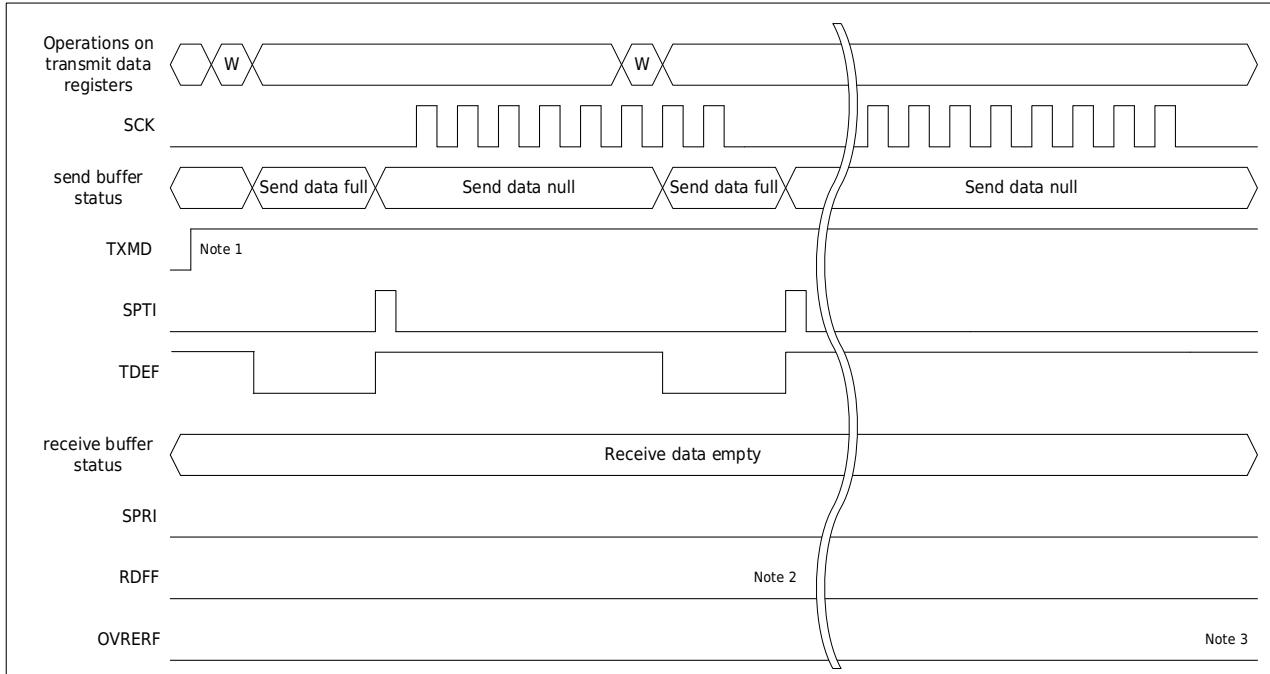


Figure 27-8 Sending Only

Note:

1. Make sure that there is no unread data in the receive buffer register (that is, RDFF is 0) and no data overload error (that is, OVRERF is 0) before setting into send-only communication.
2. At the end of this serial transmission, even if the receive data buffer register is empty, no data will be accepted, and RDFF will always remain 0.
3. The OVRERF flag bit is always in the 0 state because the receive data buffer register is always empty by sending only the communication mode.

27.6 Operating Instructions

27.6.1 Operational Mode Summary

This SPI supports 4-wire SPI mode and 3-wire clock synchronization mode. Serial communication can be performed as a host or slave in each operating mode. Set the MSTR and SPIMDS bits in the SPI control register SPI_CR1 to set the SPI mode. The relationship between the SPI mode and the setting of the SPI_CR1 register and the outline of each mode are Table 27-6 shown below.

Table 27-6 SPI Pattern and Register Setting Relationship

Pattern	Host (SPI operation)	Slave machine (SPI operation)	Host (Clock synchronous operation)	Slave machine (Clock synchronous operation)
MSTR bit setting	1	0	1	0
SPIMDS bit setting	0	0	1	1
SCK signal	Output	Input	Output	Input
MOSI signal	Output	Input	Output	Input
MISO signal	Input	Output/Hi-Z	Input	Output
SS0 signal	Output	Input	Hi-Z (not used)	Hi-Z (not used)
SS1~SS3 signal	Output	Hi-Z	Hi-Z (not used)	Hi-Z (not used)
SS Polarity Change Function	Yes	Yes	-	-
Maximum transmission rate	~PCLK/2	~PCLK/6	~PCLK/2	~PCLK/6
Timer	Internal baud rate generator	SCK input	Internal baud rate generator	SCK input
Clock polarity	2 species	2 species	2 species	2 species
Clock phase	2 species	2 species	2 species	1 species (CHPA = 1)
Start transfer bit	MSB/LSB	MSB/LSB	MSB/LSB	MSB/LSB
Transmission data length	4~32 bits	4~32 bits	4~32 bits	4~32 bits
SCK delay control	Yes	No	Yes	No
SS invalid delay control	Yes	No	Yes	No
Next Access Delay Control	Yes	No	Yes	No
Transport start method	Write send buffer by sending buffer empty interrupt request	SS Input Valid or SCK Clock Edge	Write send buffer by sending buffer empty interrupt request	SCK oscillation
Transmit buffer air detection	Yes	Yes	Yes	Yes
Receive buffer full detection	Yes (note 1)	Yes (note 1)	Yes (note 1)	Yes (note 1)
Overload error detection	Yes (note 1)	Yes (note 1)	Yes (note 1)	Yes (note 1)
Parity error detection	Yes (note 1, note 2)	Yes (note 1, note 2)	Yes (note 1, note 2)	Yes (note 1, note 2)
Underload error detection	Yes	No	Yes	No

Note:

1. When SPI _ CR1.When TXMDS bit is "1", receive buffer full detection, overload error detection and parity error detection are not performed.
2. When SPI _ CR1.When PAE bit is "0", parity error detection is not performed.

27.6.2 Host Action in SPI Operation Mode

1) Action Description for as Host

When the SPI data transmission buffer register (TX_BUFF) is empty (the TDEF flag in the status register SPI_SR is 1), write the data of the frame length set by the FTHLV[1:0] bit of the format control register SPI_CFG1 to the SPI After the data register (SPI_DR), SPI will update the SPI_DR data to TX_BUFF. If the shift register (shifter) is empty, SPI copies the TX _ BUFF data to the shift register to start serial transfer.

When transmitted data is copied to shifter, SPI will change shifter state to full state; When serial delivery is over, change to null. The status of shifter cannot be read.

When the SCK edge is needed to send the last sampling sequence of SPI, the end time of this serial transmission and SPI _ CFG2.CPHA position was independent. When the receive buffer (RX _ BUFF) is empty, SPI copies the data from shifter to RX _ BUFF after serial delivery, which can be read by the data registry SPI _ DR.

The final sampling sequence depends on the bit length of the transmitted data, and the SPI data length of the master mode depends on SPI_CFG2.DSIZE[3:0].The polarity of the SS output pin depends on the set value of SPI _ CFG1 register. For more information about SPI transport format, refer to [27.5.3 Transfer Format].

2) Initialization of SPI Host Mode

- ① Set the communication configuration register 1 (SPI_CFG1), including the setting of the baud rate, the setting of the number of frames used, the setting of various delay times, etc.
- ② Set the communication configuration register 2 (SPI_CFG2), including SS level setting, data shift sequence setting, various delay allow bit setting, data format and clock polarity phase setting, etc.
- ③ If you need to use interrupt, set the interrupt register of the system.
- ④ If you need to use DMA, set the relevant register for DMA.
- ⑤ Set the input/output pin.
- ⑥ Setting the SPI control register SPI_CR1 includes the setting of the mode and operation mode, the setting of the self-diagnosis function, the setting of the parity check, etc.
- ⑦ Verify the SPI _ CR1register setting.
- ⑧ Clear various flag bits.
- ⑨ Set the interrupt permission bit.

- ⑩ Set the SPE bit of control register SPI _ CR1 to 1 and the action begins.

27.6.3 The Slave Action in SPI Operation Mode

1) SPI as Slave Action Description

When SPI _ CFG2.CPHA bit is 0, if SPI detects that the SS0 input signal becomes active, it needs to start to output the signal to MISO to drive the active data. Therefore, when the CPHA bit is 0, the SS0 input signal level is changed from invalid to valid as a trigger signal that starts serial transmission.

When the CPHA bit is "1", if SPI detects the original SCK edge when the SS0 input signal is at an active level, it needs to start to drive valid data to the MISO output signal. Therefore, when the CPHA bit is "1", the first SCK edge of the SS0 signal in the active state is considered as the trigger signal that starts serial transmission.

If SPI detects the start of serial delivery in the shifter null state, it changes shifter to full state and cannot transfer data from TX _ BUFF to shifter during serial delivery. If shifter is full before starting serial delivery, SPI keeps shifter full.

If SPI detects the SCK edge of the last sampling sequence, then this serial transmission ends with SPI _ CFG2.CPHA position was independent. When RX _ BUFF is null, SPI copies shifter's acceptance data to RX _ BUFF after serial delivery. The data can be read by accessing SPI _ DR. SPI changes shifter to an empty state after serial delivery, independent of the state of RX _ BUFF.

During serial delivery, a mode failure error occurs if SPI detects that the SS0 input signal is invalid.

The final sampling sequence depends on the bit length of the transmitted data, and the SPI data length depends on SPI _ CFG2.DSIZE[3:0].The polarity of SS0 input signal depends on SPI _ CFG1.SS0PV bit setting. For more information about SPI transport format, refer to [27.5.3 Transfer Format].

Note:

- When SPI _ CFG2.CPHA bit is "0", the SS0 input signal level is changed from invalid to valid as a trigger signal to start serial transmission. If the SS0 input signal is fixed in a single slave mode, SPI will not start serial transmission normally. Therefore, in the structure where the SS0 input signal is fixed to the active state, the CPHA bit must be set to "1" in order to send and receive the SPI in the slave mode normally. If you need to set the CPHA bit to "0", you cannot fix the SS0 input signal.

2) SPI Slave Mode Initialization

- ① Set the communication configuration register 1 (SPI _ CFG1), mainly including the setting of the number of frames used.
- ② Set up communication configuration register2 (SPI _ CFG2), including transmission rate,

data format and clock polarity phase setting.

- ③ If you need to use interrupt, set the interrupt register of the system.
- ④ If you need to use DMA, set the relevant register for DMA.
- ⑤ Set the input/output pin.
- ⑥ Setting the SPI control register SPI_CR1 includes the setting of the mode and operation mode, the setting of the self-diagnosis function, the setting of the parity check, etc.
- ⑦ Verify the SPI _ CR1register setting.
- ⑧ Clear various flag bits.
- ⑨ Set the interrupt permission bit.
- ⑩ Set the SPE bit of control registerSPI _ CR1 to 1 and the action begins.

27.6.4 Host Action in Clock Synchronous Operation Mode

SPI is in clock synchronization mode when SPI controls SPIMDS bit 1 in registerSPI _ CR1. In this mode, SPI communicates only with SCK, MOSI and MISO pins, and SSi pins are released for common I/O functions.

Although SSi the pin is not used in the clock synchronization mode, the operation inside the module is the same as the SPI mode. However, no mode failure error was detected due to the absence of SSi the input.

1) Action Description for SPI as Host

When the SPI data transmission buffer register (TX_BUFF) is empty (the TDEF flag in the status register SPI_SR is 0), write the data with the frame length set by the FTHLV[1:0] bit of the format control register SPI_CFG1 to the SPI After the data register (SPI_DR), SPI will update the SPI_DR data to TX_BUFF. If the shift register (shifter) is empty, SPI copies the TX _ BUFF data to the shift register to start serial transfer.

When transmitted data is copied to shifter, SPI will change shifter state to full state; When serial delivery is over, change to null. The status of shifter cannot be read.

When the SCK edge is needed to send the last sampling sequence of SPI, the end time of this serial transmission and SPI _ CFG2.CPHA position was independent. When the receive buffer (RX _ BUFF) is empty, SPI copies the data from shifter to RX _ BUFF after serial delivery, which can be read by the data registry SPI _ DR.

The final sampling sequence depends on the bit length of the transmitted data, and the SPI data length of the master mode depends on SPI_CFG2.DSIZE[3:0].The polarity of the SS output pin depends on the set value of SPI _ CFG1 register. For more information about SPI transport format, refer to [27.5.3 Transfer Format].

2) Initialization Setting of Host in Clock Synchronous Operation Mode

- ① Set the communication configuration register 1 (SPI_CFG1), including the setting of the baud rate, the setting of the number of frames used, the setting of various delay times, etc.
- ② Set the communication configuration register 2 (SPI_CFG2), including the data shift sequence setting, the setting of the allowable bit of various delays, the setting of the data format and the clock polarity phase, etc.
- ③ If you need to use interrupt, set the interrupt register of the system.
- ④ If you need to use DMA, set the relevant register for DMA.
- ⑤ Set the input/output pin.
- ⑥ Setting the SPI control register SPI_CR1 includes the setting of the mode and operation mode, the setting of the self-diagnosis function, the setting of the parity check, etc.
- ⑦ Verify the SPI _ CR1register setting.
- ⑧ Clear various flag bits.
- ⑨ Set the interrupt permission bit.
- ⑩ Set the SPE bit of control registerSPI _ CR1 to 1 and the action begins.

27.6.5 Slave Maneuver in Clock Synchronous Operation Mode

1) SPI as Slave Action Description

When SPI _ CFG2.CPHA is 0, SPI needs to detect that the SS0 input signal becomes active as a trigger signal for starting serial communication. Since the SS0 pin is not used in clock synchronization mode, normal communication cannot be performed when the CPHA bit is 0.

When the CPHA bit is "1", if SPI detects the original SCK edge when the SS0 input signal is at an active level, it needs to start to drive valid data to the MISO output signal. Since the SS0 pin is not used in the clock synchronous mode of operation, when the CPHA bit is "1", the first SCK edge is considered as the trigger signal to start the serial transmission.

If SPI detects the start of serial delivery in the shifter null state, it changes shifter to full state and cannot transfer data from TX _ BUFF to shifter during serial delivery. If shifter is full before starting serial delivery, SPI keeps shifter full.

If SPI detects the SCK edge of the last sampling sequence, the serial transmission ends. When RX _ BUFF is null, SPI copies shifter's acceptance data to RX _ BUFF after serial delivery. The data can be read by accessing SPI _ DR. SPI changes shifter to an empty state after serial delivery, independent of the state of RX _ BUFF.

The final sampling timing depends on the bit length of the transmitted data, and the data length of the SPI in slave mode depends on the setting value of the SPI_CFG2.DSIZE[3:0] bits.

2) Initialization Settings of the Slave when the Clock Synchronizes the Operation Mode

- ① Set the communication configuration register 1 (SPI_CFG1), mainly including the setting of the number of frames used.
- ② Set up communication configuration register2 (SPI _ CFG2), including transmission rate, data format and clock polarity phase setting.
- ③ If you need to use interrupt, set the interrupt register of the system.
- ④ If you need to use DMA, set the relevant register for DMA.
- ⑤ Set the input/output pin.
- ⑥ Setting the SPI control register SPI_CR1 includes the setting of the mode and operation mode, the setting of the self-diagnosis function, the setting of the parity check, etc.
- ⑦ Verify the SPI _ CR1register setting.
- ⑧ Clear various flag bits.
- ⑨ Set the interrupt permission bit.
- ⑩ Set the SPE bit of control registerSPI _ CR1 to 1 and the action begins.

27.6.6 Processing Flow of Several SPI Actions

1) SPI data transfer process as host

- ① Wait for the empty interrupt of the data sending buffer register or confirm by polling that the data sending buffer register is empty.
- ② Writes the data to send to the data registry SPI_DR.
- ③ Repeat until the last data was sent.
- ④ Reset the allowable bit TXIE of send data register empty interrupt, and set the allowable bit IDIE of SPI idle state interrupt to 1.
- ⑤ Send SPI idle state interrupt.
- ⑥ Set SPE to 0, stop SPI, and reset IDIE.

2) Data reception process

- ① Wait for the full interrupt of the data receiving buffer register or confirm by polling that the data receiving buffer register is full.
- ② Read data from the receive buffer register by accessing SPI_DR.
- ③ Repeat step (2) until the last received data is read.
- ④ Clears the RXIE allow bit RXIE for the full interrupt of the data receiving buffer register.

3) Communication error handling process

- ① Wait for the communication error interrupt or confirm by polling that the communication error flag bit (MODFERF/OVRERF/UDRERF/PERF) is set to 1.
- ② Verify SS0 status and troubleshoot mode errors.
- ③ Reset SPE and stop SPI.
- ④ All SPI interrupts are allowed to be zeroed and shielded.
- ⑤ Identify the types of communication errors by error flag bits and handle them.
- ⑥ Clear the error flag bit.
- ⑦ Start SPI and restart the communication.

27.7 Self-diagnosis of Parity

Parity check circuit is composed of parity check bits for transmitting data and error detection part for receiving data. You can use the self-diagnosis function to diagnose the fault of the parity circuit as shown in the following figure.

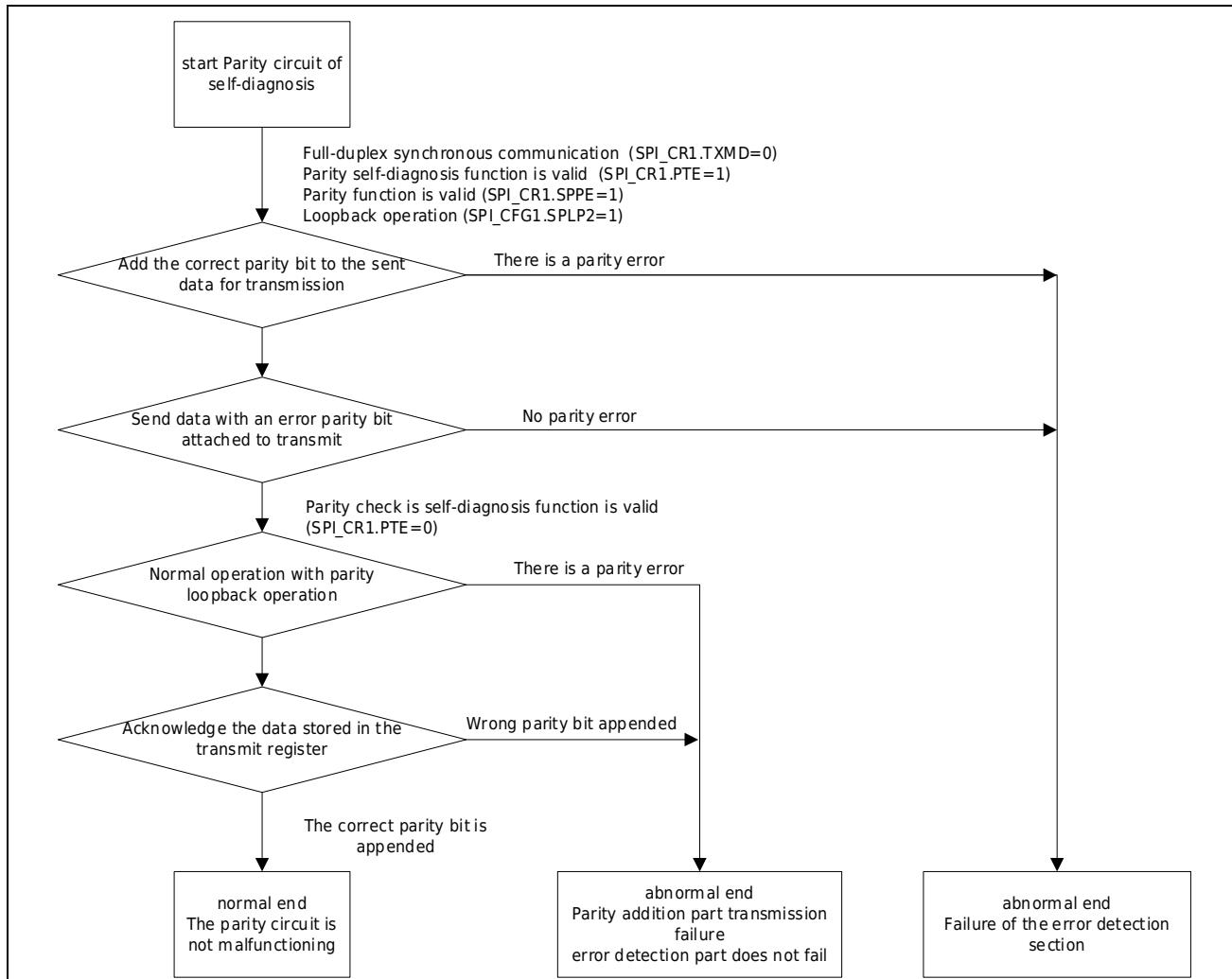


Figure 27-9 Parity Checking Process

27.8 Error Detection

In the normal SPI serial transmission, the system sends the data serially to SPI _ DRregister and obtains the data serially received by reading SPI _ DRregister. However, due to the status of the transmit/receive buffer and the SPI at the start/end of serial delivery, abnormal delivery may occur in some cases. When an abnormal transfer occurs, SPI detects this transfer as an underload error, an overload error, a parity error, or a mode failure error. The correspondence between abnormal delivery and SPI error detection is shown Table27-7 below.

Table 27-7 Correspondence Table for Error Detection

Serial number	Occurrence condition	SPI operation	Detect errors
①	Write SPI _ DRregister in full transmit buffer	<ul style="list-style-type: none"> • Keep sending buffer content • Write data loss 	No
②	Read SPI _ DRregister when Receive Buffer is empty	Output last serial received data	
③	In slave mode, serial delivery begins without data being transferred to the shift register	<ul style="list-style-type: none"> • Aborting serial transmission • Loss of sending and receiving data • Stop driving MISO output signal • Stop SPI function 	Underload error
④	Slave mode: The active level width of the SS0 pin does not reach the time required for data transmission.	<ul style="list-style-type: none"> • abort transmission • Loss of sending and receiving data • Stop SPI function 	
⑤	End serial transfer in full receiving buffer	<ul style="list-style-type: none"> • Hold Receive Buffer Content • Loss of received data 	
⑥	Error parity bits are received in full-duplex synchronous serial communication and the parity function is effective	Parity error flag is valid	Parity error

①In the cases described, SPI does not detect errors. To prevent data omissions during data write to SPI _ DRregister, data must be written to SPI _ DRregister by an empty interrupt of the send buffer. Similarly, SPI does not detect errors. To prevent unrelated data from being read, SPI _ DR data read must be done by receiving a register full interrupt request.

27.8.1 Underload Error

When the MSTR bit is 0, the SPI operates in the slave state. If the SPE is set to 1, the transmission data is not ready before the SS0 pin receives a valid level, and the SPI has an underrun error, SPI_SR.MODFERF and SPI_SR. The UDRERF flag will be set to 1.

When an underload error is detected, SPI will stop driving signal output, and SPI _ CR1.SPE is set to 0.

Monitoring of underload errors can be done by accessing SPI_SRregister directly, or by using SPI error interrupt to read SPI_SR. If you do not use the error interrupt, use the polling method to monitor for unloaded errors.

When SPI_SR.MODFERF is 1, the system disables writing 1 to SPE bits. To set SPI_CR1.SPE to 1 to enable the SPI function, the MODFERF flag must be cleared to 0 first.

27.8.2 Pattern Error

When SPI is in host mode, please do not set SPI_CR1.MODFE to 1. In slave mode, when the SSI active level width does not reach the time required for data transmission, a mode fault occurs, SPI_SR.MODFERF is set to 1, and SPI_CR1.SPE is set to 0. When transmission is required, set SPI_CR1.SPE to 1 after clearing SPI_SR.MODFERF.

27.8.3 Overload Error

SPI_SR.OVRERF flag is set to 1 if serial delivery is ended in full receive buffer. Since SPI does not copy the data of the shift register to the receive buffer in the case of an OVRERF flag of 1, the receive buffer stores the received data before an error occurs. To set the OVRERF flag to 0, you need to read SPI_SRregister in the state where the OVRERF flag is 1 before you can write "0" to the OVRERF flag.

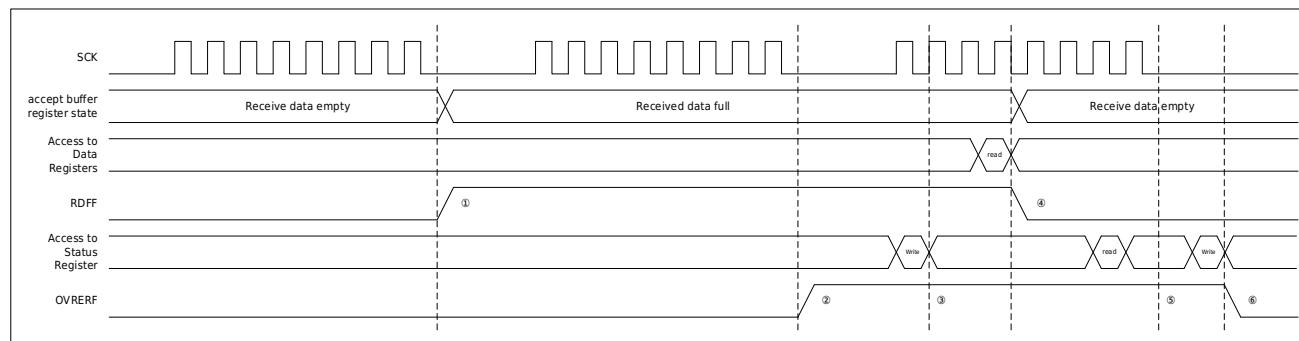


Figure 27-10 Overload Error Handling

The following describes the running contents of the flags in the time product family shown in figures ① to ⑥:

- ① The serial transmission ends when the receiving buffer is empty, and the SPI operates normally, copying the data of the shift register to the receiving buffer register, and setting the RDFF flag to 1.
- ② At the end of the serial transfer with the receive buffer full, the SPI detects an overload error and sets the OVRERF flag. SPI copies the data of the and will not set the shift register to the receive buffer. Even when the PAE bit is "1", a parity error is not detected.
- ③ Without reading the SPI_SR register, the OVRERF bit cannot be written, and clearing fails.
- ④ Read access to the data register SPI_DR allows the SPI to read the data in the receive

buffer. The RDFF flag becomes 0. Even if the receive buffer status is empty at this time, the OVRERF flag will not become 0.

- ⑤ When the serial transmission ends when the OVRERF flag is "1" (overload error), the SPI will not copy the data of the shift register to the receive buffer register, nor will it generate a receive buffer full interrupt, and the RDFF flag will remain at 0. Even if the PAE bit is "1", no parity error is detected. In the state where an overload error occurs, if the serial transmission ends without copying the received data from the shift register to the receive buffer, the SPI judges that the shift register is empty and allows data to be transferred from the transmit buffer register transfer to the shift register.
- ⑥ When the OVRERF flag is 1, read the SPI_SR register and then write 0 to the OVRERF flag, and the SPI will set the OVRERF flag to 0.

Monitoring of underload errors Yes be done by accessing SPI_SR register directly or by using SPI error interrupt to by accessing _SR. The occurrence of parity errors must be Detect as soon as possible by means of read the of transmissions DR register SPI_SR.

Normal receive operation is only possible after changing the OVRERF flag to 0.

In the host mode, if the communication auto-suspend function is enabled (set the SPI_CR1.CSUSPE bit to 1), the SPI will suspend the communication clock in the last sampling period before the overload error occurs. At this time, the shift register has not completed the last one. When the bit is received, the SPI remains in the normal communication state, and the overload error will not occur. During the suspension of the communication clock, the receiving buffer register can be read. After reading, the status of the receiving buffer register becomes empty, and the SPI restarts the communication clock to complete the last bit of data reception. For details, please refer to Figure 27-11 the in the and.

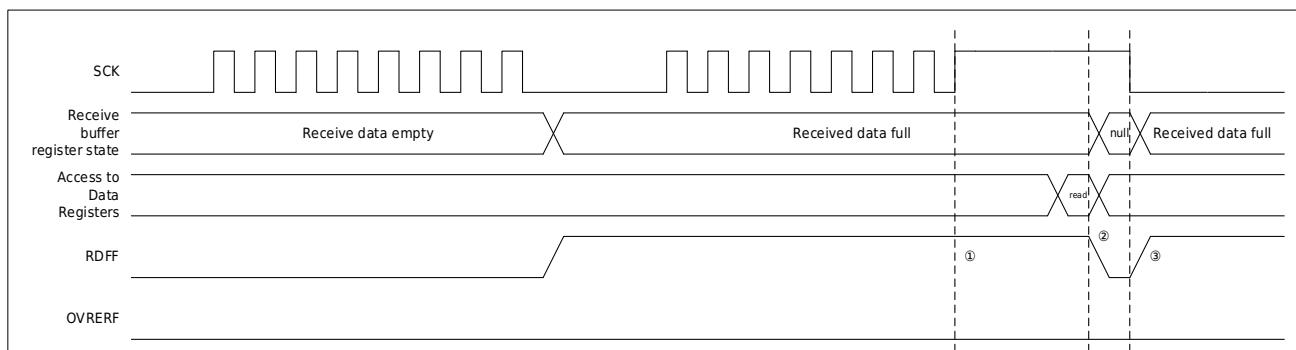


Figure 27-11 Schematic Diagram Of The Action When The Clock Automatic Stop Function Is Enabled (CPHA=1)

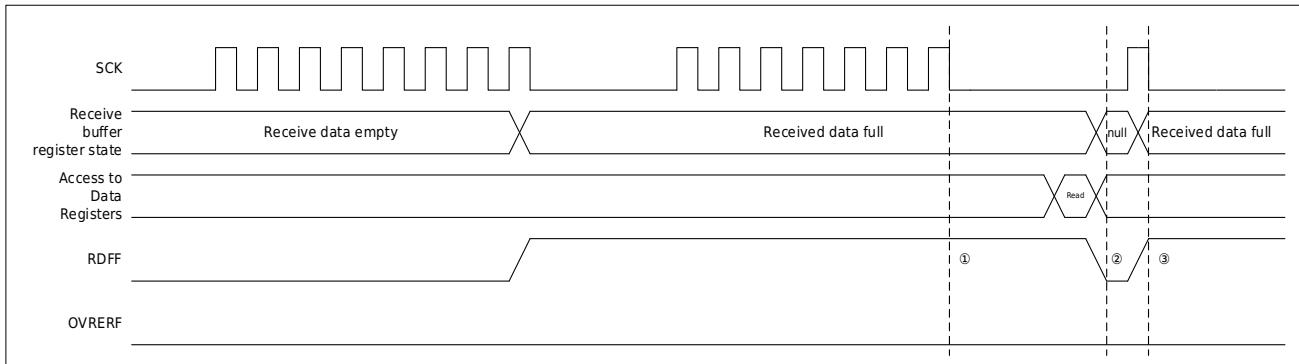


Figure 27-12 Schematic Diagram Of The Action When The Clock Automatic Stop Function Is Enabled (CPHA=0)

The following describes the running contents of the flags in the time product family shown in figures ① to ③:

- ① When the receive buffer register is full, the SPI suspends the communication clock before the last bit of data is received. At this will not occur Overload error.
- ② After reading the data in the receiving buffer register by accessing SPI_DR, the receiving buffer register becomes empty, the RDFF flag is cleared, and the SPI restarts the communication clock to complete the last bit of data communication.
- ③ The last bit of data communication is completed, the receive buffer register becomes full again, the RDFF flag is set to 1, and the received data can be read by accessing SPI_DR.

27.8.4 Parity Error

In SPI_CR1.TXMDS bit is "0" and SPI_CR1.WITH PAE bit "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, set the SPI_SR.PERF flag to 1. In the SPI_SR.OVRERF bit "1" state, the received data is not checked for parity errors because SPI does not copy the data of the shift register to the receive buffer. To clear the PERF flag, you need to read SPI_SRregister with PERF flag 1 before writing 0 to the PERF flag.

Examples of running OVRERF flags and PERF flags are shown Figure 27-13 below. In the example in the figure, SPI performs 8-bit serial transmission of full-duplex synchronous serial communication in the state where the SPI_CR1.TXMDS bit is 0 and the SPI_CR1.PAE bit is 1.

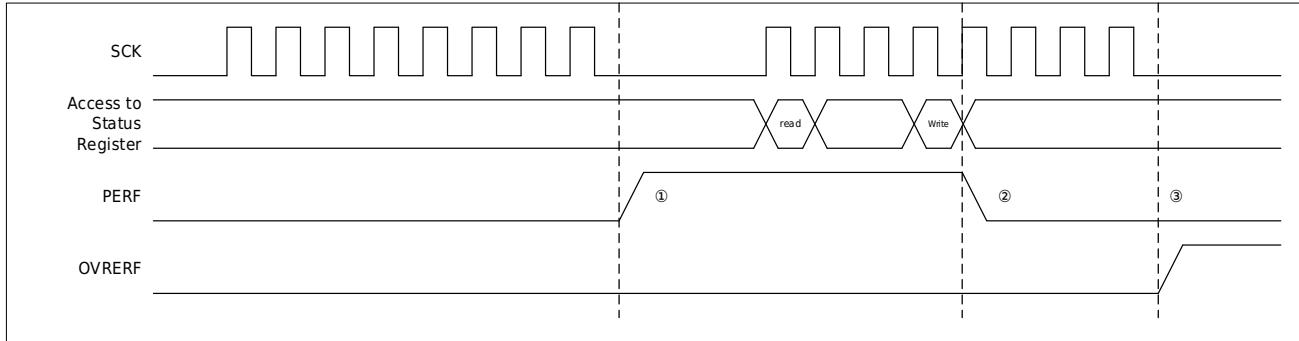


Figure 27-13 Parity Error

The following describes the running contents of the flags in the time product family shown in figures

① to ③:

- ① SPI did not detect an overload error and serial delivery ended normally. SPI copies the data of the shift register to the receive buffer. At this time, SPI performs parity check on the received data. If a parity error is detected, the PERF flag is set to 1.
- ② Read SPI _ SRregister with PERF flag 1 and write 0 to PERF flag to zero RERF flag.
- ③ SPI detects an overload error. At this time, SPI does not copy the data of the shift register to the receiving buffer, SPI does not perform parity check on the data, and no parity check error occurs.

The occurrence of parity errors can be monitored by accessing SPI _ SR register directly or by terminating SPI _ SR register via SPI error. The occurrence of parity errors must be monitored as soon as possible by means of access state register SPI _ SR.

27.9 SPI Initialization

Write operation or mode fault error detection is used to zero SPE bits, which can invalidate SPI functions and initialize some SPI functions. If a system reset occurs, all SPI functions are initialized.

27.9.1 Clear SPE Bits for Initialization

When the SPI_CR1.SPE bit is 0, the SPI performs the following initialization operations:

- Abort in-progress serial delivery.
- If you are in the slave state, stop driving the output signal (the state changes to Hi-Z).
- SPI internal state is initialized.
- Clear send buffer register, SPI_SR.TDEF flag set to 1.

SPI control bits are not initialized by zeroing SPE bits. As a result, SPI can be started in the same transport mode as before initialization, as long as SPE is relocated to 1.

Clearing SPE bits does not initialize error flag bits and sequence states. Thus, even after the SPE is cleared, it is possible to confirm the occurrence of errors during SPI transfer by reading the data of the receive buffer.

Clearing the SPE bit clears the send buffer register and sets the SPI_SR.TDEF flag to 1. Therefore, if SPI_CR1 is initialized. When TXIE bit is set to 1, SPI send buffer register is empty interrupt. To avoid this interrupt, you must reset the SPE bit to zero while setting the TXIE bit to zero.

27.9.2 System Reset Initialization

All SPI control bits, status bits and data registers are initialized by system reset.

27.10 Interrupt Source

SPI interrupt source includes receive buffer full, transmit buffer empty, mode fault, overload, underload, parity error and SPI idle. Where receive buffer full and transmit buffer empty interrupt can be used to start DMA for data transfer.

The interrupt of overload, underload and parity errors is integrated into the SPI error interrupt SPEI, so the actual interrupt source needs to be judged by the flag. A detailed description of the SPI interrupt source is shown in Table 27-8 SPI interrupt Source Description. Once the interrupt condition is established, the corresponding interrupt request is generated.

For receive buffer full and transmit buffer empty interrupt sources, data transfer changes buffer status to clear. When using DMA to send or receive, you must configure DMA before setting SPI.

Table 27-8 SPI interrupt Source Description

Interrupt source	Sketch	Interrupt condition	Start DMA
Receiver buffer full	SPRI	In SPI _ CR1.RXIE bit "1" when the receive buffer becomes full	Yes
Occurrence buffer vacancy	SPTI	In SPI _ CR1.When the TXIE bit is "1", the state delivery buffer becomes empty.	Yes
SPI errors (overload, underload, parity errors)	SPEI	When the SPI_SR.OVRERF, SPI_SR.PERF, or SPI_SR.MODFERF and SPI_SR.UDRERF flags become "1" while the SPI_CR1.EIE bit is "1"	No
SPI Idle Interrupt	SPII	In SPI _ CR1.When the IDLF flag changes to "0" in the state where the IDIĒ bit is "1"	No

27.11 Available Event Trigger Sources

SPI produces the following types of event trigger sources available:

- Data sending buffer register empty
- Data reception buffer register full
- SPI communication errors (including overload, underload, parity, etc.)
- SPI is idle
- End of SPI communication

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)].

27.12 Register Description

Register Overview

Register base address: SPI1_BASE:0x4001_C000; SPI2_BASE:0x4001_C400

SPI3_BASE:0x4002_0000; SPI4_BASE:0x4002_0400

Register name	Offset address	Reset value
SPI data register SPI_DR	0x00	0x0000_0000
SPI control register SPI_CR1	0x04	0x0000_0000
SPI Communication Configuration Register1 SPI_CFG1	0x0C	0x0000_0010
SPI status register SPI_SR	0x14	0x0000_0020
SPI Communication Configuration Register2 SPI_CFG2	0x18	0x0000_0F1D

27.12.1 SPI Data Register (SPI_DR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SPD[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SPD[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b0	SPD[31:0]	Serial data	SPI Data Storage	R/W											

27.12.2 SPI Control Register (SPI _ CR1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PAE	PAOE	PATE	MODFE	IDIE	RXIE	TXIE	EIE	CSUSPE	SPE	SPLPBK2	SPLPBK	MSTR	-	TXMDS	SPIMDS

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15	PAE	Parity permissible	0: Send data without parity bits, receive data without parity check 1: Add parity bit to send data, and perform parity check on received data (SPI_CR1.TXMDS=0); add parity bit to sent data, and not perform parity check on received data (SPI_CR1.TXMDS=1)	R/W
b14	PAOE	Parity mode selection	0: Select parity for sending and receiving 1: Select odd check for sending and receiving	R/W
b13	PATE	Parity self-diagnosis	0: The parity self-diagnostic function is invalid 1: Parity self-diagnosis function is valid	R/W
b12	MODFE	Mode fault error detection permit	0: Disable mode fault error detection 1: Allow mode fault error detection	R/W
b11	IDIE	SPI Idle Interrupt Allowed	0: Disable Idle Interrupt Request Generation 1: Allow Idle Interrupt Request Generation	R/W
b10	RXIE	SPI Receive Interrupt Permit	0: Disable SPI from receiving interrupt requests 1: Allow SPI to receive interrupt request generation	R/W
b9	TXIE	SPI Send Interrupt Permit	0: Disable SPI from sending interrupt requests 1: Allow SPI to send interrupt request generation	R/W
b8	EIE	SPI error interrupt permission	0: Disable SPI error interrupt request generation 1: Allow SPI error interrupt request generation	R/W
b7	CSUSPE	Communication auto suspend function Permissible	0: Communication auto suspend function is invalid 1: Communication auto-suspend function is valid	R/W
b6	SPE	SPI Feature Allowed	0: Invalid SPI feature 1: The SPI function is valid	R/W
b5	SPLPBK2	SPI Loop 2-bit	0: Normal mode 1: Loopback mode (send data = receive data)	R/W
b4	SPLPBK	SPI loopback position	0: Normal mode 1: Loopback mode (inverse of transmitted data = received data)	R/W
b3	MSTR	SPI Master-Slave Mode Selection	0: Slave mode 1: Host mode	R/W
b2	Reserved	-	Read as "0", write as "0"	R/W
b1	TXMDS	Communication mode selection	0: Full-duplex synchronous serial communication 1: Send Serial Communication Only	R/W
b0	SPIMDS	SPI mode selection	0: SPI operation (4-line) 1: Clock synchronous operation (3-wire)	R/W

27.12.3 SPI Communication Configuration Register1 (SPI_CFG1)

Reset value: 0x0000_0010

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-		MIDI[2:0]	-		MSSDL[2:0]	-		MSSI[2:0]	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	SS3P V	SS2P V	SS1P V	SS0P V	-	SPR DTD	-	-	-	-	-	FTHLV[1:0]

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30~b28	MIDI[2:0]	Host next access data interval idle time setting bit	b30~b28 0 0 0: 1 SCK+2 PCLK 0 0 1: 2 SCKs+2 PCLKs 0 1 0: 3 SCKs+2 SCKs 0 1 1: 4 SCKs+PCLKs 1 0 0: 5 SCKs+PCLKs 1 0 1: 6 SCKs+PCLKs 1 1 0: 7 SCKs+PCLKs 1 1 1: 8 SCKs+PCLKs	R/W
b27	Reserved	-	Read as "0", write as "0"	R/W
b26~b24	MSSDL[2:0]	Master SS invalid delay setting bit	b26~b24 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
b22~b20	MSSI[2:0]	Host SS idle time setting bit	b22~b20 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 signal polarity set	0: Low level of SS3 signal is valid 1: The high level of SS3 signal is valid	R/W
b10	SS2PV	SS2 signal polarity set	0: Low level of SS2 signal is valid 1: The high level of SS2 signal is valid	R/W
b9	SS1PV	SS1 signal polarity set	0: Low level of SS1 signal is valid 1: The high level of SS1 signal is valid	R/W
b8	SS0PV	SS0 signal polarity set	0: Low level of SS0 signal is valid 1: The high level of SS0 signal is valid	R/W
b7	Reserved	-	Read as "0", write as "0"	R/W
b6	SPRD TD	Data register read target selection	0: SPI_DR read receive buffer 1: SPI_DR read send buffer (must be read 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]	FPS	b1~b0 0 0: 1 frame 0 1: 2 frames 10: 3 Frame 1 1: 4 Frame	R/W

27.12.4 SPI Status Register (SPI_SR)

Reset value: 0x0000_0020

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

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7	RDFF	Receive buffer full flag	0: SPI_DRregister no data 1: SPI_DRregister has data Hardware set, clear, write "1" when writing	R
b6	Reserved	-	Read as "0", write as "0"	R/W
b5	TDEF	Transmit buffer null flag	0: The transmit buffer has data 1: No data in the occurrence buffer Hardware set, clear, write "1" when writing	R
b4	UDRERF	Underloaded error mark	0: Mode fault error occurs (MODFERF = 1) 1: Underload error occurs (MODFERF = 1) This bit is initialized when MODFERF = 0 Hardware setting alignment After reading 1 0 and reset	R/W
b3	PERF	Parity error flag	0: No parity error occurred 1: Parity error Hardware setting alignment After reading 1 0 and reset	R/W
b2	MODFERF	Mode fault flag	0: Unsent mode failure error 1: Mode failure error Hardware setting alignment After reading 1 0 and reset	R/W
b1	IDLNF	SPI Idle Mark	0: SPI is idle 1: SPI is transport Hardware set, clear	R
b0	OVRERF	Overload error mark	0: No overload error occurred 1: Overload error Hardware setting alignment After reading 1 0 and reset	R/W

27.12.5 SPI Communication Configuration Register2 (SPI_CFG2)

Reset value: 0x0000_0F1D

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MSSI E	MSS DLE	MIDI E	LSBF	DSIZE[3:0]	SSA[2:0]	MBR[2:0]	CPOL	CPHA							

Bit	Marking	Place Name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15	MSSIE	SCK delay allows	0: SCK delay is 1 SCK 1: SCK delay is the set value of MSSIE	R/W
b14	MSSDLE	SS Invalid delay allows	0: SS invalid delay is 1 SCK 1: SS Invalid delay is the set value of MSSDLE	R/W
b13	MIDIE	SPI next delay allows	0: The next access delay is 1 SCK+2 PCLK 1: next delay is the set value of MIDI	R/W
b12	LSBF	SPI LSB first bit	0: MSB first 1: LSB first	R/W
b11~b8	DSIZE[3: 0]	Positioning SPI Data Length	b11~b8 0 0 0 0: 40-bit 0 0 0 1: 5-bit 0 0 1 0: 6-bit 0 0 1 1: 7-bit 0 1 0 0: 8-bit 0 1 0 1: 9-bit 0 1 1 0: 10-bit 0 1 1 1: 11-bit 1 0 0 0: 12-bit 1 0 0 1: 13-bit 1 0 1 0: 140-bit 1 0 1 1: 15-bit 1 1 0 0: 16-bit 1 1 0 1: 20-bit 1 1 1 0: 240-bit 1 1 1 1: 32-bit	R/W
b7~b5	SSA[2: 0]	SS signal valid setting bit	b7~b5 0 0 0: SS0 0 0 1: SS1 0 1 0: SS2 0 1 1: SS3 1 x x: Setting prohibited	R/W
b4~b2	MBR[2: 0]	Bit Rate Frequency Division Location	b4~b2 0 0: Select the 2-division frequency of the base rate 0 0 1: Selecting 4-Division Frequency of Base Rate 0 0 0: Selecting 8-Division Frequency of Base Rate 0 1: Selecting 16-Division Frequency of Base Rate 1 0: Select the 32-division frequency of the base rate 1 0 1: Selecting 64-Division Frequency of Base Rate 1 0 0: Select the 128 division frequency of the base rate 1 1: Select the 256-division frequency of the base rate	R/W
b1	CPOL	SCK polarity set bit	0: Low SCK at idle 1: SCK is high when idle	R/W
b0	CPHA	SCK phase setting bit	0: Data sampling at odd edges, changing at even edges 1: Changes in data at odd edges and data sampling at even edges	R/W

28 Quad-wire Serial Peripheral Interface (QSPI)

28.1 Introduction

The Quad Serial Peripheral Interface (QSPI) is a memory control module designed to communicate with serial ROMs with an SPI-compatible interface. Its objects mainly include serial flash memory, serial EEPROM and serial FeRAM.

Table 28-1 QSPI main Specifications

Parameter	Specifications
Number of channels	1 channel
SPI	<ul style="list-style-type: none">Support multiple protocols such as extended SPI, two-wire SPI and four-wire SPI
	<ul style="list-style-type: none">SPI mode 0 and SPI mode 3 are supported.
	<ul style="list-style-type: none">Address line width can be selected from 8 bits/16 bits/24 bits/32 bits
timing adjustment	Can be adjusted through timing to support various serial flash drives
flash read	<ul style="list-style-type: none">Support multiple reading methods<ul style="list-style-type: none">a. Standard Read/Fast Readb. Two-wire output fast read/two-wire input and output fast readc. Two-wire output fast read/two-wire input and output fast readFree setting commandAdjustable number of dummy cycles16-byte read-ahead functionstatus query functionSPI bus cycle extension functionXIP control Function
	Flexible and extensive support for a large number of serial flash software control commands, including erasing, writing, ID reading and power-down control, etc.
Interrupt source	Hardware Error interrupt
Module stop function	can be Module stop reduce power consumption

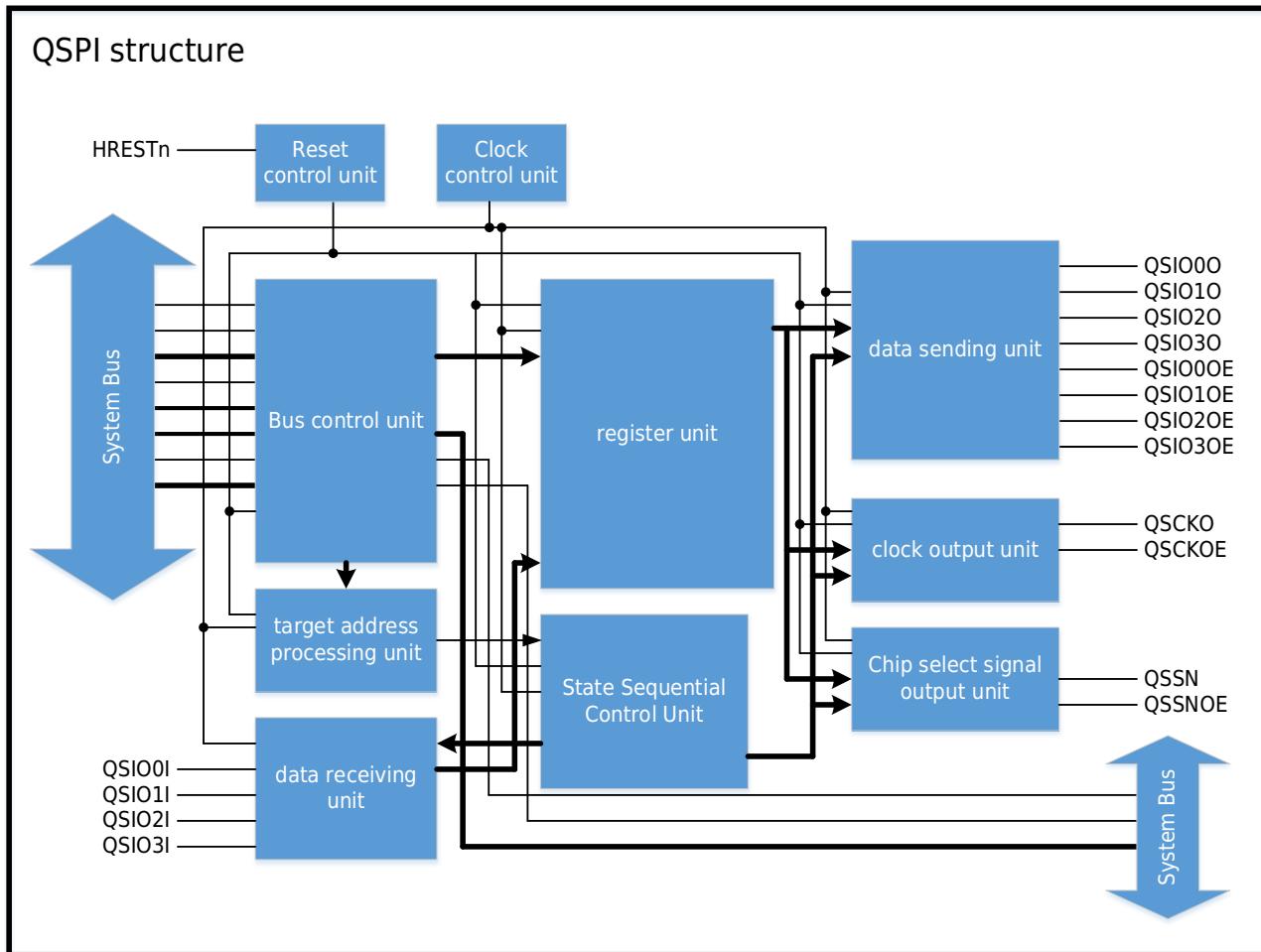


Figure 28-1 Module composition diagram of QSPI

Table 28-2 QSPI Pin

Pin name	Input/Output	Functional description
QSCK	Output	QSPI clock output pin
QSSN	Output	QSPI Select pin from machine
QSI00	Input/Output	Data line 0
QSI01	Input/Output	Data line 1
QSI02	Input/Output	Data line 2
QSI03	Input/Output	Data line 3

28.2 Memory Map

28.2.1 Internal Bus Space

The location of the serial flash memory and related control registers in the AHB bus space is determined by the overall address range configuration.

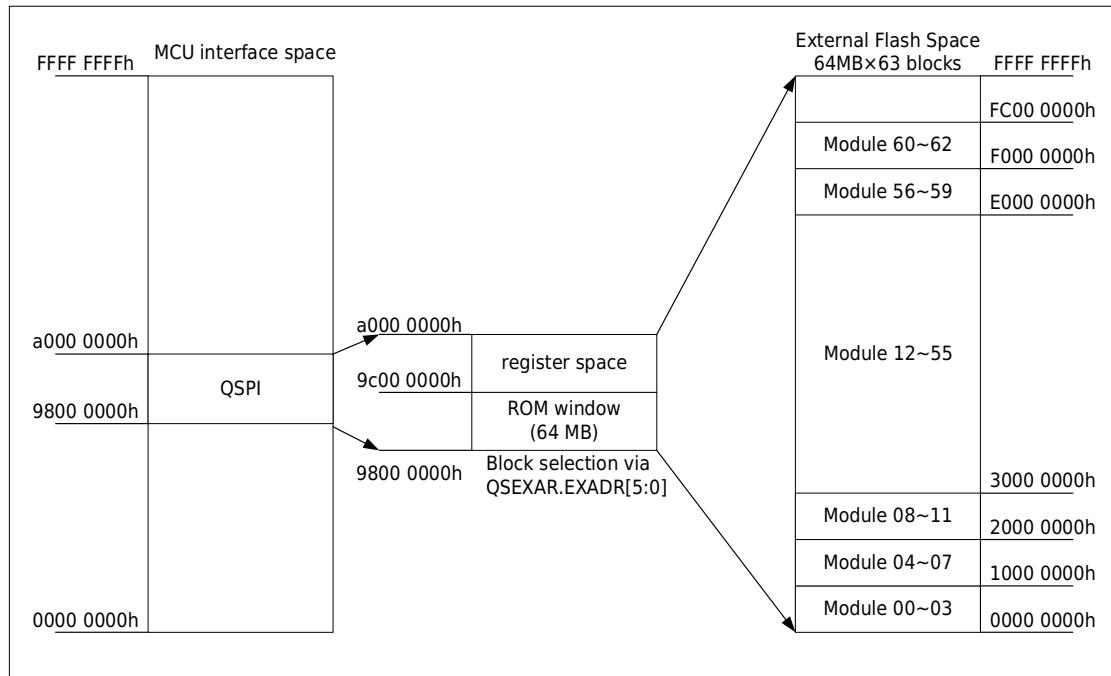


Figure 28-2 Default Area Setting and AHB Bus Space Memory Mapping Relationship Diagram

28.2.2 ROM Space and Bus Address Width

QSPI has a 32-bit address bus width to match serial flash memory. Whenever the ROM space of QSPI is read and accessed, the QSPI bus starts to work automatically and transmits the data read from the serial flash memory.

QSPI not only can only use 32-bit address bus width, but also can choose to use 8-bit/16-bit/24-bit address bus width by setting AWSL[1:0] in the QSFCR register.

If the address bus width of 8-bit/16-bit/24-bit is selected, only the low-order space whose address matches it can be accessed normally, that is, accessing the high-order serial flash image space in QSPI will repeatedly appear in the low-order space Content.

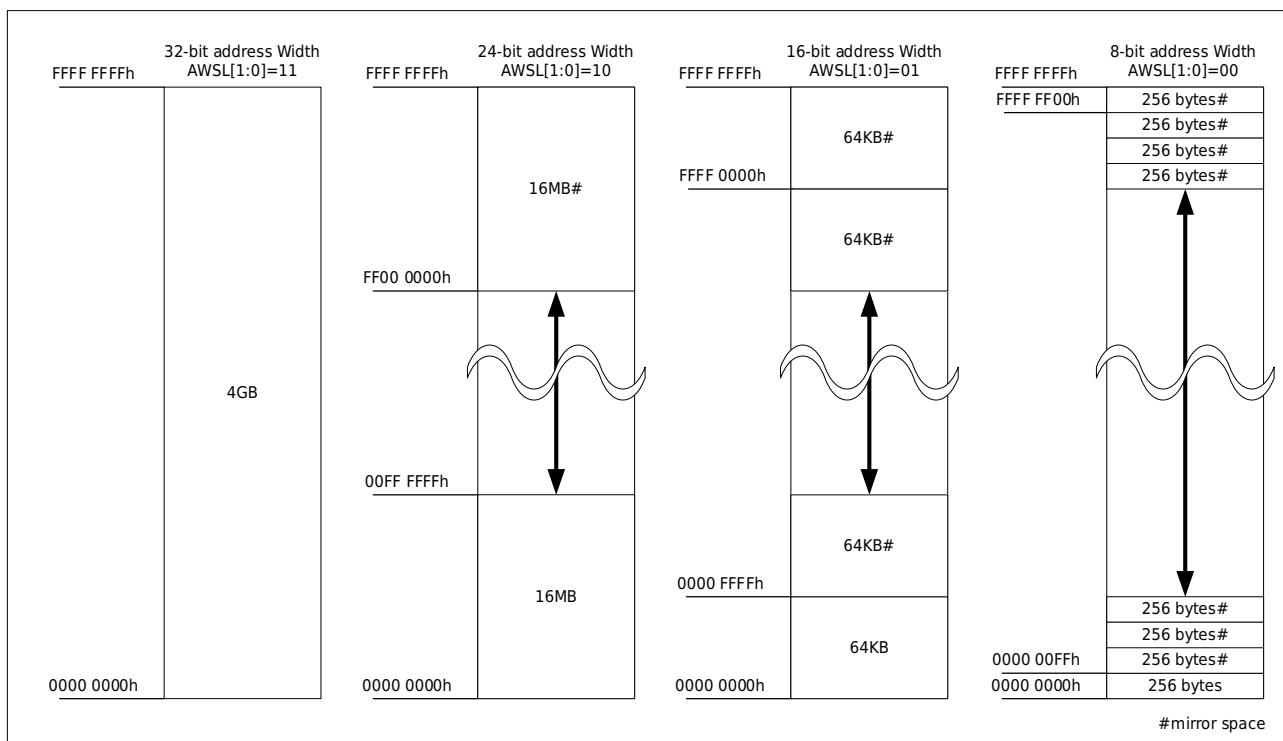


Figure 28-3 QSPI-ROM Space Memory Map

Note:

- The address bus width can be selected to use 8 bits/16 bits/24 bits/32 bits by setting AWSL[1:0] in the QSFCR register.

28.3QSPI Bus

28.3.1 SPI Protocol

This QSPI Support three ways to as extended wire SPI, two-wire SPI and four-wire SPI. The initial default protocol is the extended SPI protocol. The protocol of each stage can be configured separately by setting the DPRSL[1:0]/APRSL[1:0]/IPRSL[1:0] bits in the QSCR register. The extended SPI protocol only uses the QSIO0 pin single-wire for instruction output, and the subsequent addresses and data are output in single-wire/two-wire/four-wire according to the specific read mode instructions.

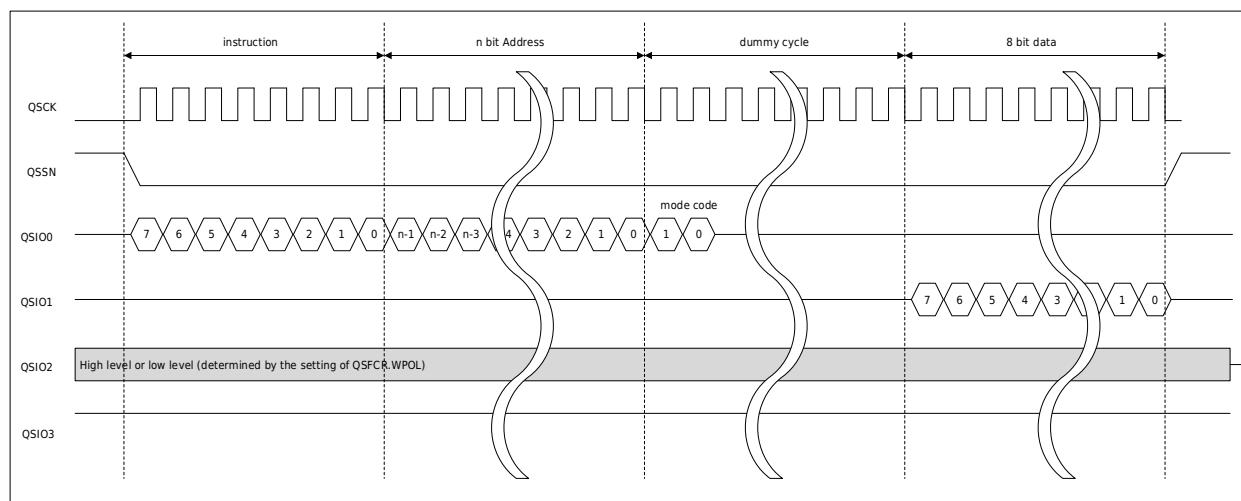


Figure 28-4 Schematic Diagram of Extended SPI Protocol Action 1 (Fast Read Mode)

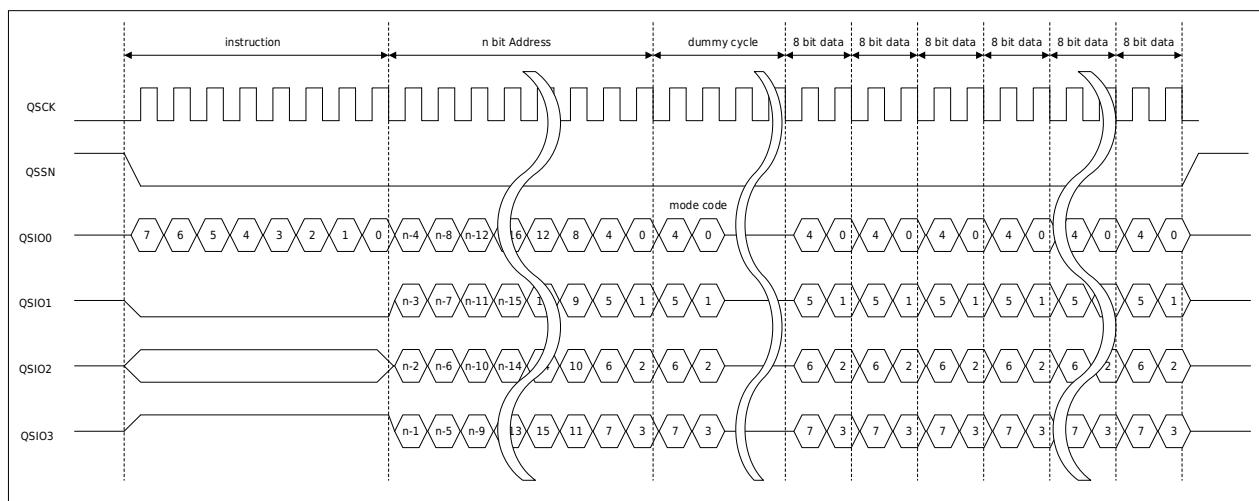


Figure 28-5 Schematic Diagram Of Extended SPI Protocol Action 2 (Wire Input Fast Read Mode)

The two-wire SPI protocol uses the two pins QSIO0 and QSIO1 to perform corresponding operations, including sending instructions, addresses, receiving data, and so on.

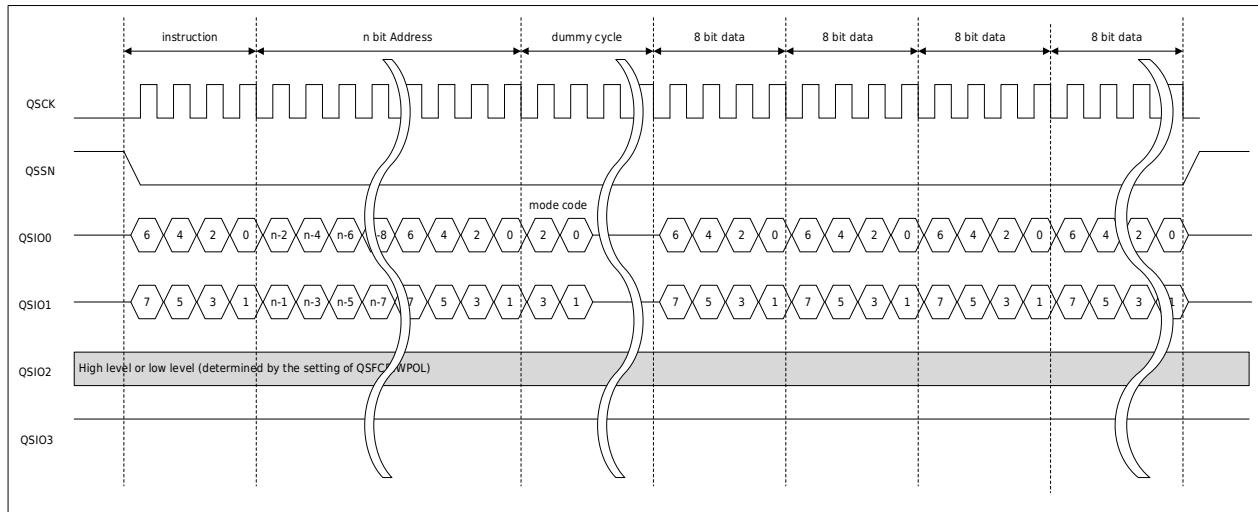


Figure 28-6 Schematic diagram of two-wire SPI protocol action (fast read mode)

The four-wire SPI protocol uses four pins QSIO0, QSIO1, QSIO2, and QSIO3 to perform all related operations such as sending instructions, addressing, and receiving data.

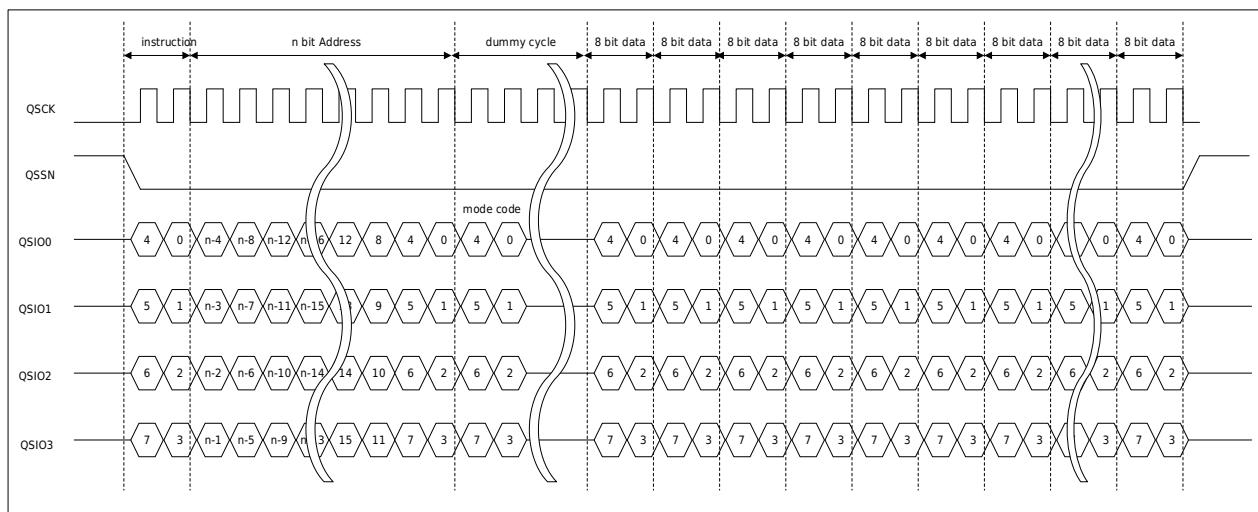


Figure 28-7 Schematic Diagram Wire SPI Protocol Action (Fast Read Mode)

28.3.2 SPI mode

There are two SPI modes, Mode 0 and Mode 3, and the mode switching can be realized by setting the SPIMD3 bit in the QSCR register. The difference between SPI mode 0 and mode 3 is that the level of QSCK is different in standby state. In SPI mode 0, the standby level of QSCK is low, while in mode 3, the standby level is high.

Serial data is output from the QSPI on the falling edge of the serial clock and read into the external flash memory on the rising edge. The external flash memory outputs serial data at the falling edge of the serial clock and is read in by QSPI at the falling edge of the next clock.

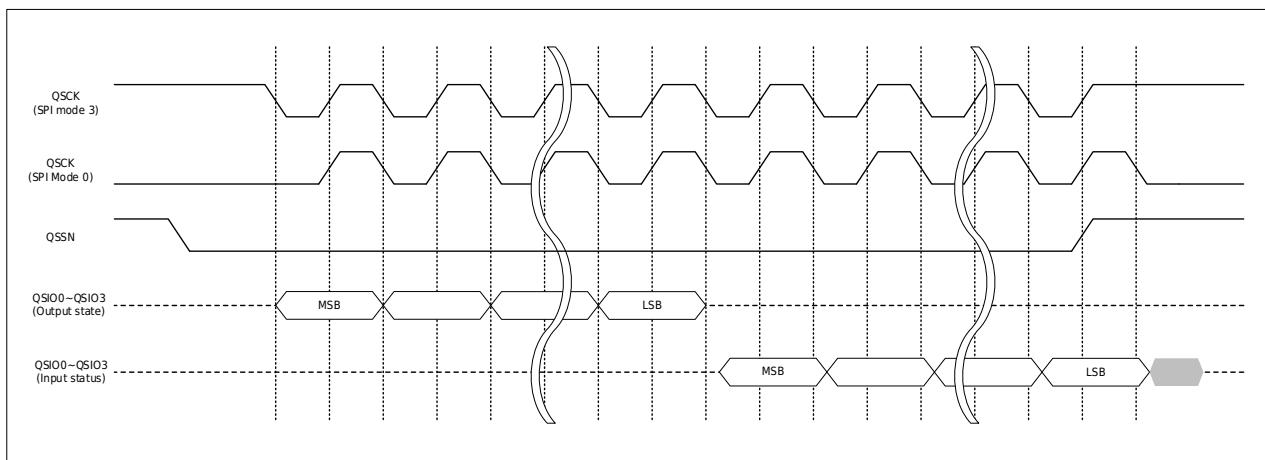


Figure 28-8 Basic Timing Diagram of the Serial Interface

28.4 Timing Adjustment of QSPI Bus

The timing of QSPI bus signals can be fine-tuned through registers, and the timing after fine-tuning can be better applied to various QSPI bus accesses, whether it is ROM access mode or direct communication mode.

28.4.1 QSPI Bus Reference Clock

The reference clock of the QSPI bus is obtained by HCLK after frequency division. By setting the DIV[5:0] bits of the QSCR register, various clock sources such as HCLK divided by 2 to 64 can be selected as the reference clock of the QSPI bus.

Table 28-3 QSPI Bus Reference Clock Selection List

DIV[5:0]	Frequency division ratio	Actual frequency	DIV[5:0]	Frequency division ratio	Actual frequency
		(HCLK=200MHz)			(HCLK=200MHz)
6'b0000000	2	100.00	6'b1000000	33	6.06
6'b0000001	2	100.00	6'b1000001	34	5.88
6'b0000010	3	66.67	6'b1000010	35	5.71
6'b0000011	4	50.00	6'b1000011	36	5.56
6'b0000100	5	40.00	6'b1000100	37	5.41
6'b0000101	6	33.33	6'b1000101	38	5.26
6'b0000110	7	28.57	6'b1000110	39	5.13
6'b0000111	8	25.00	6'b1000111	40	5.00
6'b0010000	9	22.22	6'b1010000	41	4.88
6'b0010001	10	20.00	6'b1010001	42	4.76
6'b0010010	11	18.18	6'b1010010	43	4.65
6'b0010011	12	16.67	6'b1010011	44	4.55
6'b0011000	13	15.38	6'b1011000	45	4.44
6'b0011001	14	14.29	6'b1011001	46	4.35
6'b0011100	15	13.33	6'b1011100	47	4.26
6'b0011111	16	12.50	6'b1011111	48	4.17
6'b0100000	17	11.76	6'b1100000	49	4.08
6'b0100001	18	11.11	6'b1100001	50	4.00
6'b0100010	19	10.53	6'b1100010	51	3.92
6'b0100011	20	10.00	6'b1100011	52	3.85
6'b0101000	21	9.52	6'b1101000	53	3.77
6'b0101001	22	9.09	6'b1101001	54	3.70
6'b0101010	23	8.70	6'b1101010	55	3.64
6'b0101011	24	8.33	6'b1101011	56	3.57
6'b0110000	25	8.00	6'b1110000	57	3.51
6'b0110001	26	7.69	6'b1110001	58	3.45

DIV[5:0]	Frequency division ratio	Actual frequency	DIV[5:0]	Frequency division ratio	Actual frequency
		(HCLK=200MHz)			(HCLK=200MHz)
6'b011010	27	7.41	6'b111010	59	3.39
6'b011011	28	7.14	6'b111011	60	3.33
6'b011100	29	6.90	6'b111100	61	3.28
6'b011101	30	6.67	6'b111101	62	3.23
6'b011110	31	6.45	6'b111110	63	3.17
6'b011111	32	6.25	6'b111111	64	3.13

28.4.2 SPI Bus Reference Clock

When the reference clock selects the even multiple frequency division of HCLK, the high and low level time of the QSCK signal is consistent. If the odd multiple frequency division is selected, the high level time of the QSCK signal will be one HCLK cycle longer than the low level.

If you want QSCK to output a clock signal with a duty cycle of about 50% when selecting an odd frequency division, you can set the DUTY bit in the QSFCR register to 1. Through this setting, the output time of the rising edge of the QSCK signal will be half a HCLK period later than before the adjustment, and the output time of the falling edge remains unchanged. Thus, a QSCK signal with a duty cycle of 50% can be obtained. When the base clock is divided by an even multiple of HCLK, please set the DUTY bit to 0. In the initial state, the DUTY bit defaults to 1.

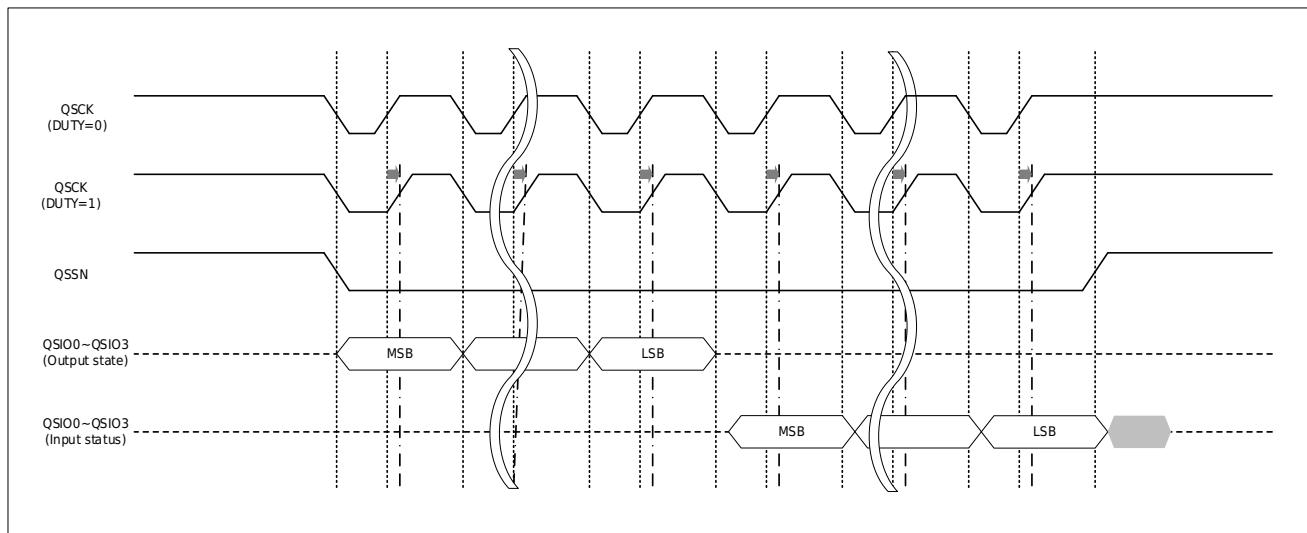


Figure 28-9 Schematic Diagram Of Output Clock Duty Ratio Correction When HCLK Is Divided By Three As The Reference Clock

28.4.3 QSSN Signal The Minimum High Level Width

In order to meet the cancellation time requirement required by the serial flash memory, the QSSN signal must maintain a high level state (that is, an idle state) for a long enough time between two adjacent QSPI bus cycles. The minimum high-level width of QSSN can be selected by setting the

SSHW[3:0] bits in the QSCSCR register, and the selection range is 1 to 16 QSPI reference clock periods.

28.4.4 QSSN of Establishment Time

The time from when the QSSN signal starts to output low level (that is, becomes active) to the first rising edge of the QSCK signal output is called the establishment time of QSSN. This time can be configured through register settings to meet the requirements of the external serial flash memory. Requirement: Set the SSNLD bit of the register QSFCR to select the QSSN setup time as 0.5 or 1.5 QSPI reference clock cycles. This setting can also be used to configure the setup time of the data pin from the data output permission to the first rising edge of the QSCK signal output, which can be reasonably used according to the needs.

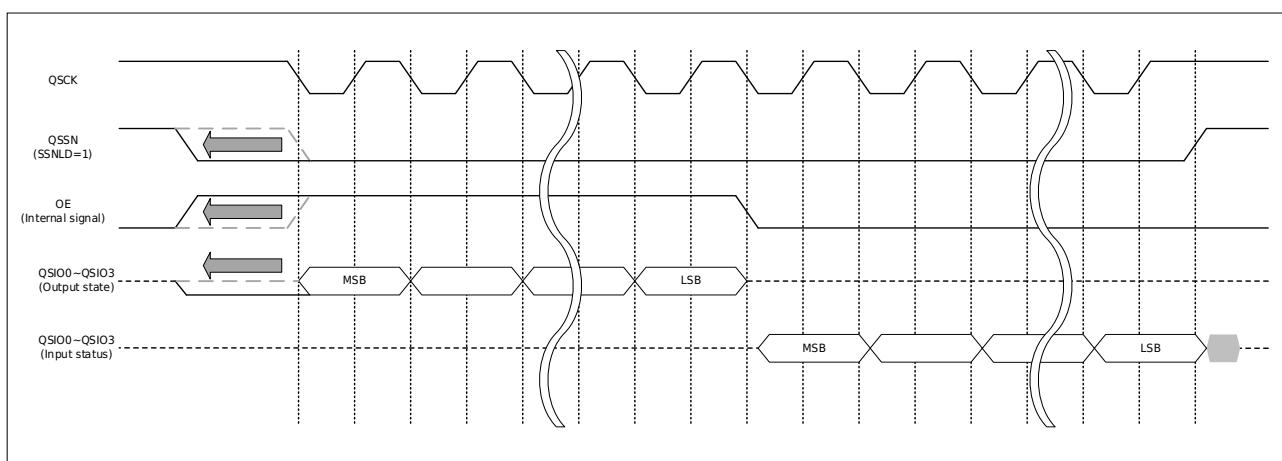


Figure 28-10 Schematic Diagram Of QSSN Establishment Time Configuration

28.4.5 QSSN of Hold Time

The time from when the QSSN signal starts to output low level (that is, becomes active) to the first rising edge of the QSCK signal output is called the establishment time of QSSN. This time can be configured through register settings to meet the requirements of the external serial flash memory. Requirement: Set the SSNHD bit of the register QSFCR to select the QSSN Hold on as 0.5 or 1.5 QSPI reference clock cycles.

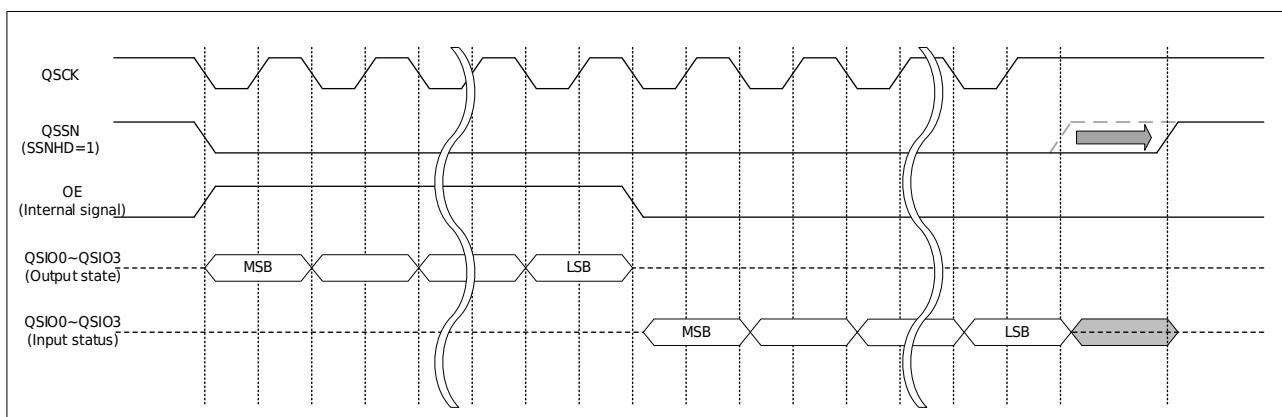


Figure 28-11 Schematic Diagram Of QSSN Hold-on Time Configuration

28.4.6 Serial Data Receive Delay

The data of the serial flash memory is output synchronously with the falling edge of QSCK, and the QSPI receives the data on the next falling edge of QSCK. The period from when the serial flash starts outputting data to when the data is received by QSPI is called receive latency. QSPI adds a delay adjustment cycle before the first data receive cycle. From the perspective of serial flash memory, this cycle can be regarded as an increase in the counterparty data receiving cycle. This delay adjustment cycle will only be generated when data is received.

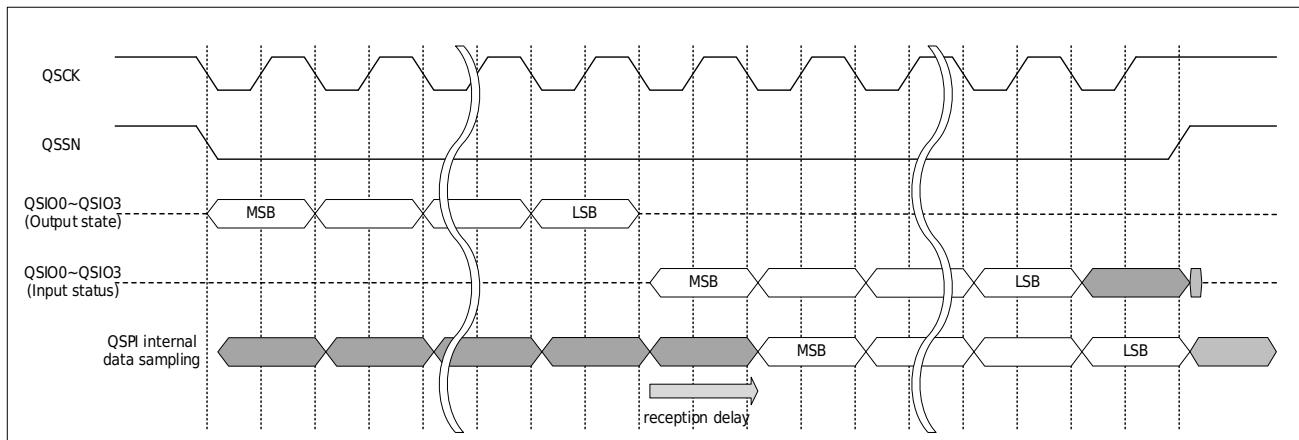


Figure 28-12 Schematic Diagram Of Data Receiving Delay

28.5 Introduction to SPI Instructions for ROM Access

28.5.1 Existing QSPI-ROM Instruction Reference

Table 28-4 List of Reference Commands

Pattern	instruction code	Note
4-byte command mode	standard read	8'h13
	fast read	8'h0C
	Two-wire output fast read	8'h3C
	Two-wire input and output fast read	8'hbC
	Four-wire output fast read	8'h6C
	Four-wire input and output fast read	8'hEC
	Exit 4-byte command mode	8'HB7
3-byte command mode	standard read	8'h03
	standard read	8'h0B
	fast read	8'h0B
	Two-wire output fast read	8'h3B
	Two-wire input and output fast read	8'hBB
	Four-wire output fast read	8'h6B
	Four-wire input and output fast read	8'hEB
	Enter 4-byte command mode	8'hE9
—	write mode	8'h06

When accessing the serial flash memory, the command needs to be set through the command register QSCCMD.

28.5.2 Standard Read

The standard read command is a common read command supported by most serial flash memory. When a serial bus cycle starts, the serial flash selection signal is set to an active state, and then QSPI outputs the instruction code (03h/13h)*1 of the instruction, and then outputs the target address. The width of the address can be passed The AWSL[1:0] bits in the QSFCR register are set. Then the data can be received. The instruction selected by the initial state of QSPI is the standard read instruction.

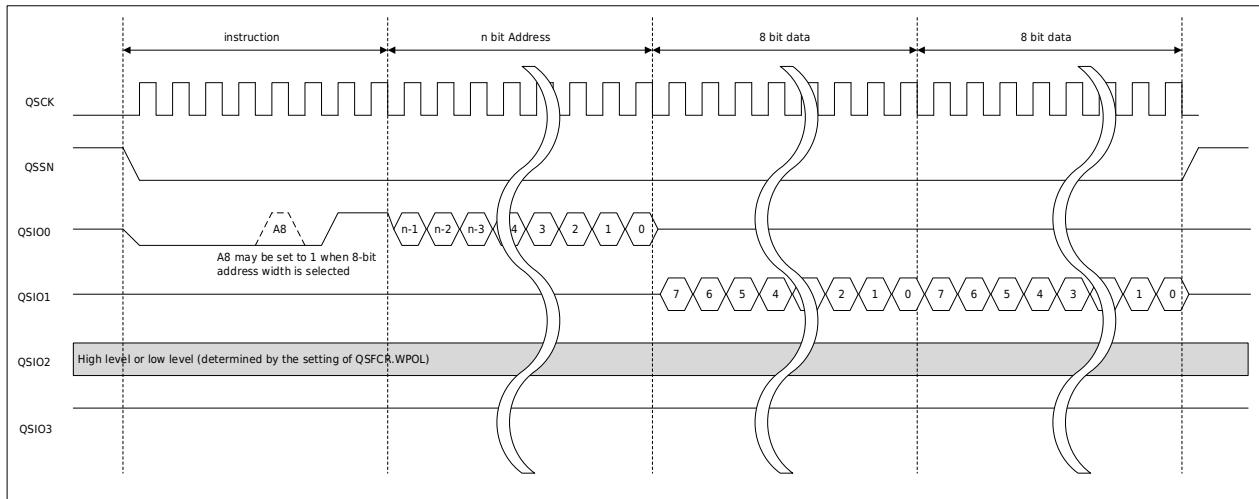


Figure 28-13 Schematic Diagram Of Standard Read Bus Cycle

28.5.3 Fast Read Command

The fast read command is a read command that supports a faster communication clock. When a serial bus cycle starts, the serial flash selection signal is set to an active state, and then QSPI outputs the instruction code (0Bh/0Ch) of the instruction, and then outputs the target address. The width of the address can be passed The AWSL[1:0] bits in the QSFCR register are set. After the address output is a certain number of dummy cycles, the specific number is determined by DMCYCN[3:0] in the QSFCR register. Next is the reception of data.

The first two cycles of the dummy cycle 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 [28.7 XIP Control].

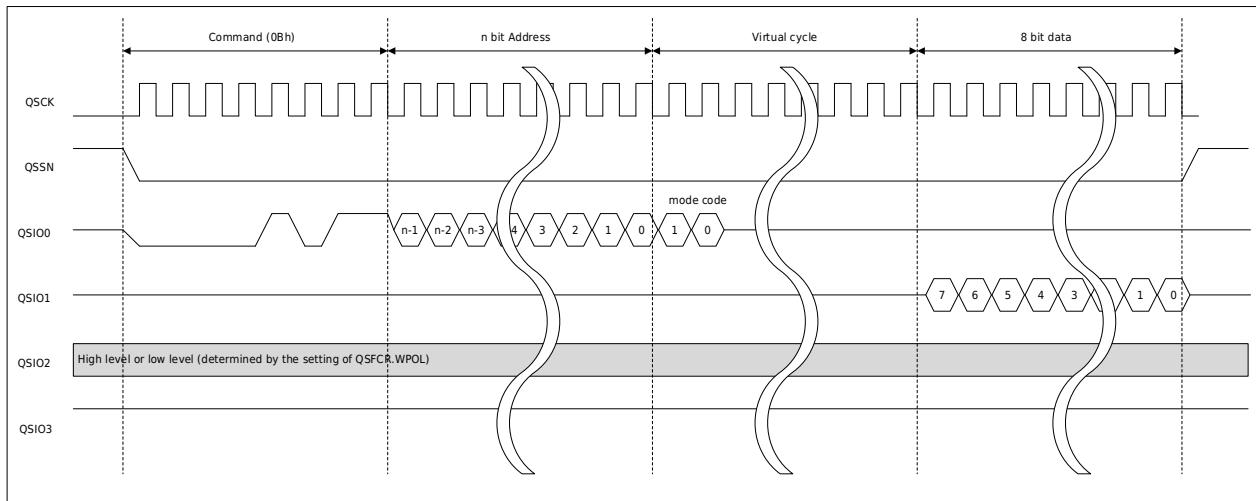


Figure 28-14 Schematic Diagram Of Fast Read Bus Cycle

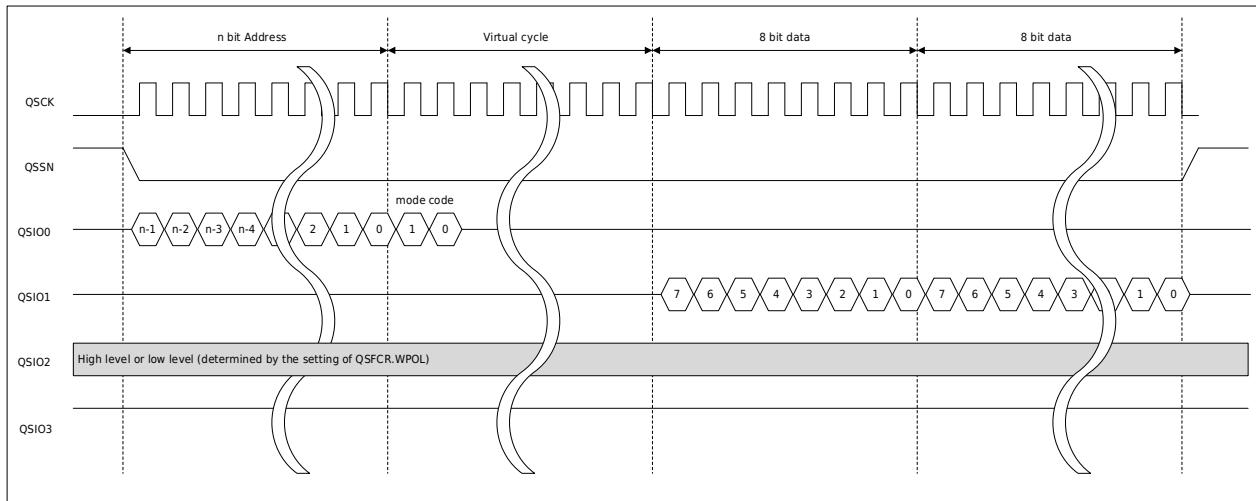


Figure 28-15 Schematic diagram of aSelect XIP Mode Four Fast Read Bus Cycle

Note:

- To use the fast read command, make sure to use a serial flash that supports the fast read function.

28.5.4 Two-Wire Output Fast Read Instruction

Two-wire output fast read is a read command that uses two signal lines for data reception. When a serial bus cycle starts, the serial flash selection signal is set to an active state, and the QSPI starts QSIO0 pins outputs the instruction code (3Bh/3Ch) of the instruction and the target address. The width of the address can be passed. The AWSL[1:0] bits in the QSFCR register are set. Then there is a certain number of dummy cycles, the specific number is determined by DMCYCN[3:0] in the QSFCR register. Then start to receive data through the two pins QSIO0 and QSIO1. Even bit data is received at QSIO0 and odd bit at QSIO1.

The first two cycles of the dummy cycle 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 [28.7 XIP Control].

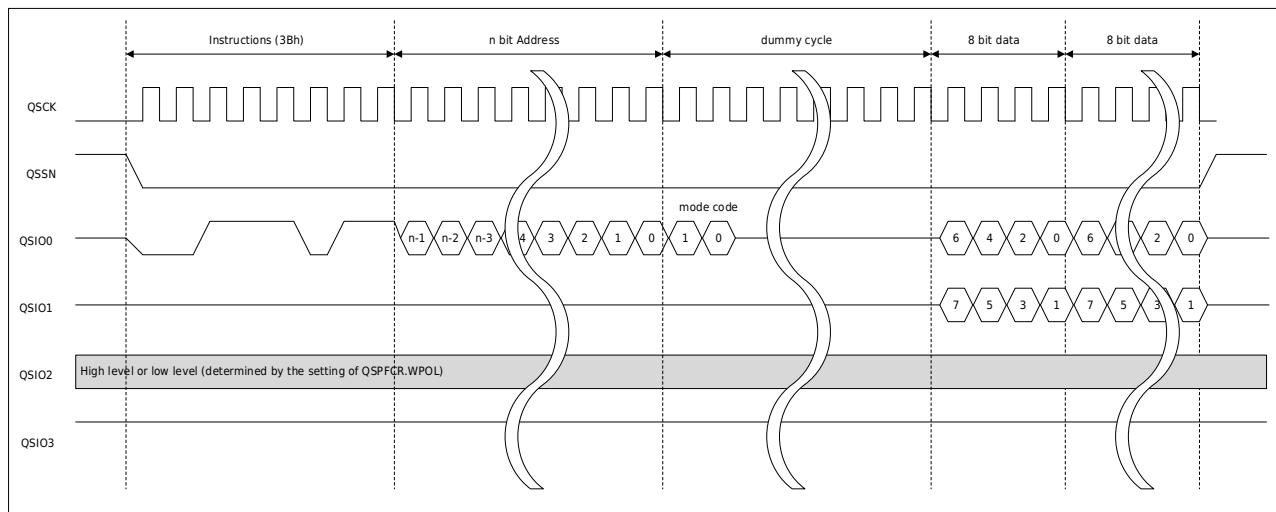


Figure 28-16 Schematic Diagram Of Fast Read Bus Cycle

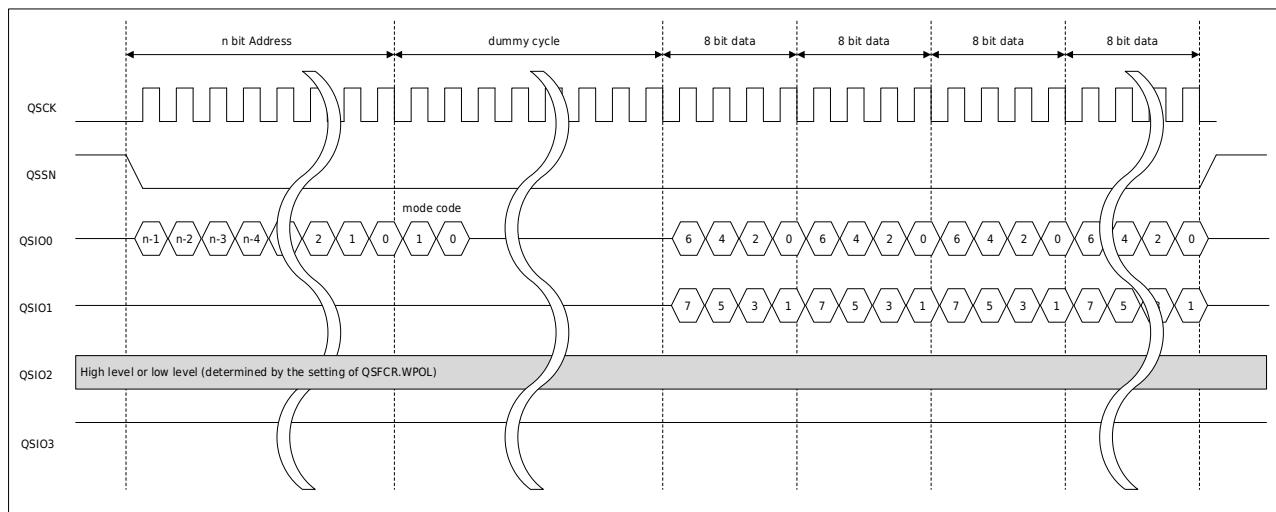


Figure 28-17 Schematic Diagram of Select XIP Mode Four Fast Read Bus Cycle

Note:

- To use the fast read command, make sure to use a serial flash that supports the fast read function.

28.5.5 Two-Wire Input Output Fast Read Instruction

Two-wire output fast read is a read command that uses two signal lines for sending and data reception. When a serial bus cycle starts, the serial flash select signal is asserted, and the QSPI starts to output the instruction code (BBh/BCh) of the instruction from the QSIO0 pin. After that, QSPI outputs the target address from the two pins QSIO0 and QSIO1, and the address width can be set by the AWSL[1:0] bits in the QSFCR register. Then start to receive data through the two pins QSIO0 and QSIO1. The transmission and data reception of even-numbered addresses and dummy cycles (including XIP mode selection information) use QSIO0 pins, and odd-numbered bits use QSIO1 pins.

The first two cycles of the dummy cycle 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 [28.7 XIP Control].

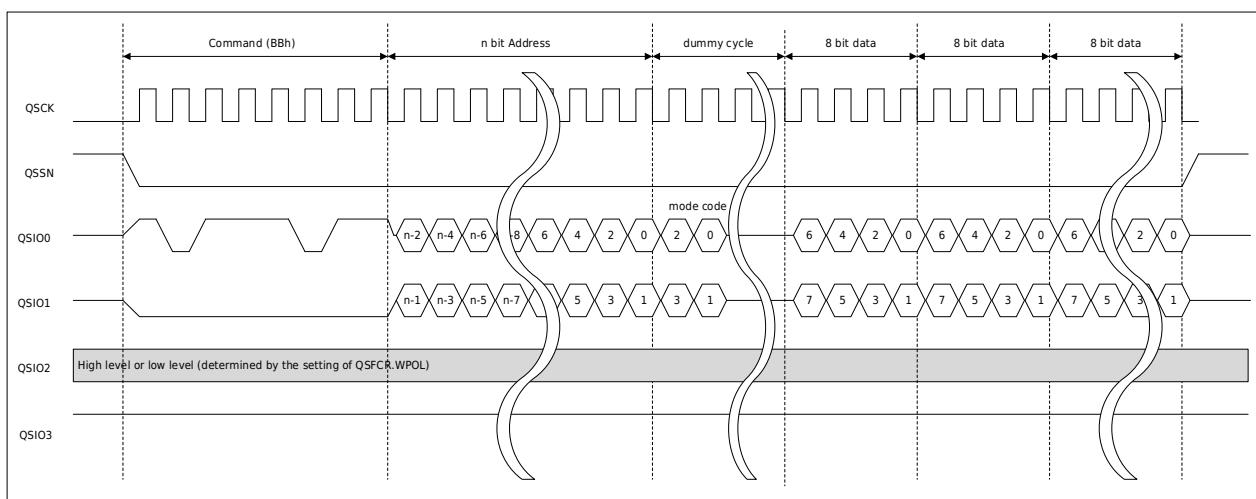


Figure 28-18 Schematic Diagram Of Fast Read Bus Cycle

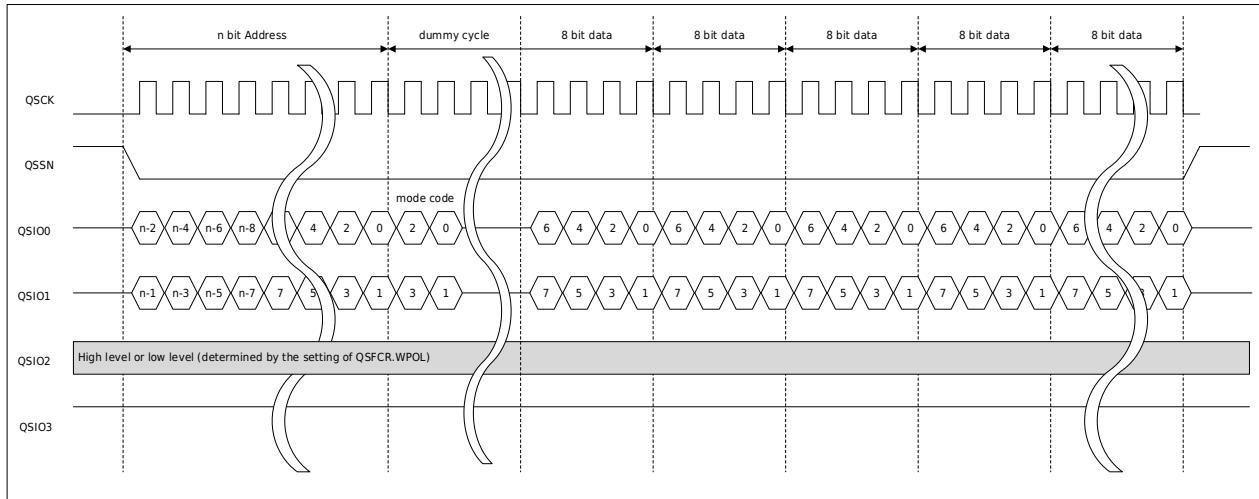


Figure 28-19 Schematic Diagram of Select XIP Mode Four Fast Read Bus Cycle

Note:

- To use the fast read command, make sure to use a serial flash that supports the fast read function.

28.5.6 Wire-Wire Output Fast Read Instruction

Four wire output fast read is a read command that uses two signal lines for data reception. When a serial bus cycle starts, the serial flash selection signal is set to an active state, and the QSPI starts QSIO0 pins outputs the instruction code (6Bh/6Ch) of the instruction and the target address. The width of the address can be passed. The AWSL[1:0] bits in the QSFCR register are set. Then there is a certain number of dummy cycles, the specific number is determined by DMCYCN[3:0] in the QSFCR register. Then start to receive data through the four pins QSIO0, QSIO1, QSIO2 and QSIO3.

The first two cycles of the dummy cycle 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 [28.7 XIP Control].

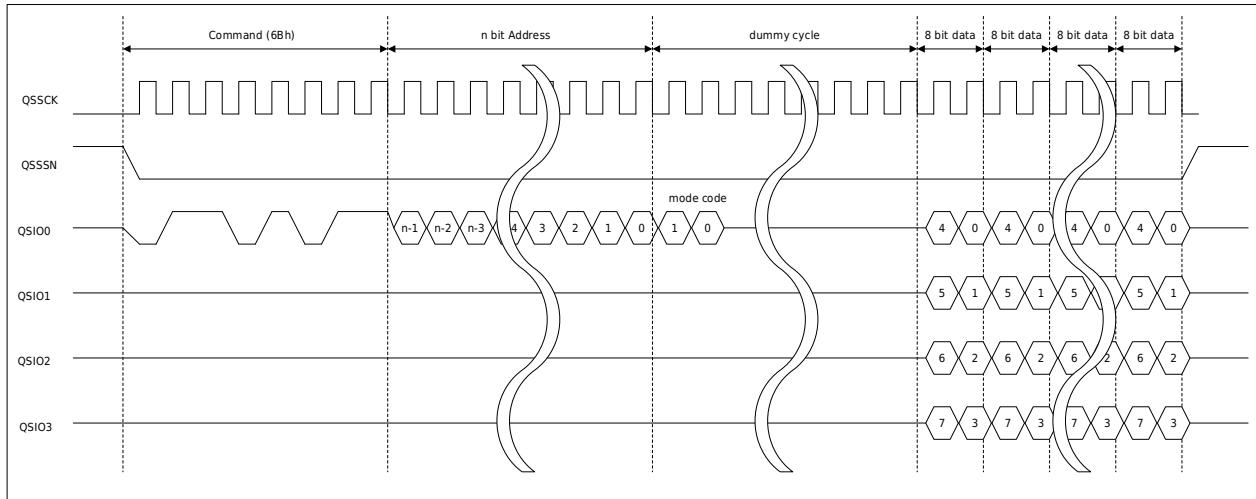


Figure 28-20 Schematic Diagram of Fast Read Bus Cycle

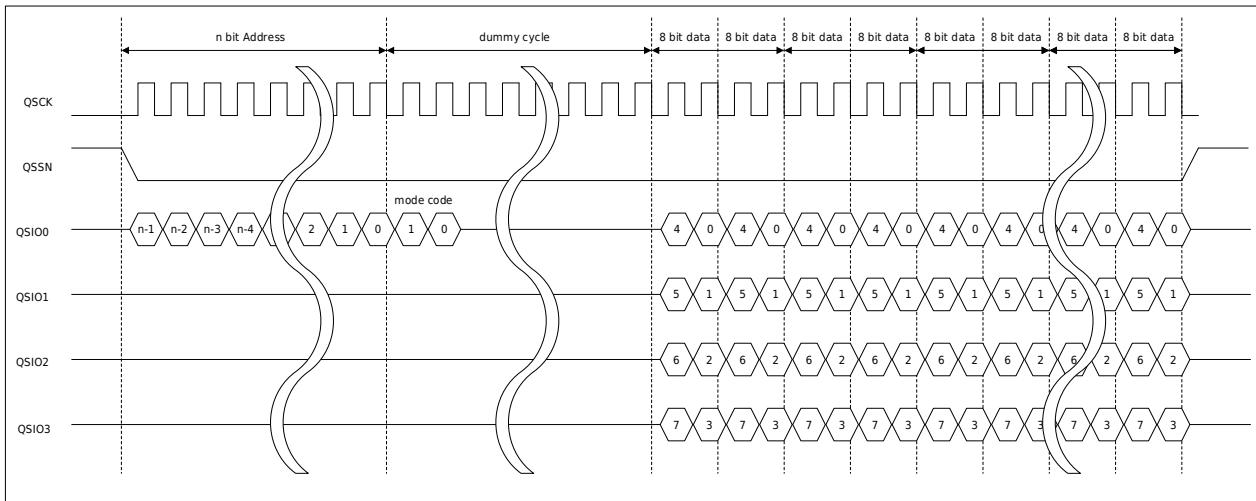


Figure 28-21 Schematic Diagram Of Select XIP Mode Four Fast Read Bus Cycle

Note:

- To use the fast read command, make sure to use a serial flash that supports the fast read function.

28.5.7 Wire-Wire Input Output Fast Read Instruction

Four wire output fast read is a read command that uses two signal lines for sending and data reception. When a serial bus cycle starts, the serial flash select signal is asserted, and the QSPI starts to output the instruction code (BBh/BCh) of the instruction from the QSI00 pin. After that, QSPI outputs the target address from the four pins QSI00, QSI01, QSI02 and QSI03, and the address width can be set by the AWSL[1:0] bits in the QSFCR register. Then start to receive data through the four pins QSI00, QSI01, QSI02 and QSI03.

The first two cycles of the dummy cycle 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 [28.7 XIP Control].

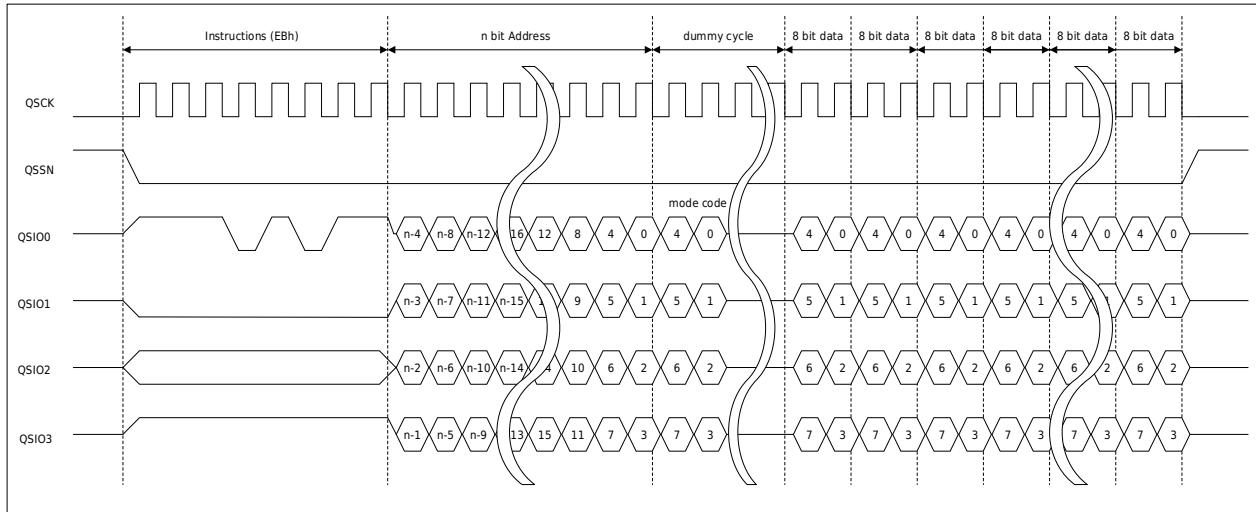


Figure 28-22 Schematic Diagram Of Fast Read Bus Cycle

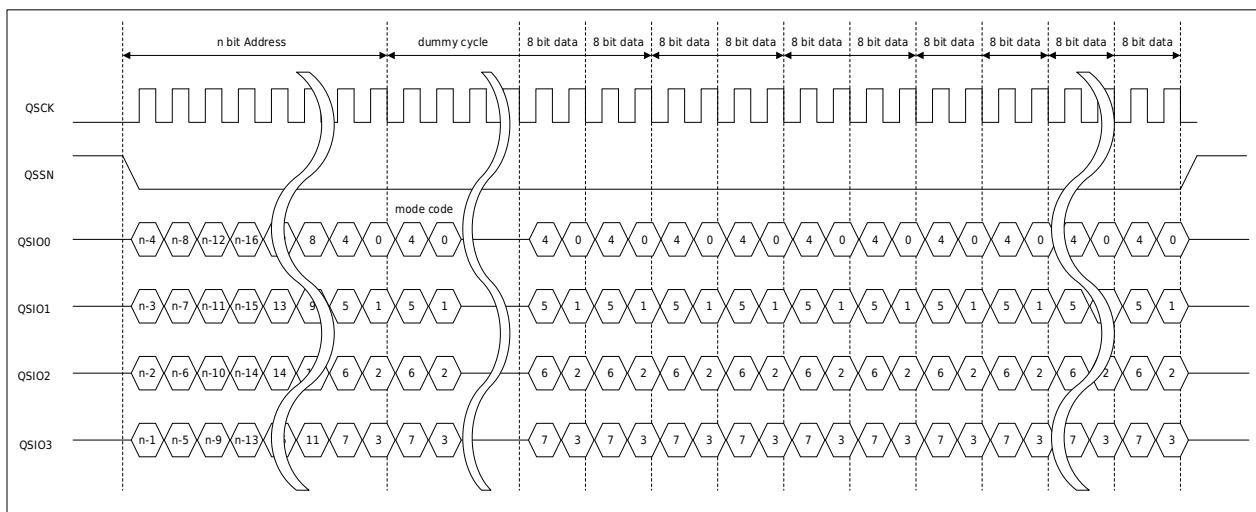


Figure 28-23 Schematic Diagram of Select XIP Mode Four Fast Read Bus Cycle

Note:

- To use the fast read command, make sure to use a serial flash that supports the fast read function.

28.5.8 Enter 4-Byte mode command

The command to enter 4-Byte mode can set the address width of the serial flash memory to 4 bytes. When a serial bus cycle starts, the serial flash select signal is asserted, and the QSPI starts outputting the instruction code (B7h) of the instruction from the QSIO0 pin.

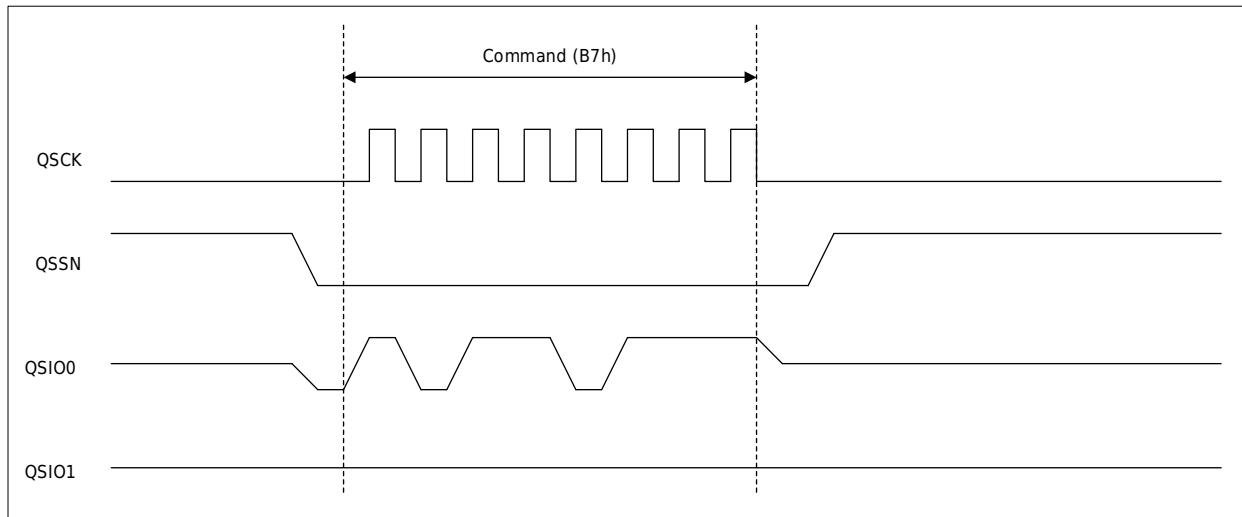


Figure 28-24 Enter 4-Byte Mode Command Bus Cycle Diagram

Note:

- This command can be issued regardless of whether the serial flash memory is in 4-Byte or 3-Byte mode.

28.5.9 Exit 4-Byte Mode Command

The command to exit 4-Byte mode can set the address width of the serial flash memory to 3 bytes. When a serial bus cycle starts, the serial flash select signal is asserted, and the QSPI starts to output the instruction code (BBh/BCh) of the instruction from the QSIO0 pin.

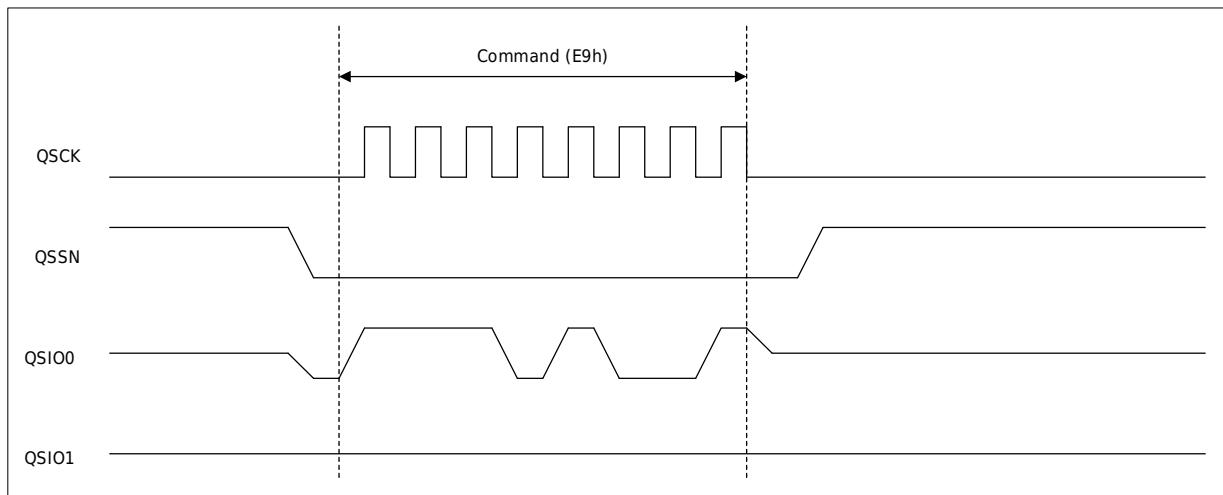


Figure 28-25 Exit 4-Byte Mode Command Bus Cycle Diagram

Note:

- This command can be issued regardless of whether the serial flash memory is in 4-Byte or 3-Byte mode.

28.5.10 Write Permission

The Write Permit command allows changing the address width of the serial flash memory. When a serial bus cycle starts, the serial flash select signal is asserted, and the QSPI starts to output the instruction code (06h) of the instruction from the QSIO0 pin.

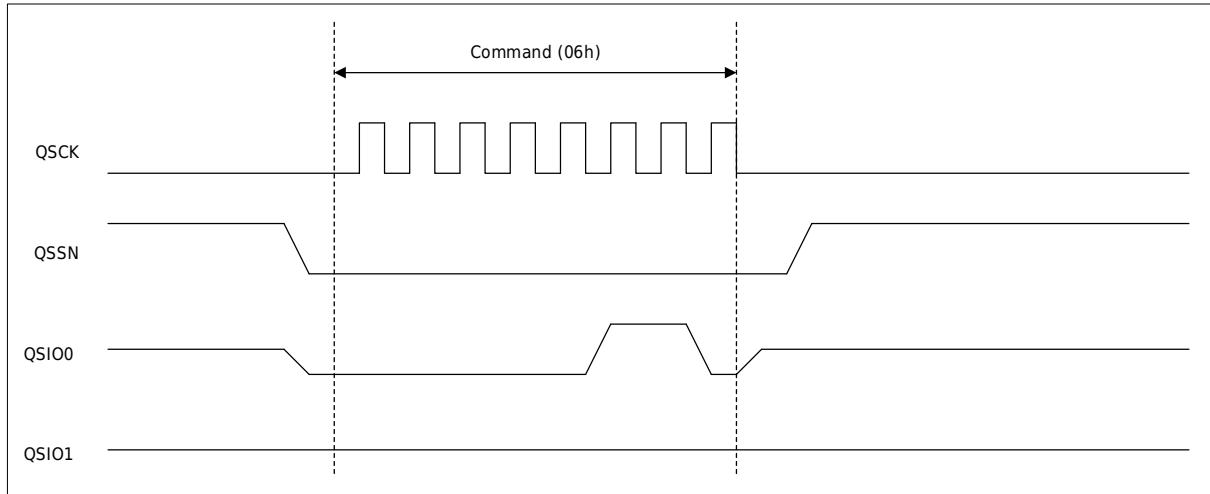


Figure 28-26 Schematic Diagram of Write Permission Command Bus Cycle

28.6 Scheduling of QSPI Bus Cycle

28.6.1 Single Flash Read with Independent Conversion

A single read command for ROM will be independently converted one-to-one from the internal bus cycle of the chip to the QSPI bus cycle. 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 memory, the QSSN signal becomes inactive, and the QSPI bus cycle is declared complete.

When another ROM read bus cycle is detected, the QSSN signal will be set to an active state again after ensuring that the invalid hold time has exceeded the minimum invalid hold width, and a new QSPI bus cycle begins.

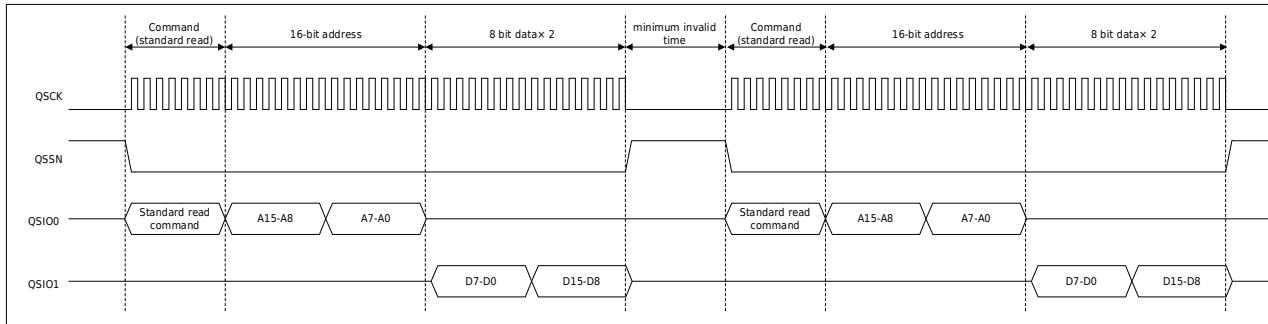


Figure 28-27 Schematic Diagram Of A Single Flash Memory Data Read Operation With Independent Conversion

28.6.2 Flash Read Using Read-Ahead

For the transfer of CPU instructions or data blocks, the system usually reads data in a sequentially increasing flash memory address sequence. Serial flash memory has continuous data transfer capability without sending instruction codes and addresses again. However, if the internal bus cycle released by the MCU is converted independently, the QSPI bus cycle is also divided into independent individuals, resulting in the inability to effectively utilize the advantages of continuous data transmission of the serial flash memory. In this regard, QSPI provides a pre-read function for continuous data reception.

The prefetch function is activated by setting the PFE bit in the QSCR register to 1. When this function is enabled, data will be continuously received and stored in the buffer without waiting for another flash read request. When the MCU issues a flash read operation, the QSPI will match the access address. If the match is successful, the data in the buffer at the corresponding position will be transmitted 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 pre-read buffer can store a maximum of 16 bytes of data. In addition, there is a 2-byte data receiving buffer that can also store pre-read data. When all the buffer data is full, The QSPI bus cycle ends. When the buffer data is read and a new buffer space is generated, QSPI will automatically start a new QSPI bus cycle to resume the pre-read action.

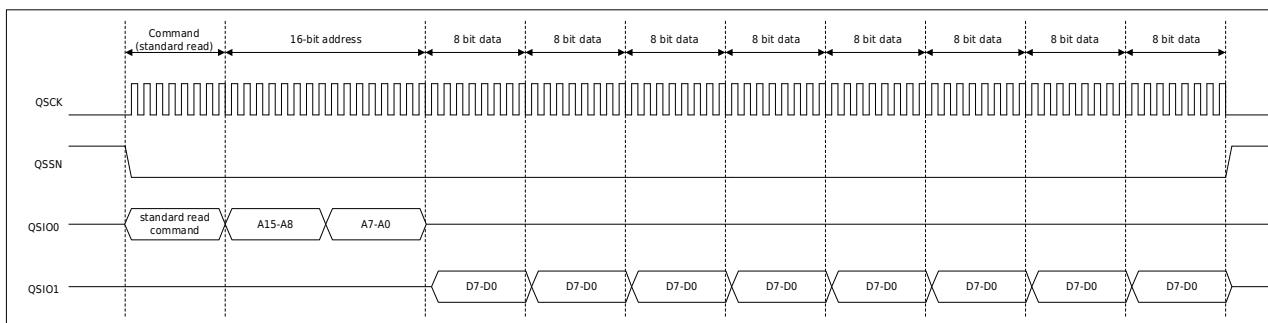


Figure 28-28 Schematic Diagram Of Data Reading Operation When The Pre-Reading Function is Valid

28.6.3 Termination of Prefetching

If a ROM read bus cycle to other addresses of the serial flash memory occurs during the pre-read transfer, the original pre-read action will be terminated and a new QSPI bus cycle will start. Normally, the pre-read operation will be terminated after the current byte transmission is completed, but if the PFSAE bit in the QSCR register is set to 1, QSPI will immediately stop the pre-read operation without waiting until the current byte transmission is completed. Use of this feature requires the serial flash device to support the instant stop action feature.

28.6.4 Prefetch Status Monitoring

Reading data from a low-speed serial flash memory will increase the load on the system, because the internal bus needs to be in a waiting state until the QSPI bus cycle of the receiving signal is completed. QSPI provides read-ahead status monitoring to reduce this load.

In the pre-read status register QSSR, the PFAN bit shows the current pre-read working status, the PFFUL bit indicates that the pre-read data buffer is full, and PFNUM[4:0] shows that the buffer has been read The number of bytes of data in the zone. By using these status bits, it is convenient to determine the current pre read status through a CPU instruction.

Note:

- When executing a read-ahead status monitoring program, place the program code outside the target serial flash area or enable instruction cache. Otherwise, the pre-reading object will frequently switch between the object data area and the instruction area, losing the meaning of pre-reading, and the monitoring program will enter an infinite loop because the pre-reading cannot be completed.

28.6.5 Flash Read using QSPI Bus Cycle Stretching

If the SSNW[1:0] in the QSCSCR register is set to a value other than 00, the QSPI bus cycle will be in a hold state after receiving the data and wait for the next data to be read. At this time, the QSSN signal will remain a low-level active state, while the QSCK is in a stopped state. If the next flash memory read instruction arrives, if the address of the read object is sequentially incremented next to the current address, QSPI will restart QSCK and directly accept data. If it is a discontinuous address, the QSSN signal will be set to a high-level inactive state to end the previously maintained QSPI bus cycle, and then start a new SPI bus cycle.

This function can prevent the system from repeatedly sending instruction codes and addresses when performing discontinuous reading of continuously increasing addresses, thereby improving reading efficiency.

The extension time of the QSPI bus cycle can be set by SSNW[1:0]. When the next reading does not occur after the set time, QSSN will be automatically set to a high-level invalid state 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 this will increase the power consumption of the serial flash memory.

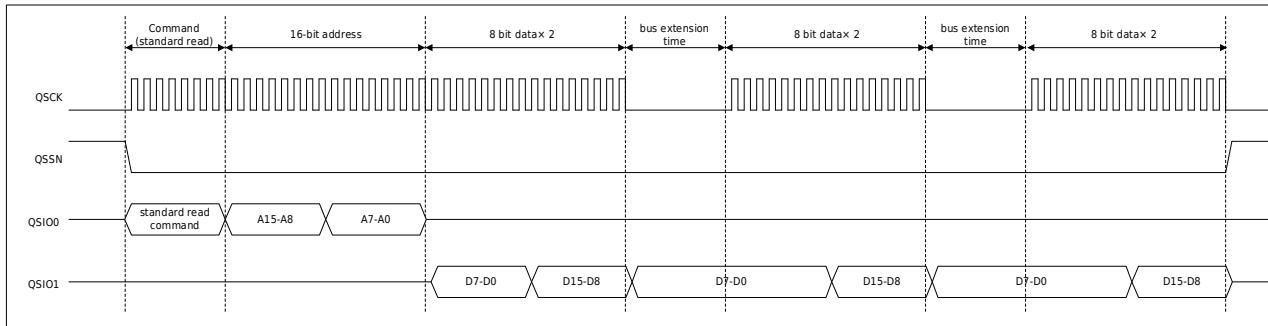


Figure 28-29 Schematic Diagram Of Data Read Operation Using QSPI Bus Cycle Extension Function

28.7 XIP Control

Some serial flash devices can reduce latency by omitting to receive a read command. This function is selected by a mode code sent during the dummy cycle.

During dummy cycles at fast speed instructions, QSPI controls the XIP mode of the serial flash by sending XIP mode codes during the first two cycles. Different serial flash memories have different XIP mode codes, which can be set specifically through the XIPMC[7:0] bits of the register QSXCMD. Specific reference below Figure28-30.

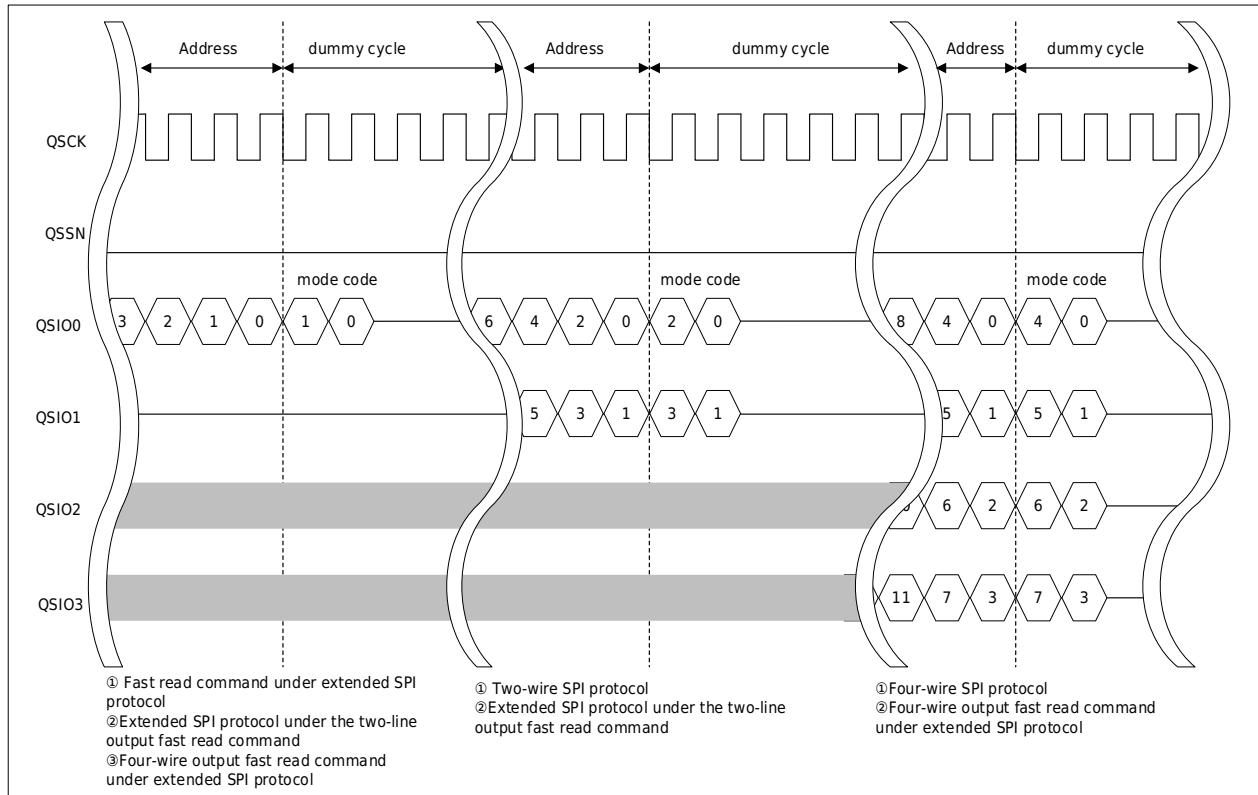


Figure 28-30 Schematic Diagram Of XIP Mode Control

28.7.1 Setting of XIP Mode

When the XIP mode code corresponding to the serial flash memory is written into XIPMC[7:0] of the register QSXCMD and the XIPE bit of the register QSCR is set to 1, when the next fast read instruction occurs, the set mode code will be in the virtual During the first two cycles of the cycle, it is sent to the target serial flash memory. After receiving the mode code, the serial flash memory and its control unit start the XIP mode. You can confirm whether you have entered XIP mode by accessing the XIPF bit of QSSR.

Note:

- To start the XIP mode of serial flash memory, you need to set the corresponding mode code in QSXCMD[7:0]. For the XIP mode of the control part, you only need to set the XIPE bit to 1, regardless of the value of QSXCMD[7:0].

28.7.2 Exit from XIP Mode

When the Exit XIP mode code corresponding to the serial flash memory is written into XIPMC[7:0] of the register QSXCMD and the XIPE bit of the register QSCR is set to 0, when the next fast read instruction occurs, the set mode code will be in the virtual During the first two cycles of the cycle, it is sent to the target serial flash memory. After receiving the mode code, the serial flash memory and its control unit of XIP mode. You can confirm whether you have Exit XIP mode by accessing the XIPF bit of QSSR.

Note:

- To Exit the XIP mode of serial flash memory, you need to set the corresponding mode code in QSXCMD[7:0]. For the XIP mode of the control part, you only need to set the XIPE bit bit is cleared, regardless of the value of QSXCMD[7:0].

28.8Pin Status of QSIO2 and QSIO3

The state of the QSIO2 and QSIO3 pins depends on the serial read mode set by the MDSEL[2:0] bits in the QSCR register.

Table 28-5 Pin Status of QIO2 and QIO3

QSCR register MDSEL[2:0] bits	QSIO2 status	QSIO3 status	Remark
000			Standard read (initial state)
001			fast read
010			Two-wire output fast read
011			Two-wire input and output fast read
100	As the third data line for input or output, the standby state is Hi-Z	As the third data line for input or output, the standby state is Hi-Z	Four-wire output fast read
101			Four-wire input and output fast read
110	Refer to the specific protocol settings for each stage	Refer to the specific protocol settings for 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.
- HOLD or QSIO3 pin can also be used as the RESET# function of the serial flash.

28.9 Direct Communication Mode

28.9.1 About Communication Mode

QSPI can read serial flash memory by automatically converting the MCU's external ROM read bus cycle to QSPI bus cycle. But the serial flash memory also has many different additional functions, such as ID information reading, erasing, writing, and status information reading. These functions do not have a set of standard instructions to set, and with the rapid increase of new functions of serial flash memory, the correspondence at the hardware level becomes more and more difficult.

In response to this situation, QSPI provides a direct communication mode, and users can directly control the serial flash memory through software. This mode software can generate any desired QSPI bus cycle.

28.9.2 Setting of Direct Communication Mode

The direct communication mode can be entered by setting the DCOME bit of the QSCR register to 1. Once it enters the direct communication mode, it will not be able to perform normal flash memory read operations. If you want to perform regular flash memory read operations, you need to clear the DCOME bit to exit the direct communication mode.

Note:

- If you are in XIP mode, you need to exit XIP mode first and then start direct communication mode.

28.9.3 QSPI Bus Cycle Generation In Direct Communication Mode

A complete QSPI bus cycle in direct communication mode starts from the first operation of DCOM[7:0] of the register QSDCOM and ends after a write operation of the QSCR register. During this period, multiple operations can be performed on DCOM[7:0]. Writing to DCOM[7:0] will be converted into a single-byte data transfer on the QSPI bus, while reading DCOM[7:0] will be converted into a single-byte data reception of the QSPI bus.

From the first operation to the DCOM[7:0] of the register QSDCOM to the last write operation to the QSCR register, the QSSN signal always maintains a low-level active state during this period.

Direct communication mode does not support multi-line actions.

Note:

- Registers other than QSCR and QSDCOM cannot be written in direct communication mode. Writing to other registers will exit direct communication mode. Exiting in this way may lead to unpredictable situations and is not recommended.

28.10 Interrupt

When a read access operation to the ROM is detected in the direct communication mode, the RAER bit of the QSSR register is set to 1, and at this time, 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.

28.11 Precautions for Use

28.11.1 Setting Sequence of QSPI Registers

The QSPI control register can be dynamically set or changed during system operation. But not paying attention to the setting order of the registers may cause the QSPI bus cycle to start before the registers are fully set, so please configure the setting order of the registers carefully to avoid such situations.

28.11.2 Module Stop Signal Setting

QSPI is in the module stop state after the system is reset, and the register can only be set after the QSPI module stop signal in the module stop control register is cleared. Please refer to [5.5 Methods of Reducing Power Consumption].

28.12 Register Description

Register Reference address: 0x9C00_0000

Table 28-6 List of QSPI Registers

Register name	Offset address	Reset value
Control register QSCR	0x0000	0x003F_0000
Chip selection Control register QSCSCR	0x0004	0x0000_000F
Format Control register QSFCR	0x0008	0x0000_8033
Status register QSSR	0x000C	0x0000_8000
Direct communication instruction register QSDCOM	0x0010	-
Instruction code register QSCCMD	0x0014	0x0000_0000
XIP Mode Code Register QSXCMD	0x0018	0x0000_00FF
Flag clear register QSSR2 (write only)	0x0024	-
External address register QSEXAR	0x0804	0x0000_0000

28.12.1 QSPI Control Register (QSCR)

Reset value: 0x003F_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16									
—	—	—	—	—	—	—	—	—	—	—	DIV[5:0]													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0									
—	—	DPRSL[1:0]	APRSL[1:0]	IPRSL[1:0]	SPIMD3	XIPE	DCOME	PFSAE	PFE	MDSEL[2:0]														

Bit	Marking	Place name	Function	Read and write
b31~b21	Reserved	—	Read as "0", write as "0"	R/W
b20~b16	DIV[5:0]	reference clock selection bit	Serial interface reference clock selection b5 b4 b3 b2 b1 b0 0 0 0 0 0 0: 2 HCLK cycles 0 0 0 0 0 1: 2 HCLK cycles* 0 0 0 0 1 0: 3 HCLK cycles 0 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 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	R/W

			0 0 0 0 0 0: 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 receiving phase	SPI protocol selection in data receiving phase. b1 b0 0 0: Extended SPI protocol 0 1: Two-wire SPI protocol 1 0: Four-wire SPI protocol 1 1: Prohibitions	R/W
b11~b10	APRSL[1:0]	SPI protocol selection in address sending phase	SPI protocol selection in address sending phase b1 b0 0 0: Extended SPI protocol 0 1: Two-wire SPI protocol 1 0: Four-wire SPI protocol 1 1: Prohibitions	R/W
b9~b8	IPRSL[1:0]	SPI protocol selection in command sending phase	SPI protocol selection instruction sending phase b1 b0 0 0: Extended SPI protocol 0 1: Two-wire SPI protocol 1 0: Four-wire 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 License	0: XIP mode Prohibition 1: XIP mode permission	R/W
b5	DCOME	Direct Communications License	QSPI bus Communication mode selection 0: ROM access mode 1: Direct Communication mode	R/W
b4	PFSAE	Read-Ahead Immediate Stop Permission	Choose where to reset the prefetch action 0: The current prefetch operation is aborted on a byte boundary 1: The current prefetch operation is aborted on a byte boundary	R/W
b3	PFE	Read-Ahead Immediate Stop Permission	Enable/disable selection of prefetch function 0: Invalid read the feature 1: The read the function is valid	R/W
b2~b0	MDSEL[2:0]	QSPI read the mode selection	Serial interface read mode selection b2 b1 b0 0 0 0: standard read 0 0 1: fast read 0 1 0: two-wire output fast read 0 1 1: Two-wire input and output fast read 1 0 0: Four-wire output fast read 1 0 1: Four-wire input and output fast read 1 1 0: custom standard read 1 1 1: Custom fast read	R/W

28.12.2 QSPI Chip selection Control Register (QSCSCR)

Reset value: 0x0000_000F

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

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R/W
b5~b4	SSNW[1:0]	QSSN effective time set	QSSN valid time extension function selection after QSPI bus access b5 b4 0 0: Does not QSSN effective time 0 1: Extend the valid time of QSSN by 32 QSCK cycles 1 0: Extend the valid time of QSSN by 128 QSCK cycles 1 1: Extend the valid time of QSSN indefinitely	R/W
b3~b0	SSHW[3:0]	QSSN minimum invalid time Setup	QSSN signal minimum invalid time selection b3 b2 b1 b0 0 0 0 0 0: 1 QSCK cycles 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 periods 1 1 0 0 0: 10 QSCK cycles 0 0 0 0 1 0: 11 QSCK cycles 1 0 1 1: 12 QSCK periods 1 1 0 0: 13 QSCK cycles 1 1 1 0 0 0: 14 QSCK cycles 1 1 1 0 0 1: 15 QSCK cycles 1 1 1 1: 16 QSCK cycles	R/W

28.12.3 QSPI Format Control Register (QSFCR)

Reset value: 0x0000_8033

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

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15	DUTY	Duty correction	QSCK output waveform duty ratio correction 0: Do not perform duty correction 1: Delay the rising edge of QSCK by 0.5 HCLK cycle (valid 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 setting	Dummy Cycles Selection When Using Fast Read Instructions b3 b2 b1 b0 0 0 0 0 0: 3 QSCK cycles* 1 0 0 0 1 0 0: 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 periods 1 1 0 0 0 0: 12 QSCK cycles 0 0 0 0 1 0: 13 QSCK cycles 1 0 1 1: 14 QSCK periods 1 1 0 0: 15 QSCK cycles 1 1 1 0 0 0: 16 QSCK cycles 1 1 1 0 0 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 (QIO2) level setting 0: low level 1: high level	WP pin (QIO2) level setting 0: low level 1: high level	R/W
b5	SSNLD	QSSN signal output time delay setting	QSSN signal Output Selection 0: Output QSSN 0.5 QSCK ahead of the first rising edge of QSCK 1: Output QSSN 1.5 QSCK ahead of the first rising edge of QSCK	R/W
b4	SSNHD	QSSN signal release time delay setting	QSSN signal release Selection 0: Release QSSN 0.5 QSCK later than the last rising edge of QSCK 1: Release QSSN 1.5 QSCK later than the last rising edge of QSCK	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2	Four_BIC	4-byte address read instruction code selection	Read instruction code selection when the address width is 4 bytes 0: Not 4-byte address read instruction code 1: Using 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 conflicts when the QIO0 terminal switches the input and output states, when the QSSMD.QSOEX bit is set to 1 (the processing permission signal is extended by 1 cycle), please select a dummy cycle of more than 4 QSPICK cycles.

28.12.4 QSPI Status Register (QSSR)

Reset value: 0x0000_8000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PFAN	PFFUL	-	PFNUM[4:0]				RAER	XIPF	-	-	-	-	-	-	BUSY

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15	PFAN	read-ahead action status	Pre-read action status signal 0: Read ahead is active 1: read the is stopped	R
b14	PFFUL	read-ahead buffer status	Pre-read buffer status signal 0: There is space left in the prefetch buffer 1: Pre-read buffer data is full	R
b13	Reserved	-	Read as "0", write as "0"	R/W
b12~b8	PFNUM[4:0]	The number of bytes of data stored in the read-ahead buffer	Display of the number of data bytes stored in the pre-read buffer b4 b3 b2 b1 b0 0 0 0 0 0: 0 byte 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 Other settings are invalid	R
b7	RAER*1	ROM access error flag	Error flag bit for ROM access in direct communication mode 0: No ROM access detected 1: ROM access 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 bit in direct communication mode 0: The bus is idle, no serial transmission in progress 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 QSSR2

28.12.5 QSPI Command Code Register (QSCCMD)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16								
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0								
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—								
—	—	—	—	—	—	—	—	—	RIC [7:0]														

Bit	Marking	Place name	Function	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 codes to replace default instructions	R/W

28.12.6 QSPI Direct Communication Command Register (QSDCOM)

Reset value: none

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16								
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0								
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—								
—	—	—	—	—	—	—	—	—	DCOM[7:0]														

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b0	DCOM[7:0]	Direct communication mode command	The interface in the direct communication mode performs direct communication through the QSPI bus. Read and write accesses to this interface are translated into a corresponding QSPI bus cycle. This interface is only valid in direct communication mode, and it is forbidden to access in ROM access mode.	R/W

28.12.7 QSPI XIP Mode Code Register (QSXCMD)

Reset value: 0x0000_00FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16								
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0								
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—								
—	—	—	—	—	—	—	—	—	XIPMC [7:0]														

Bit	Marking	Place name	Function	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

28.12.8 QSPI System Configuration Register (QSSR2)

Reset value: none

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

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Write "0" on write	W
b7	RAERCLR	RAERclear	Clears the RAER bit in the QSSR when writing a 1	W
b6~b0	Reserved	-	Write "0" on write	W

28.12.9 QSPI External Extended Address Register (QSEXAR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EXADR[5:0]							-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit	Marking	Place name	Function	Read and write
b31~b26	EXADR[5:0]	External Extended Address Code	The upper 6 bits of the QSPI external address are set, and the ROM access window address of the QSPI can access a maximum of 64MB×63 blocks of external ROM space	R/W
b25~b0	Reserved	-	Read as "0", write as "0"	R/W

29 Integrated Circuit Built-in Audio Bus Module (I2S)

29.1 Introduction

I2S (Inter_IC Sound Bus), integrated circuit built-in audio bus, which is dedicated to data transmission between audio devices.

Table 29-1 I2S Main Features

Function	Main Features
Communication mode	<ul style="list-style-type: none">Support full-duplex and half-duplex communicationSupport host mode or slave mode operation
Data format	<ul style="list-style-type: none">Optional channel length: 16/32 bitsOptional transmission data length: 16/24/32 bitsData shift order: MSB start
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 driving 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 StandardLSB Alignment StandardPCM standard
data buffer	<ul style="list-style-type: none">With 2-word deep, 32-bit wide input and output FIFO buffer area
Timer	<ul style="list-style-type: none">Can use internal I2SCLK (UPLL/R/UPLLQ/UPLL/PPLL/R/PPLLQ/PPLL); can also be supplied by external clock on I2S_EXCK pin
Interrupt	<ul style="list-style-type: none">An interrupt is generated when the effective space of the send buffer reaches the alarm thresholdAn interrupt is generated when the effective space of the receive buffer reaches the alarm thresholdThe receiving data area is full and there is still a request to write data, and the receiving overflowsThe sending data area is empty and there are still sending requests, sending underflowThe sending data area is full and there is still a request to write data, sending overflow

29.2 I2S System Block Diagram

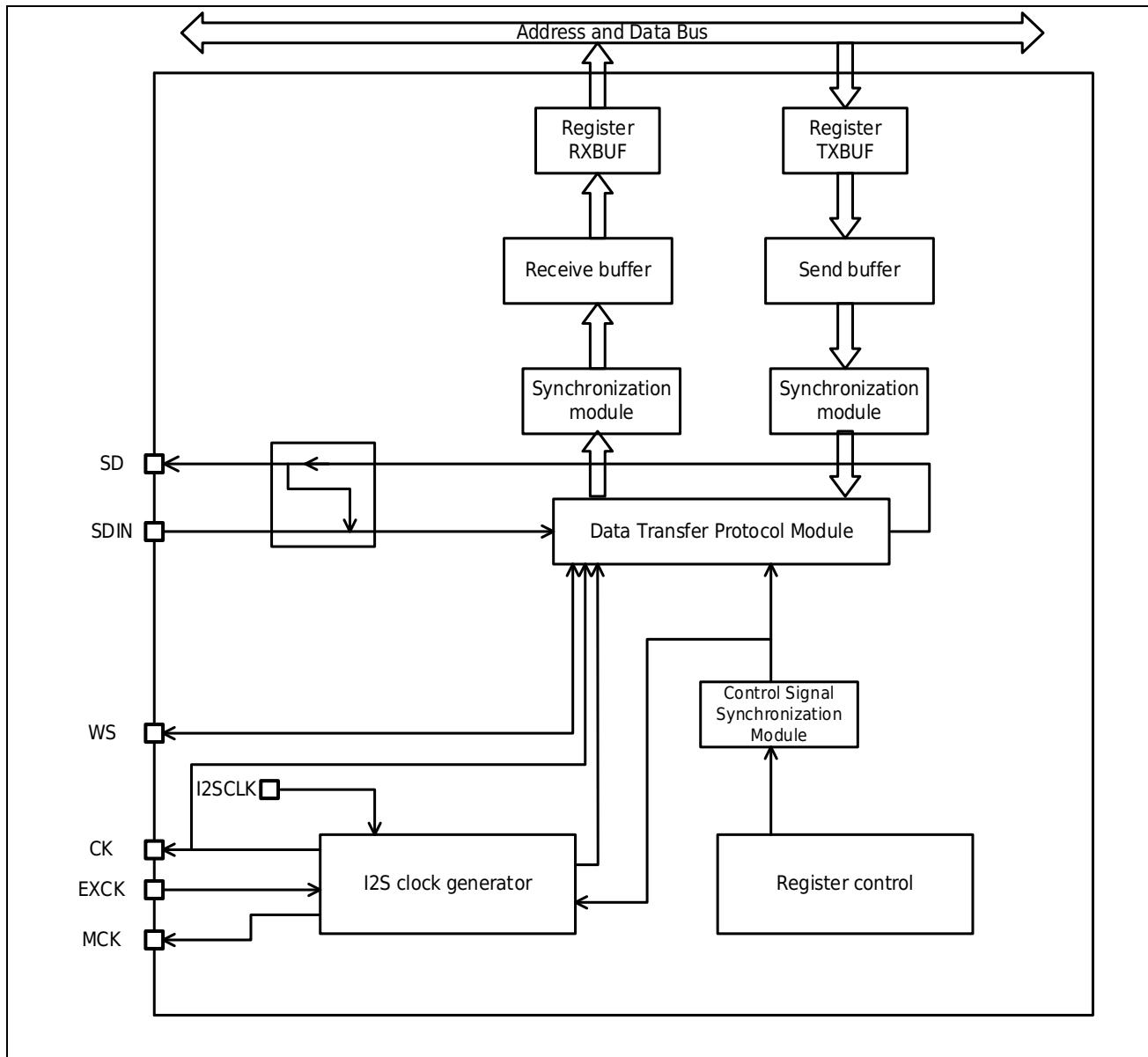


Figure 29-1 I2S System Block Diagram

29.3 Pin Description

Table 29-2 I2S Pin Description

Pin name	Direction	Functional description
I2Sn_CK	Input output	Communication clock
I2Sn_WS	Input output	word selection
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

29.4 Functional Description

This chapter will describe the function of I2S in detail.

29.4.1 I2S General Description

I2S pin function

- I2Sn_SD: Serial data, for half-duplex mode data input, or half/full-duplex mode data output.
- I2Sn_WS: Word selection, which is the data control signal output in the master mode and the data control signal input in the slave mode.
- I2Sn_CK: Serial clock, which is the serial clock output in master mode and the serial clock input in slave mode.
- I2Sn_EXCK: External clock source, the clock generator in master mode selects the external clock as the frequency division clock source.
- I2Sn_SDIN: A 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.

29.4.2 Communication Mode

It supports half-duplex and full-duplex communication modes, which can be selected through the DUPLEX bit of the I2S control register (I2S_CTRL).

When the I2S_CTRL.DUPLEX bit is 0, I2S operates in the half-duplex communication mode, uses the I2Sn_SD pin as the output data pin when only sending, and uses 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.

29.4.3 Supported Audio Protocols

There are four data and frame format combinations to send data in the following formats:

- Pack 16-bit data in 16-bit frames
- Pack 16-bit data in 32-bit frames
- Pack 24-bit data in 32-bit frames
- Pack 32-bit data in 32-bit frames

When using 16 bits of data in a 32-bit packet, the first 16 bits (MSB) are valid and the 16 LSB are forced to zero without any software action (just one read/write operation).

When using 24 bits of data in a 32-bit packet, the first 24 bits (MSB) are valid and the 8 LSBs are forced to zero without any software action (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.

29.4.3.1 I2S Philips Standard

Use the WS signal to indicate which channel the data currently being sent belongs to. This signal is valid one clock before the first bit (MSB) of the current channel's data.

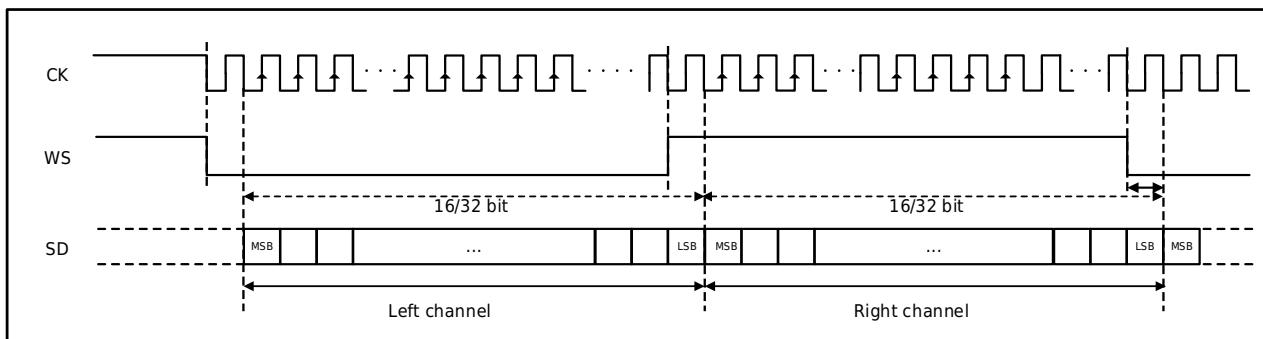


Figure 29-2 I2S Philips protocol waveform (16/32 bit full precision)

16 bits are loaded in a 16-bit frame. In transmit mode, write to I2S_TXBUF register 0xFFFF_3344, SD outputs serial data 0x3344; in receive mode, SD inputs serial data 0xEEEE, and I2S_RXBUF register reads data 0x0000_EEEE.

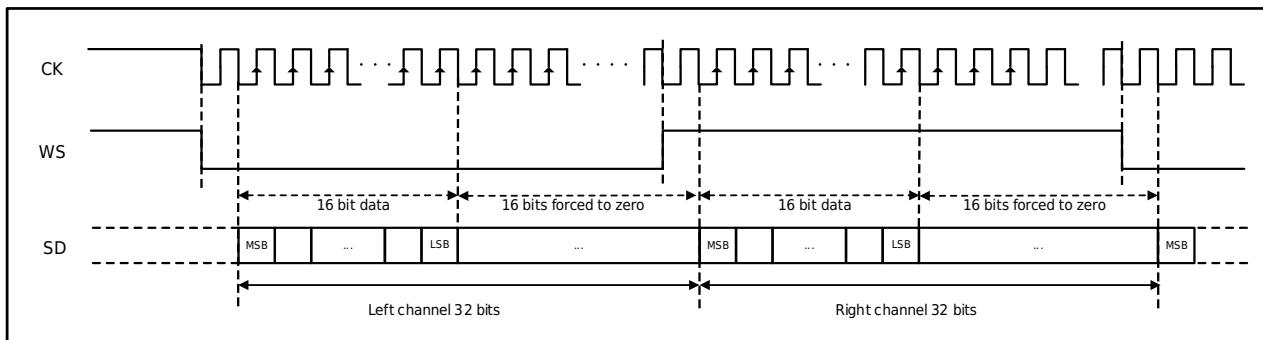


Figure 29-3 I2S Philips Protocol Waveform (16-Bit Data Encapsulated In 32-Bit Frame)

16 bits are loaded in a 32-bit frame. In the sending mode, write to the I2S_TXBUF register 0xFFFF_3344, and the SD outputs the serial data 0x3344_0000; in the receiving mode, the SD inputs the serial data 0xEEEE_XXXX, and the I2S_RXBUF register reads the data 0x0000_EEEE.

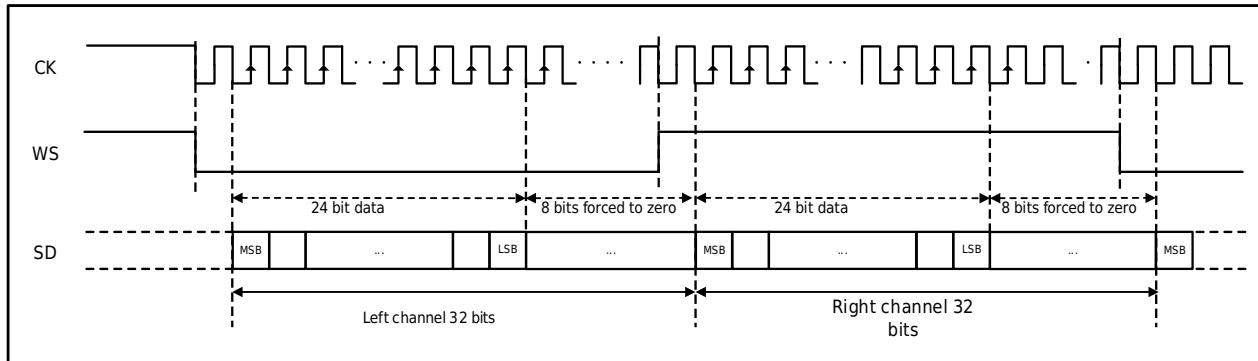


Figure 29-4 I2S Philips Protocol Waveform (24-Bit Data Encapsulated in 32-Bit Frame)

24 bits are loaded in a 32-bit frame. In the sending mode, write to the I2S_TXBUF register 0xXX22_3344, and the SD outputs the serial data 0x2233_4400; in the receiving mode, the SD inputs the serial data 0xEEDD_11XX, and the I2S_RXBUF register reads the data 0x00EE_DD11.

29.4.3.2 MSB Alignment Standard

This standard generates both the WS signal and the first data bit (ie MSBit).

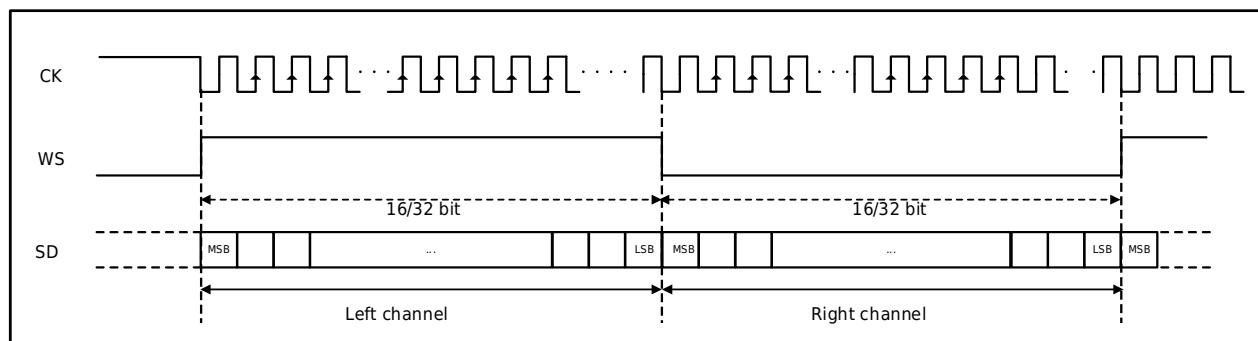


Figure 29-5 I2S MSB Protocol Waveform (16/32 Bit Full Precision)

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 0xFFFF_3344, SD outputs serial data 0x3344; in receive mode, SD inputs serial data 0xEEDD, and I2S_RXBUF register reads data 0x0000_EEDD.

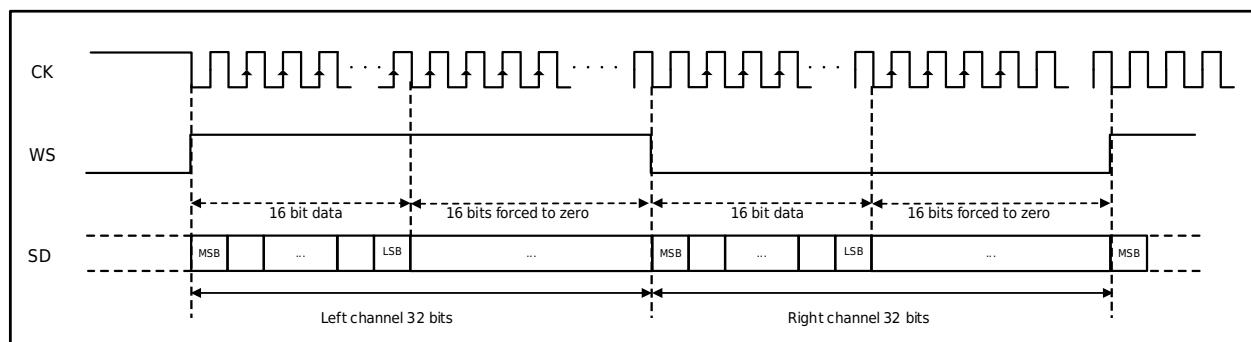


Figure 29-6 I2S MSB Protocol Waveform (16-Bit Data Encapsulated In 32-Bit Frame)

16 bits are loaded in a 32-bit frame. In the sending mode, write to the I2S_TXBUF register 0xXXXX_3344, and the SD outputs the serial data 0x3344_0000; in the receiving mode, the SD inputs the serial data 0xEEDD_XXXX, and the I2S_RXBUF register reads the data 0x0000_EEDD.

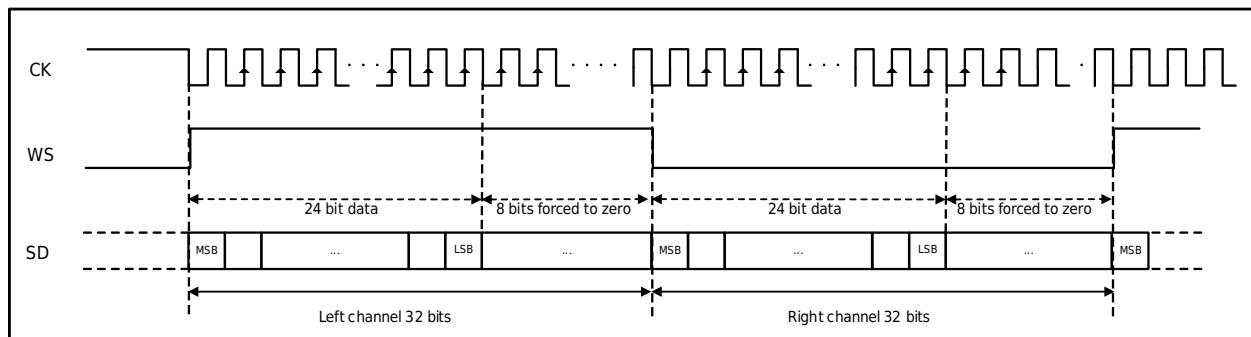


Figure 29-7 I2S MSB Protocol Waveform (24-Bit Data Encapsulated In 32-Bit Frame)

24 bits are loaded in a 32-bit frame. In the sending mode, write to the I2S_TXBUF register 0xXX22_3344, and the SD outputs the serial data 0x2233_4400; in the receiving mode, the SD inputs the serial data 0xEEDD_11XX, and the I2S_RXBUF register reads the data 0x00EE_DD11.

29.4.3.3 LSB Alignment Standard

This standard is similar to the MSB alignment standard (for 16-bit and 32-bit full-precision frame formats, there is no difference).

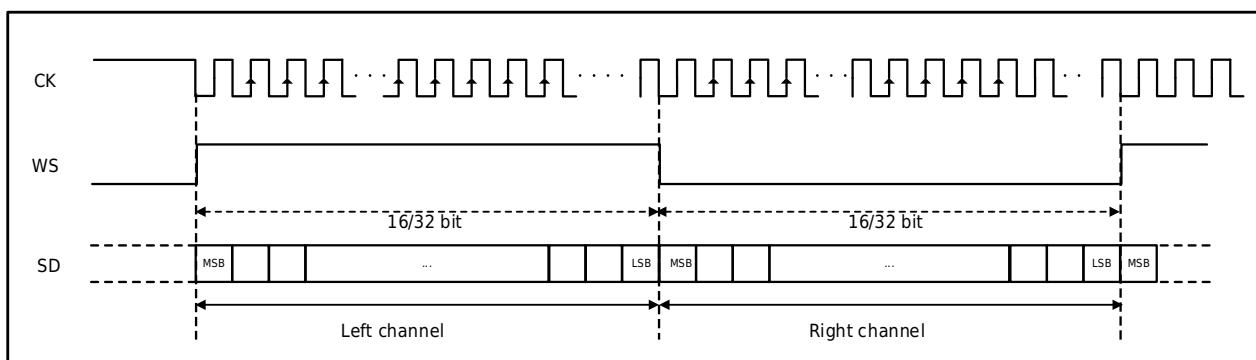


Figure 29-8 I2S LSB Protocol Waveform (16/32 Bit Full Precision)

16 bits are loaded in a 16-bit frame. In transmit mode, write to I2S_TXBUF register 0xFFFF_3344, SD outputs serial data 0x3344; in receive mode, SD inputs serial data 0xEEDD, and I2S_RXBUF register reads data 0x0000_EEDD.

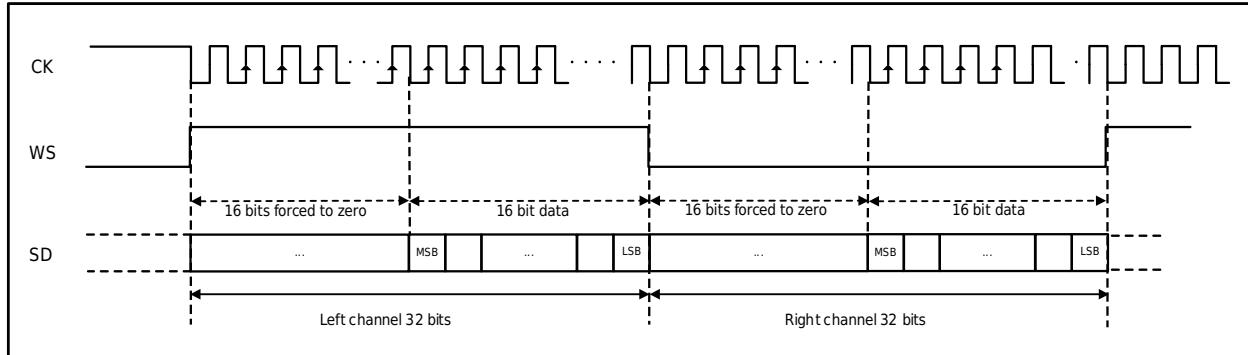


Figure 29-9 I2S LSB Protocol Waveform (16-Bit Data Encapsulated In 32-Bit Frame)

16 bits are loaded in a 32-bit frame. In transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD outputs serial data 0x0000_3344; in receive mode, SD inputs serial data 0xXXXX_1122, and I2S_RXBUF register reads data 0x0000_1122.

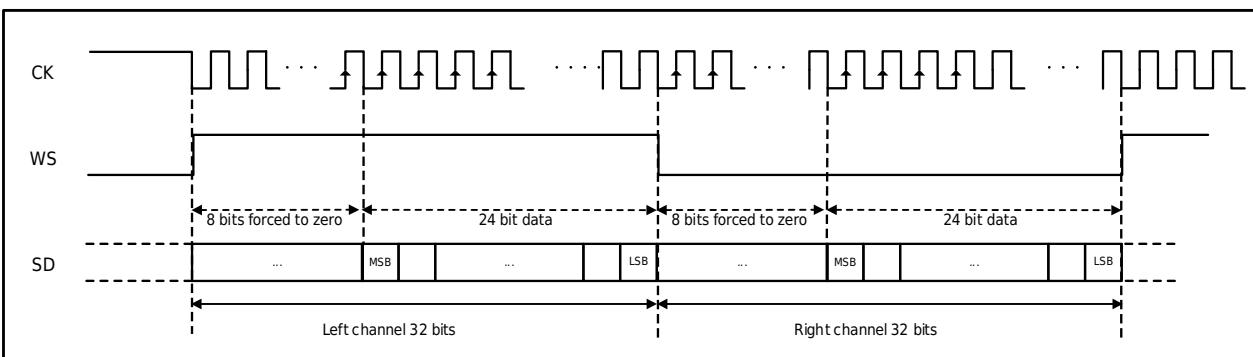


Figure 29-10 I2S LSB Protocol Waveform (24-Bit Data Encapsulated In 32-Bit Frame)

24 bits are loaded in a 32-bit frame. In the sending mode, write to the I2S_TXBUF register 0xXX22_3344, and the SD outputs the serial data 0x0022_3344; in the receiving mode, the SD inputs the serial data 0xXXDD_1122, and the I2S_RXBUF register reads the data 0x00DD_1122.

29.4.3.4 PCM Standard

For the PCM standard, there is no need to use channel information. There are two PCM modes (Short Frame and Long Frame) and are configurable using the PCMSYNC bit in I2S_CFGR.

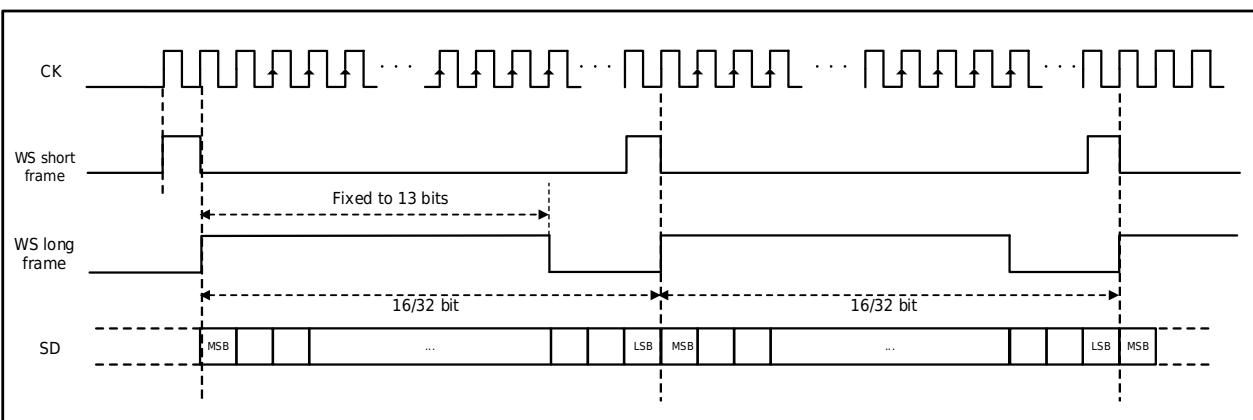


Figure 29-11 I2S PCM Protocol Waveform (16/32 Bit Full Precision)

For long frame sync, the WS signal is maintained 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 0xXXXX_3344, SD outputs serial data 0x3344; in receive mode, SD inputs serial data 0xEEDD, and I2S_RXBUF register reads data 0x0000_EEDD.

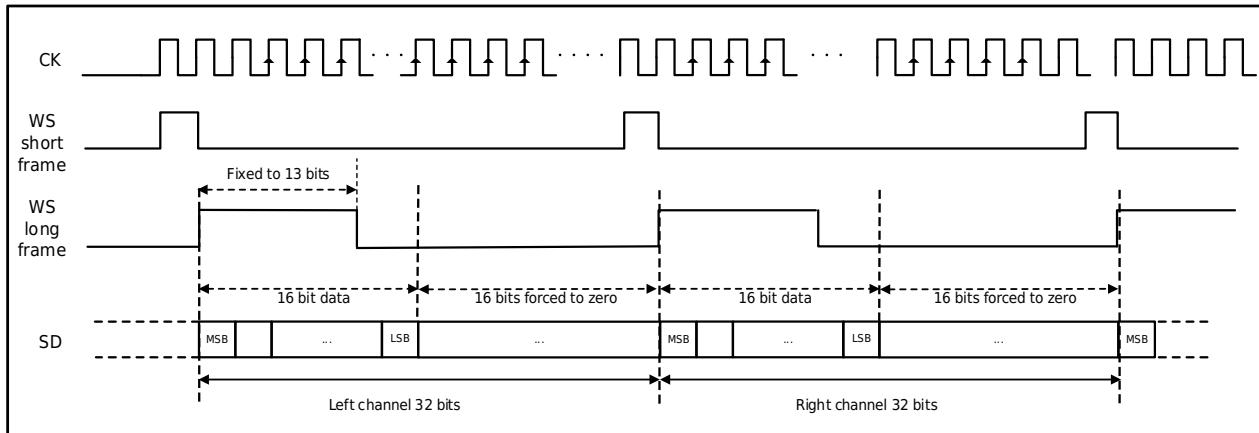


Figure 29-12 I2S PCM Protocol Waveform (16 Bit Data Encapsulated in 32-Bit Frames)

16 bits are loaded in a 32-bit frame. In transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD outputs serial data 0x0000_3344; in receive mode, SD inputs serial data 0xEEDD_XXXX, and I2S_RXBUF register reads data 0x0000_EEDD.

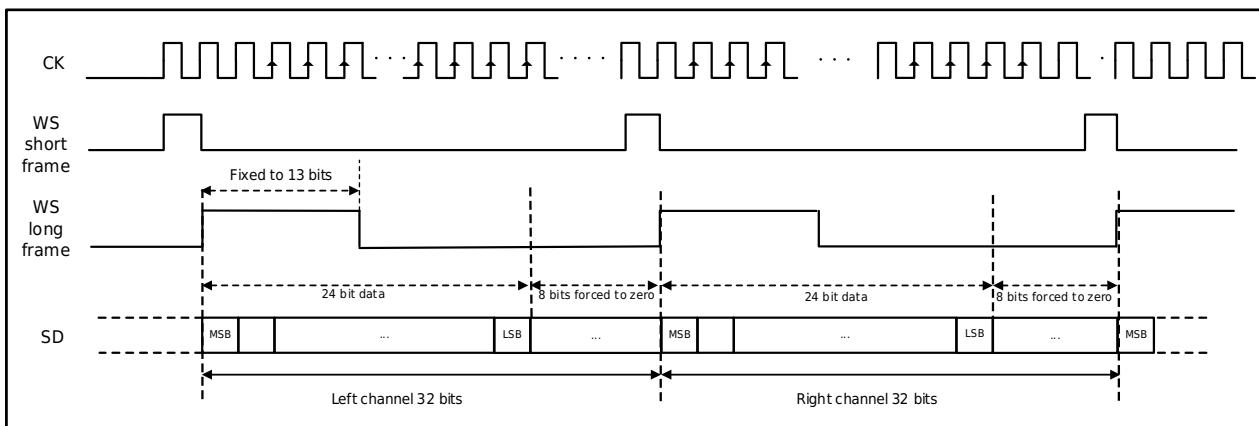


Figure 29-13 I2S PCM Protocol Waveform (24-Bit Data Encapsulated in 32-Bit Frame)

24 bits are loaded in a 32-bit frame. In the sending mode, write to the I2S_TXBUF register 0xXX22_3344, and the SD outputs the serial data 0x0022_3344; in the receiving mode, the SD inputs the serial data 0xEEDD_11XX, and the I2S_RXBUF register reads the data 0x00EE_DD11.

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).

29.4.4 Clock Generator

The I2S bit rate is used to determine the data flow on the I2S data line and the I2S clock signal frequency.

I2S bit rate = number of bits per channel × number of channels × audio sampling frequency

For 16-bit dual-channel audio, the calculation formula of I2S bit rate is as follows: I2S bit rate = $16 \times 2 \times F_s$. If the packet is 32 bits wide, then the I2S bitrate = $32 \times 2 \times F_s$.

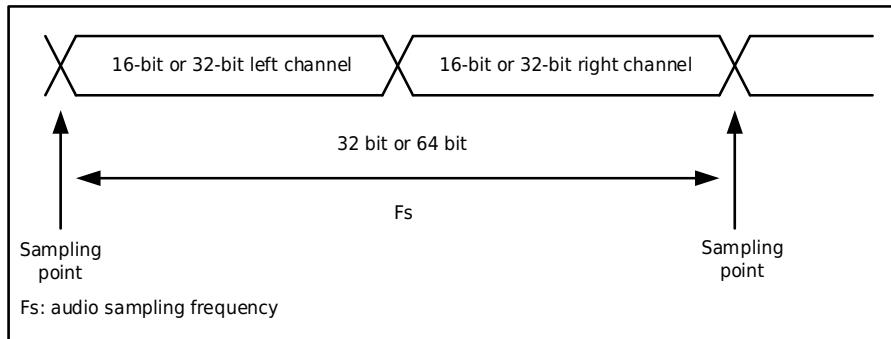


Figure 29-14 Audio Sampling Frequency Definition

When configuring master mode, the linear crossover needs to be set up correctly to communicate at the desired audio frequency.

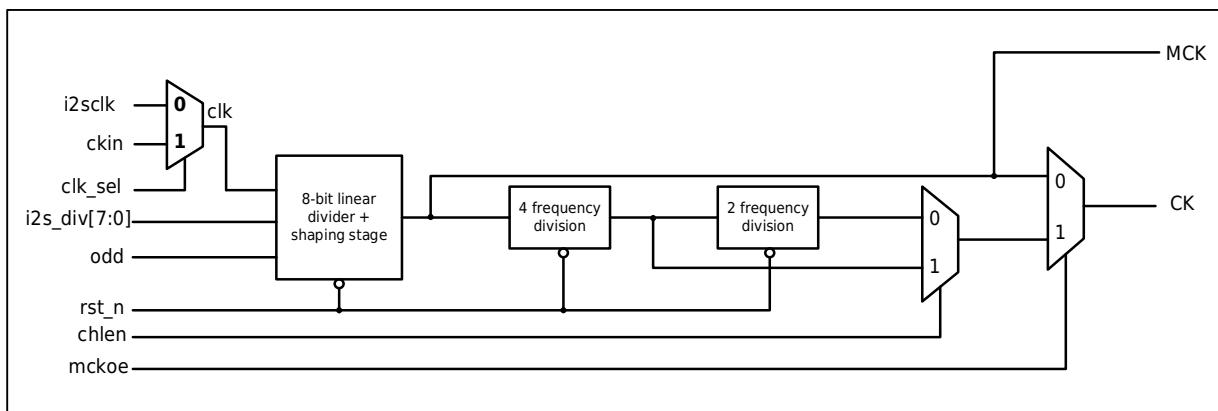


Figure 29-15 Clock Generator Architecture

The clk clock source can be I2SCLK output or external clock.

The 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 drive clock (mckoe is set to 1):

$$F_s = \frac{clk}{[(16 \times 2) \times ((2 \times I2SDIV + ODD) \times 8)]} \quad (\text{when the channel frame width is 16 bits})$$

$$F_s = \frac{clk}{[(32 \times 2) \times ((2 \times I2SDIV + ODD) \times 4)]} \quad (\text{when the channel frame width is 32 bits})$$

When the drive clock is turned off (mckoe is cleared):

$$F_s = \frac{clk}{[(16 \times 2) \times (2 \times I2SDIV + ODD)]} \quad (\text{when the channel frame width is 16 bits})$$

$F_s = \text{clk} / [(32 \times 2) \times (2 \times I2SDIV + ODD)]$ (when the channel frame width is 32 bits)

The table below provides example accuracy values for different clock configurations. There are other configurations that achieve better clock accuracy.

Table 29-3 Audio Frequency Accuracy (for VCO input frequency=1MHz)

Drive clock	Target fS(Hz)	Data format	Multiplication factor	Output division ratio	I2SDIV	ODD	Real-time fS(Hz)	Error
Output off	8000	16-bit	288	3	187	1	8000	0.0000%
		32-bit	256	4	62	1	8000	0.0000%
	16000	16-bit	256	4	62	1	16000	0.0000%
		32-bit	256	2	62	1	16000	0.0000%
	32000	16-bit	256	2	62	1	32000	0.0000%
		32-bit	256	5	12	1	32000	0.0000%
	48000	16-bit	384	10	12	1	48000	0.0000%
		32-bit	384	5	12	1	48000	0.0000%
	96000	16-bit	384	5	12	1	96000	0.0000%
		32-bit	424	3	11	1	96014.49219	0.0151%
	22050	16-bit	290	3	68	1	22049.87695	0.0006%
		32-bit	302	2	53	1	22050.23438	0.0011%
	44100	16-bit	302	2	53	1	44100.46875	0.0011%
		32-bit	429	4	19	0	44099.50781	0.0011%
	192000	16-bit	424	3	11	1	192028.9844	0.0151%
		32-bit	258	3	3	1	191964.2813	0.0186%
Output enable	8000	无关	256	5	12	1	8000	0.0000%
	16000	无关	426	4	13	0	16000.60059	0.0038%
	32000	无关	426	4	6	1	32001.20117	0.0038%
	48000	无关	258	3	3	1	47991.07031	0.0186%
	96000	无关	344	2	3	1	95982.14063	0.0186%
	22050	无关	429	4	9	1	22049.75291	0.0011%
	44100	无关	271	2	6	0	44108.07422	0.0183%

Note:

- The frequency multiplication factor can be UPLL/N/MPLL/N, when the frequency multiplication factor is UPLL/N, the output frequency division ratio can be UPLL/P/UPLL/Q/UPLL/R, and when the frequency multiplication factor is MPLL/N, the output frequency division ratio can be MPLL/P/MPLL/Q/MPLL/R. For specific settings, refer to the CMU UPLL configuration register and CMU MPLL configuration register of Clock Controller (CMU).

29.4.5 I2S Host Mode

I2S can be configured as follows:

- Transmit master or Receive master (half-duplex mode using I2S)
- Simultaneous transceiver master (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. You can choose to output or not output the drive clock (MCK) by setting the MCKOE bit of the register I2S_CTRL.

Step

1. Set the pins that I2S needs to use.
2. Select the clock source through 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 for proper audio sampling frequency.
4. Set the I2S configuration register (I2S_CFGR), select the I2S standard through the I2SSSTD[1:0] and PCMSYNC bits, select the data length through the DATLEN[1:0] bits and select the number of bits per channel by configuring the CHLEN bits.
5. If you need to use interrupt, set the interrupt register of the system.
6. If you need to use DMA, set the relevant register for 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 each interrupt writes two data, first turn off the transmit interrupt enable flag TXIE after the interrupt starts, and then turn on TXIE after writing the data. 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 the data is written to the transmit buffer. A complete frame means that the left channel data is sent first and then the right channel data is sent. There are no partial frames where only the left channel is sent. During the first bit transmission, the data is loaded into the shift register in parallel, then shifted in serial and output to the SD pin (MSB first). See 29.4.3Supported Audio Protocols for more details on write operations in various I2S standard modes.

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, and the RXE bit is set to 1 to allow reception. See 29.4.3 Supported Audio Protocols for more details on read operations in various I2S standard modes. If new data is received while previously received data has not been read, an overflow error will be generated and the I2S_ER.RXERR flag will be set. If the I2S_CTRL.EIE bit is set, an interrupt will be generated to indicate the error.

29.4.6 I2S Slave Mode

I2S can be configured as follows:

- Transmit slave or Receive slave (half-duplex mode using I2S)
- Simultaneous transmit and receive slaves (full-duplex mode using I2S).

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 device connected to the I2S interface. This way, the user does not need to configure the clock.

Step

1. Set the pins that I2S needs to use.
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 by configuring the CHLEN bits.
3. If you need to use interrupt, set the interrupt register of the system.
4. If you need to use DMA, set the relevant register for DMA.
5. To send data, 1~4 data to be sent should be pre-written 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 select I2S host 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, TXE is set, and the external master transmits the clock and requests data transfer via the WS signal. When communication starts, data is transferred from the transmit buffer to the shift register. During first bit transmission, data is loaded from the internal bus into the shift register in parallel, then shifted in serial and output to the SD pin (MSB first). Every time the transmit buffer FIFO space is larger than the set threshold, if the I2S_CTRL.TXIE

bit is set, an interrupt will be generated. See 29.4.3Supported Audio Protocols for more details on write operations in various I2S standard modes.

In order to ensure continuous audio data transmission, the next data to be sent must be written into the TX FIFO before the end of the current data transmission. If the first clock edge of the next data communication arrives when the data has not 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, and the RXE bit is set to 1 to allow reception. See 29.4.3Supported Audio Protocols for more details on read operations in various I2S standard modes. If new data is received while previously received data has not been read, a receive overflow error will be generated and the RXERR flag will be set. If the I2S_CTRL.EIE bit is set, an interrupt will be generated to indicate the error.

29.4.7 I2S Interrupt

The interrupt sources of I2S include sending buffer effective space greater than the alarm threshold, receiving buffer effective space less than the alarm threshold, receiving overflow, sending underflow and sending overflow. The transmit underflow and transmit overflow are integrated into the I2S transmit error interrupt TXERR, so it is necessary to judge the actual interrupt source through the flag. The specific description of the I2S interrupt source is shown in Table 29-4. 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 29-4 is the list of I2S interrupts.

Table 29-4 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 effective space of the receive buffer is less than the alarm threshold	RXBA	RXIE
The receiving data area is full and there is still a request to write data, and the receiving overflows	RXERR	EIE
The sending data area is empty and there are still sending requests, sending underflow	TXERR	EIE
The sending data area is full and there is still a request to write data, sending overflow	TXERR	EIE

29.4.8 Precautions for Use

29.4.8.1 Precautions when using as a host

1) When I2S acts as the master to send data only, if all the data in I2S_TXBUF has been sent and no new data is written, I2S will pause the action after the last data is sent, and I2S will no longer generate communication at this time clock, the transmit error flag bit TXERR will be set to 1. At this time, the user can choose to write the I2S_CTRL.TXE bit as 0 to close the I2S, or write new sending data to I2S_TXBUF to continue sending. When continuing to send, WS will restart from the left channel (Philips, MSB/LSB mode).

If the I2S_CTRL.TXE bit is directly written as 0 during the sending action, the I2S will be turned off immediately, and the current data sending will be terminated. This approach will lead to the inability to grasp the state of the slave, and the slave will receive data confusion when restarting communication 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 the paused state.

2) When I2S is used as the host to only receive data, if you want to temporarily stop the receiving action, you can write two frames of dummy data in advance, and set I2S_CTRL.TXE to 1 when you need to pause and count to the corresponding position. When the baud rate is 8k~96k, advance 4 pieces of data TXE are set to 1, and when the baud rate is 192k, 5 pieces of data TXE are set to 1 in advance. After the two frames of dummy data are sent, I2S will pause. At this time, I2S will no longer generate a 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 turn off TXE when restarting to receive the first data after the 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 receive-only state. Please refer to the figure below for details.

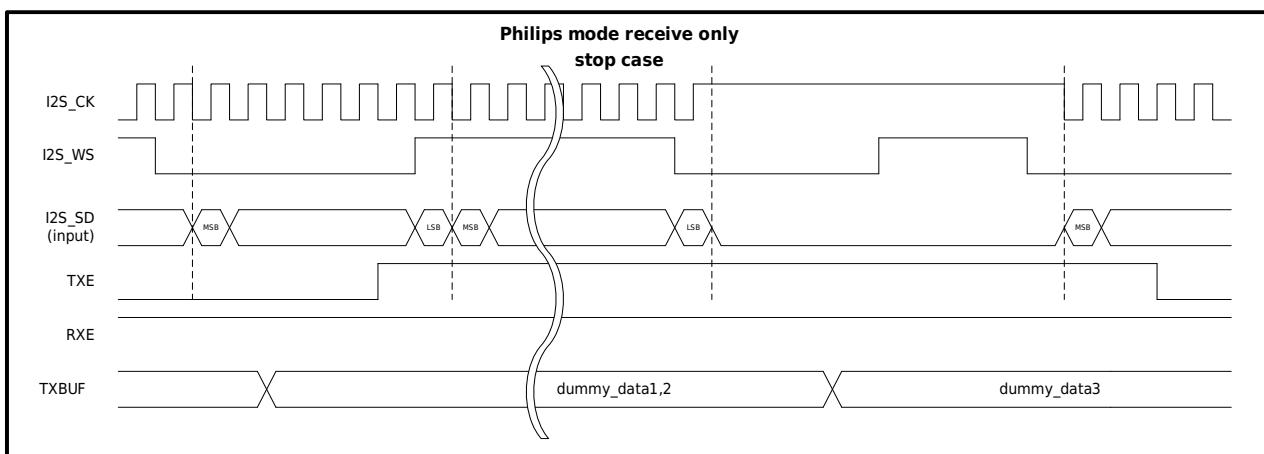


Figure 29-16 The Host Only Receives And Temporarily Stops Receiving

If the I2S_CTRL.RXE bit is directly written to 0 during the receiving action, the I2S will be

turned off immediately, and the current data reception will be terminated. This approach will cause the status of the slave to be unpredictable, 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 when the I2S is in a paused state. to turn off I2S.

3) When I2S acts as the master in full-duplex operation, if all the 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 bit 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 sending data to I2S_TXBUF to continue sending and receiving operations. When continuing to send, WS will restart from the left channel (Philips, MSB/LSB mode).

If the I2S_CTRL.TXE and I2S_CTRL.RXE bits are directly written as 0 during the full-duplex operation, the I2S will be turned off immediately, and the current data sending and receiving will be terminated. This approach will cause the status of the slave to be unclear, and the communication data will be confused when restarting the communication 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 the paused 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 status of WS every time it receives data, then after I2S pauses, you need to set I2S_CTRL.TXE to 0 first, then write new send data to I2S_TXBUF and then set I2S_CTRL.TXE to 1 to restart the transmission. The specific actions are shown in the figure below.

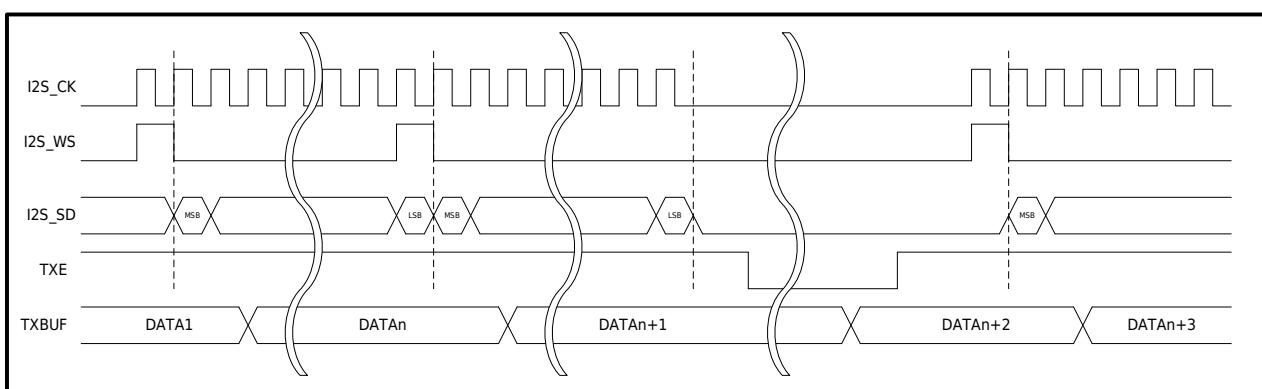


Figure 29-17 PCM Short Frame Host Resends After The Suspension Of Transmission Method 1

If the slave only detects the WS state when receiving the first frame of data, after the I2S pauses, it can directly write new sending data to I2S_TXBUF to restart the transmission. The specific actions are shown in the figure below.

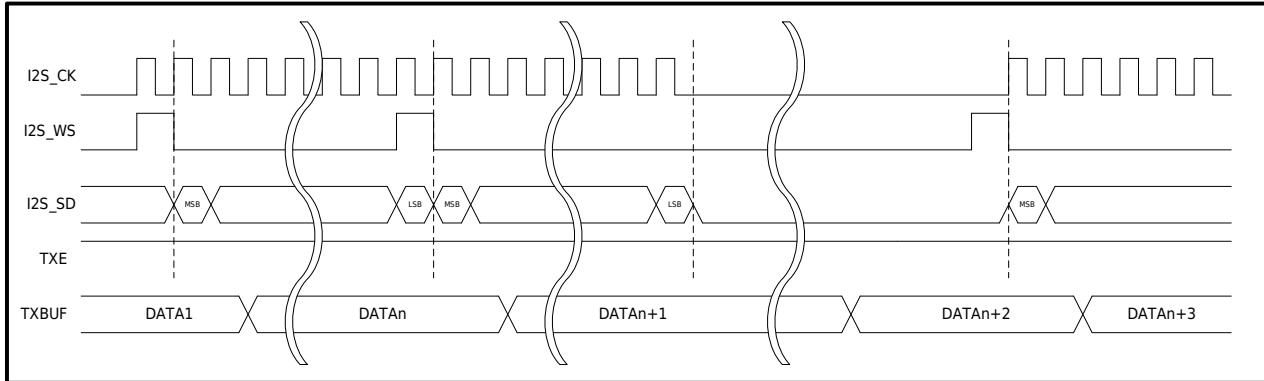


Figure 29-18 PCM Short Frame Host Resends After The Suspension Of Transmission Method 2

29.4.8.2 Precautions when using as a slave

- 1) When I2S acts as a slave, you need to ensure that after all register configurations are completed, turn on I2S_CTRL.TXE or I2S_CTRL.RXE to start the slave operation.
- 2) When I2S starts the slave operation in Philips, MSB/LSB mode, it is necessary to ensure that the WS signal is at the right channel level state when starting. When I2S starts the slave operation in PCM mode, it is necessary to ensure that the WS signal is low when starting. level status.
- 3) When I2S is used as a slave to receive data, the data received each time will be read after the next frame of data reception starts. Therefore, when the communication is suspended or terminated, the last frame of data received by I2S will be read in the next frame. Can only be read when the newsletter starts.
- 4) When I2S performs data reception as a slave in Philips, MSB/LSB mode, it will check whether WS is the left channel level when receiving each frame of left channel data. When I2S is receiving data as a slave in PCM mode, before each frame of data is received, it will check whether WS has generated a valid level according to the standard communication protocol.

29.5 Register Description

I2S1 base address: 0x4001_E000

I2S2 base address: 0x4001_E400

I2S3 base address: 0x4002_2000

I2S4 base address: 0x4002_2400

Table 29-5 List of I2S Registers

Register name	Symbol	Offset address	Bit width	Reset value
I2S Control Register	I2S_CTRL	0x000	32	0x0000_2200
I2S state register	I2S_SR	0x004	32	0x0000_0014
I2S Error Status Register	I2S_ER	0x008	32	0x0000_0000
I2S configuration register	I2S_CFGR	0x00C	32	0x0000_0000
I2S transmit buffer FIFO data register	I2S_TXBUF	0x010	32	0x0000_0000
I2S receive buffer FIFO data register	I2S_RXBUF	0x014	32	0x0000_0000
I2S Prescaler Register	I2S_PR	0x018	32	0x0000_0002

Note: Only 32-bit write registers are supported

CMU_BASE_ADDR2: 0x40054000

Register name	Symbol	Offset address	Bit width	Reset value
CMU I2S Clock Configuration Register	CMU_I2SCKSEL	0x012	16	0xB BBBB

Note: This register is detailed in the CMU chapter. When I2S master mode clock source selects I2SPLL, use this register to configure the clock source, which can be configured as UPLL/UPLLQ/UPLLP/MPLL/R/MPLLQ/MPLLP.

29.5.1 I2S Control Register (I2S_CTRL)

Offset address: 0x000

Reset value: 0x0000_2200

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	CLKSEL	DUPLEX	CKOE	LRCKOE	SDOE	I2SPLLSEL	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~b27	Reserved	-	Read as "0", write as "0"	R
b23	CLKSEL	Timer selection	0: select I2SPLL 1: select external clock	R/W
b22	DUPLEX	Communication method selection	0: half duplex 1: full duplex	R/W
b21	CKOE	Communication clock output 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	I2SPLLSEL	I2SPLL input selection	0: input disabled 1: Enter permission	R/W
b17	CODECRC	Codec Reset Control	0: Software Reset 1: Decommission	R/W
b16	FIFOR	fifo reset	0: Decommission 1: Software Reset	R/W
b15	Reserved	-	Read as "0", write as "0"	R
b14~b12	RXBIRQWL[2:0]	Receive buffer interrupt request level	Interrupt request triggers when the available space is less than the set value Note: It can only be set to 0/1/2, because the fifo space is 2	R/W
b11	Reserved	-	Read as "0", write as "0"	R
b10~b8	TXBIRQWL[2:0]	Transmit buffer interrupt request level	Interrupt request triggers when the available space is more than the set value Note: It can only be set to 0/1/2, because the fifo space is 2	R/W
b7	MCKOE	Drive clock output enable	0: disable driving clock output 1: Enable drive 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 is valid	R/W
b2	RXE	Receive enable	0: Disable reception 1: Allow receiving	R/W
b1	TXIE	Transmit interrupt enable	0: send interrupt invalid 1: Send interrupt is valid	R/W
b0	TXE	Send enable	0: Disable sending 1: Allow sending	R/W

29.5.2 I2S Status Register (I2S_SR)

Offset address: 0x004

Reset value: 0x0000_0014

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

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R
b5	RXBF	Receive buffer full	0: Receive buffer is not full 1: Receive buffer is full	R
b4	RXBE	Receive buffer empty	0: receive buffer is not empty 1: Receive buffer is empty	R
b3	TXBF	Send buffer full	0: send buffer is not full 1: send buffer is not full	R
b2	TXBE	Send buffer empty	0: send buffer is not empty 1: Send buffer is empty	R
b1	RXBA	Receive buffer alarm (related to water level)	0: receive buffer no alarm 1: receive buffer alarm	R
b0	TXBA	Send buffered alarm (related to water level)	0: send buffer no alarm 1: Send buffer alarm	R

29.5.3 I2S Error Status Register (I2S_ER)

Offset address: 0x008

Reset value: 0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b2	Reserved	-	Read as "0", write as "0"	R
b1	RXERR	receive error	0: Do not clear the flag bit 1: clear flag bit	R/W
b0	TXERR	Send Error	0: Do not clear the flag bit 1: clear flag bit	R/W

TXERR=1 when transmit overflow/underflow occurs, RXERR=1 when receive overflow occurs.

Writing 1 to the TXERR bit can clear the flag bit when sending overflow/underflow occurs, and writing 1 to the RXERR bit can clear the flag bit when receiving overflow occurs.

29.5.4 I2S Configuration Register (I2S_CFGR)

Offset address: 0x00C

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	PCMSY NC	CHLEN	DATLEN[1:0]	I2SSTD[1:0]		

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R
b5	PCMSYNC	PCM frame synchronization	PCM frame synchronization 0: short frame synchronization 1: long frame synchronization Note: This bit is meaningful 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 agreement 01: MSB justified protocol (left justified) 10: LSB justified protocol (right justified) 11: PCM protocol	R/W

29.5.5 I2S Transmit Buffer FIFO Data Register (I2S_TXBUF)

Offset address: 0x010

Reset value: 0x0000_0000

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]															

Bit	Marking	Place name	Function	Read and write
b31~b0	TXBUF[31:0]	Send data	Store send data	W

Note:

- For 16-bit frames, TXBUF[15:0] stores one frame of left channel or one frame of right channel transmission data.
- For 32-bit frames, TXBUF[31:0] stores one frame of left channel or one frame of right channel transmission data.

29.5.6 I2S Receive Buffer FIFO Data Register (I2S_RXBUF)

Offset address: 0x014

Reset value: 0x0000_0000

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:

- For 16-bit frames, RXBUF[15:0] stores one frame of left channel or one frame of right channel received data.
- For 32-bit frames, RXBUF[31:0] stores one frame of left channel or one frame of right channel received data.

29.5.7 I2S Prescaler Register (I2S_PR)

Offset address: 0x018

Reset value: 0x0000_0002

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	I2SDIV[[7:0]]						
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b8	Reserved	-	Read as "0", write as "0"	R											
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															
b7~b0	I2SDIV[7:0]	Frequency division factor		R/W											

30 Controller Area Network (CAN)

30.1 Introduction

CAN (Controller Area Network) bus is a bus standard that can realize mutual communication between microprocessors or devices without a host. This module follows the CAN bus protocol 2.0A and 2.0B and is upwardly compatible with CAN-FD. The CAN bus controller can handle the data sending and receiving on the bus. In this product, CAN has 8 groups of filters. Filters are used to select messages for an application to receive.

The application sends data to the bus through a high-priority primary transmit buffer (Primary Transmit Buffer, hereinafter referred to as PTB) and 4 secondary transmit buffers (Secondary Transmit Buffer, hereinafter referred to as STB), which is determined by the transmit scheduler Email sending order. The bus data is obtained through 10 receive buffers (Receive Buffer, hereinafter referred to as RB). 4 STBs and 10 RBs can be understood as a 4-level FIFO and a 10-level FIFO, and the FIFO is completely controlled by hardware.

The CAN bus controller can also support time-triggered CAN communication (Time-trigger communication).

Main Features of CAN:

- Fully support CAN2.0A/CAN2.0B protocol.
- Upwardly compatible with CAN-FD protocol.
- Support the highest communication baud rate 1Mbit/s
- Support 1~1/256 baud rate prescaler, flexible configuration of baud rate.
- 10 receive buffers
 - FIFO mode
 - Errors or unreceived data will not overwrite stored messages
- 1 high priority main transmit buffer PTB
- 4 secondary transmit buffers STB
 - FIFO mode
 - priority arbitration
- 8 sets of independent filters
 - Support 11-bit standard ID and 29-bit extended ID
 - Programmable ID CODE bits and MASK bits
- Both PTB/STB support single send mode
- Support silent mode
- Support loopback mode
- Supports capturing transmission error types and locating arbitration failure locations
- Programmable error warning value

- Support ISO11898-4 specified time trigger CAN and receive timestamp

30.2 CAN System Block Diagram

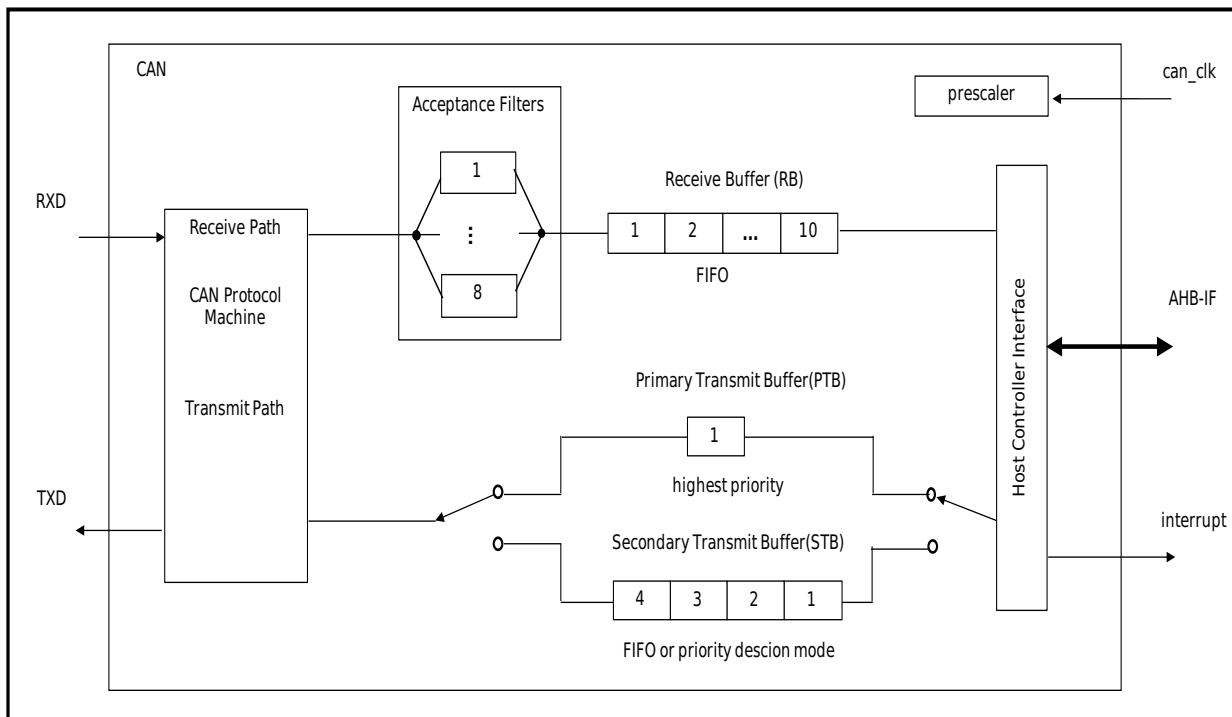


Figure 30-1 CAN System Block Diagram

30.3 Pin Description

Table 30-1 CAN Pin Descriptions

Pin name	Direction	Functional description
CAN_RxD	Input	CAN receive data signal
CAN_TxD	Output	CAN sends data signal

30.4 Functional Description

This chapter will describe the CAN function in detail.

30.4.1 Baud Rate Setting

The clock source of the clock can_clk used by CAN communication is an external high-speed oscillator. Before using the CAN module, it is necessary to set the CAN communication clock in the CMU chapter. The clock selection must meet the setting condition that EXCLK (CAN control logic clock) is 1.5 times or more than can_clk (CAN communication clock).

The figure below shows the CAN bit time definition diagram. 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 the CAN controller CAN-CTRL. Among them, segment1 and segment2 can be set through the register BT. The BT register can only be set when CFG_STAT.RESET=1, that is, when the CAN software is reset.

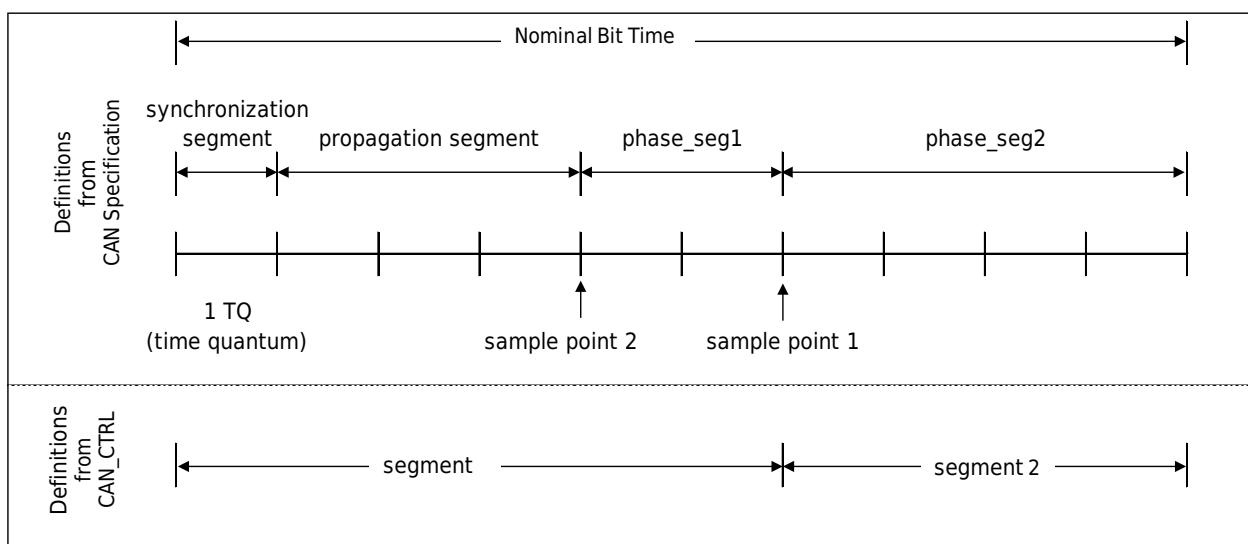


Figure 30-2 CAN bit time definition diagram

Please refer to the following formula for the TQ calculation method, where PRESC is set by the PRESC bit of the BT register. f_{can_clk} It is CAN communication clock frequency.

$$TQ = \frac{PRESC+1}{f_{can_clk}}$$

For the calculation method of sampling points, please refer to the following instructions.

- 1) If the PRESC bit of the BT register is set to ≥ 1 , refer to sample point 1 in Figure 30-2, the sampling point is located at the boundary between segment1 and segment2.
- 2) If the PRESC bit of the BT register is set to 0, as shown in sample point 2 in Figure 30-2, the sampling point is 2 TQ ahead of the segment1 and segment2 boundary points.

It is recommended to set the PRESC bit of the BT register to a value ≥ 1 .

Please refer to the following formula for the bit time calculation method, where SEG_1 and SEG_2 are set by the SEG_1 and SEG_2 bits of the BT register.

$$BT=t_{SEG1}+t_{SEG2}=((SEG_1+2)+(SEG_2+1))\times TQ$$

Table 30-2 CAN bit time setting rules

Bit	Predetermined area	Rule
SEG_1 bit of BT register	0~63	SEG_1 ≥ SEG_2 + 1 SEG_2 ≥ SJW
SEG_2 bit of BT register	0~7	
SJW bit of BT register	0~7	

30.4.2 Send Buffer

CAN_CTRL provides two sending buffers for sending data, the main sending data buffer PTB and the secondary sending buffer STB. PTB has the highest priority, but only one frame of data can be buffered. STB has a lower priority than PTB, but it can buffer 4 frames of data, and 4 frames of data in STB can work in FIFO mode or priority arbitration mode.

The 4 frames of data in the STB can be sent by setting the TSALL bit of the TCMD register to 1. In the FIFO mode, the first written data is sent first. In the priority mode, the data with the smaller ID is sent first.

Data in a PTB has the highest priority, so a PTB send can postpone an STB send, but an STB that has already won arbitration and started sending cannot be delayed by a PTB send.

A frame of data in PTB and STB takes 4 words and can be accessed through the TBUF register. Select PTB or STB through the TBSEL bit of the TCMD register, TBSEL=0, select PTB, TBSEL=1, select STB. The next SLOT in the STB is selected by the TSNEXT bit of the TCTRL register. The corresponding relationship is shown in the figure below:

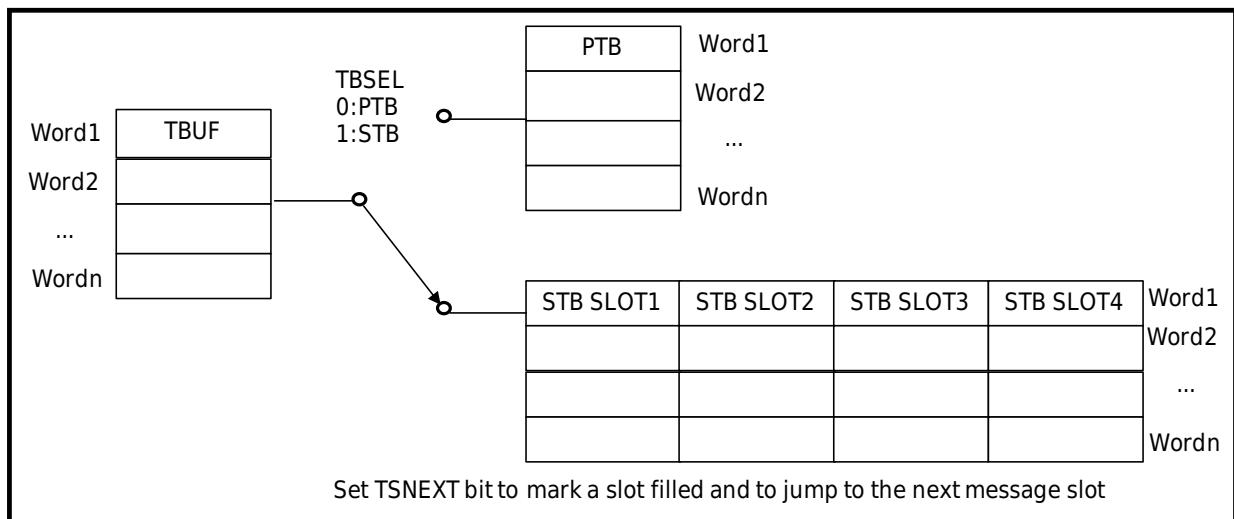


Figure 30-3 CAN TBUF Register Write Transmit Buffer And Schematic Diagram

30.4.3 receive buffer

CAN_CTRL provides 10 SLOT receive buffers for storing received data, and the 10 SLOT receive buffers work in FIFO mode. Each RB SLOT needs to occupy 4 words, read the received data through the RBUF register, always read the earliest received data first, and set the RREL of the RCTRL register to 1 to release the read RB SLOT, and Points to the next RB SLOT.

The schematic diagram of reading RB SLOT through RBUF is as follows.

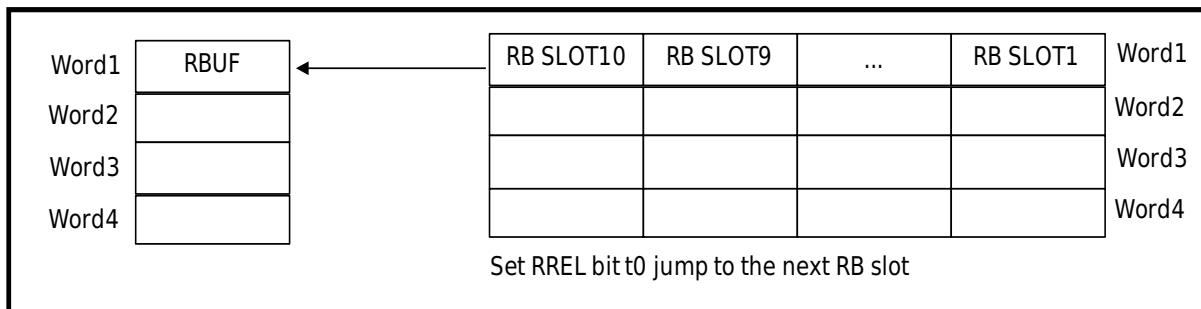


Figure 30-4 Schematic Diagram Of CAN RBUF Register Read Receive Buffer

30.4.4 Receive Filter Register Set

CAN_CTRL provides 8 sets of 32-bit filters to filter the received data to reduce CPU load. The filters can support standard format 11-bit ID or extended format 29-bit ID. Each set of filters 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 bit 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 of the 8 sets of filters, and the received data will be stored in RB, otherwise the data will not be received or stored.

Each group of filters is enabled or disabled through the ACFEN register. ID CODE and ID MASK are set by the SELMASK bit of the ACFCTRL register. When SELMASK=0, it points to ID CODE, and when SELMASK=1, it points to ID MASK. The filter is selected by the ACFADR bit of the ACFCTRL register. ID CODE and ID MASK are accessed through the ACF register and can only be set when CFG_STAT.RESET=1, that is, CAN software reset. Please refer to the figure below for the method of ACF registering and accessing the filter register group.

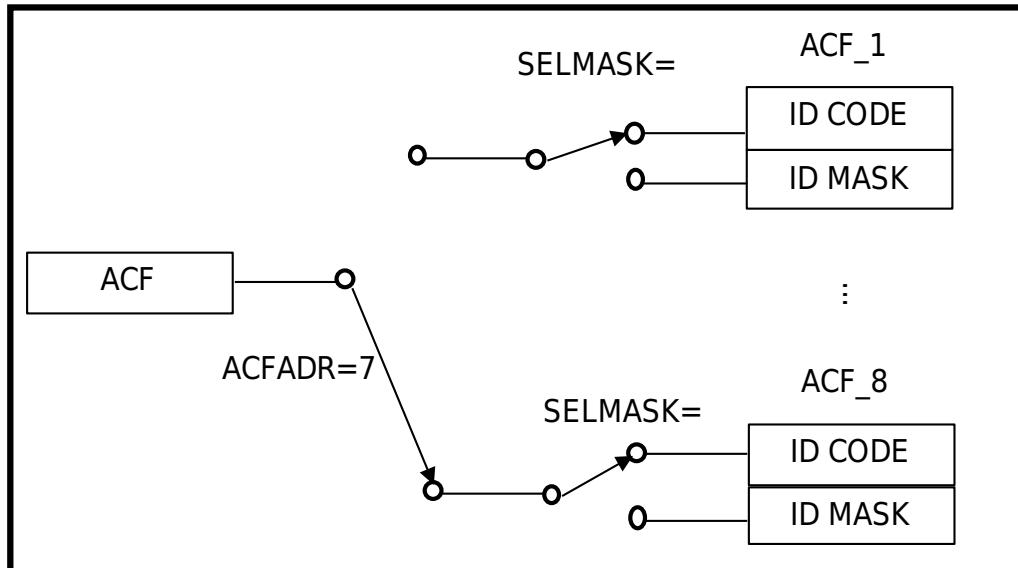


Figure 30-5 Schematic Diagram Of CAN ACF Register Access Filter Group

30.4.5 Data Transmission

Before starting to send, you must ensure that the data to be sent by the PTB or STB has been filled, and then start the PTB or STB to send. Data is not allowed to be filled again during sending.

The sending data setting steps are as follows:

1. Set TBSEL to select sending BUF from PTB and STB
2. Write the data to be sent through the TBUF register
3. If STB is selected, set TSNEXT=1 to complete the loading of all STB SLOTS
4. Send enable
 - PTB sent using TPE
 - STB sends using TSALL or TSONE
5. send complete status acknowledgment
 - PTB transmission is completed using TPIF, TPIE is used to enable TPIF
 - STB uses TSONE to send TSIF when it is completed, and TSIE is used to enable TSIF
 - STB uses TSIF when TSALL transmission is completed. At this time, after all the STB SLOT data that needs to be set are sent, TSIF is set, and TSIE is used to enable TSIF

30.4.6 Single Data Transmission

When the automatic resend function is not required, it can be set to single-send mode through the register. The TPSS bit of the CFG_STAT register is used for the single-send mode setting of PTB, and the TSSS bit is used for the single-send mode of STB. When the data is successfully sent, the action of single sending and normal sending mode is the same. But when the data is not sent successfully, the following results will appear:

- When TPIF is set (TPIE=1), the corresponding BUF SLOT data will be cleared.

- When an error is sent, KOER is updated and BEIF is set (BEIE=1).
- Arbitration fails, ALIF is set (ALIE=1).

In the single sending mode, TPIF cannot be used alone to judge whether the sending is completed. It needs to judge whether the sending is completed together with BEIF and ALIF.

30.4.7 Cancel Data Sending

Data transmissions that have been requested but not yet executed can be canceled via TPA or TSA. Cancellation of data transmission will occur in the following situations:

- in arbitration
 - If the node arbitration fails, the data transmission is canceled.
 - If the node arbitration is successful, it will continue to send.
- data sending
 - If data is successfully sent and ACK is received, the corresponding flag and status are normally set. Data transmission is not canceled.
 - If the data is sent successfully but no ACK is received, the data sending is canceled and the error counter is incremented.
 - The sending data set by TSALL=1, the STB SLOT data that is being sent are sent normally, and the STB SLOT that has not started sending is canceled.

The result of canceling data sending has the following two situations:

- TPA releases PTB and makes TPE=0.
- The TSA releases one STB SLOT or all STB SLOTS depending on whether TSONE or TSALL enabled transmission.

30.4.8 Data Reception

The receive filter group can filter out unnecessary received data, reduce the occurrence of interrupts and the reading of RB, thereby reducing the CPU load. The receiving data setting steps are as follows:

1. Set filter groups.
2. Set RFIE, RAFIE and AFWL.
3. Wait for RFIF or RAIF.
4. Read the earliest received data from RB FIFO via RBUF.
5. Set RREL=1 to select the next RB SLOT.
6. Repeat 4, 5 until the RB is confirmed to be empty by RSTAT.

30.4.9 Error Handling

On the one hand, CAN_CTRL can automatically handle some errors, such as automatically resending data or discarding received frames containing errors, and on the other hand, reports errors to the CPU through interrupts.

CAN nodes have the following three error states:

- Error Active: The node automatically sends an active error flag when it detects an error.
- Error Passive: A passive error flag is automatically sent when a node detects an error.
- Node closed: In the closed state, this node no longer affects the entire CAN network.

CAN_CTRL provides TECNT and RECNT two counters for counting errors. TECNT and RECNT counters increase or decrease according to the rules stipulated in the CAN2.0B agreement. In addition, a programmable CAN error warning LIMIT register is used to generate an error interrupt to notify the CPU.

There are the following five error types during CAN communication, and the error types can be identified by the KOER bit of the EALCAP register.

- Bit error
- Wrong form
- Filling error
- Wrong answer
- CRC error

30.4.10 Node Down

When the number of sending errors is greater than 255, the CAN node automatically enters the node shutdown state and does not participate in CAN communication until it returns to the error active state. The CAN node shutdown status can be confirmed by the BUSOFF bit of the CFG_STAT register. BUSOFF is set and EIF interrupt is generated at the same time.

There are two methods for CAN to recover from the node off state to the error active state:

- Power-on reset
- Received 128 consecutive 11-bit stealth bit sequences (recovery sequence)

In the node shutdown state, the TECNT value remains unchanged, and the RECNT is used for the count recovery sequence. After recovering from node shutdown, TECNT and RECNT are reset to 0.

When the node shutdown flag BUSOFF is set, the RESET bit of the CFG_STAT register is also set.

30.4.11 Lost Arbitration Position Capture

CAN_CTRL can accurately capture the position of the arbitration failure bit and reflect it in the ALC register. The ALC register stores the position of the last arbitration failure 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 data bit ALC is 0, the second ID data bit ALC is 1, and so on. Since arbitration only takes place within the arbitration field, the maximum value of ALC is 31. For

example, a standard format remote frame arbitrates with an extended frame, if the extended frame fails at the IDE bit, then ALC=12.

30.4.12 Loop 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 testing purposes.

In the internal loopback mode, the module internally connects the receiving data line to the sending data line, and the sending data is not output. Internal loopback mode, nodes generate self-acknowledgment signals to avoid ACK errors.

The external loopback mode maintains the connection with the transceiver so that the data sent can still appear on the CAN bus. With the help of the transceiver, CAN can receive the data sent by itself. In the external loopback mode, the SACK bit of the RCTRL register can be used to determine whether to generate a self-acknowledgment signal. When SACK=0, no self-acknowledge signal is generated. When SACK=1, a self-acknowledge signal is generated.

In external loopback mode, when SACK=0, the following two situations will occur:

- Other nodes also receive the data frame sent by the node and send a response signal. In this case, the node can successfully send and receive data.
- If no other node returns an acknowledgment signal, an acknowledgment error is generated, the data is resent and the error counter is incremented. At this time, it is recommended to use the single sending mode.

When returning to normal mode from loopback mode, in addition to clearing the mode bit, a software reset of CAN_CTRL is required.

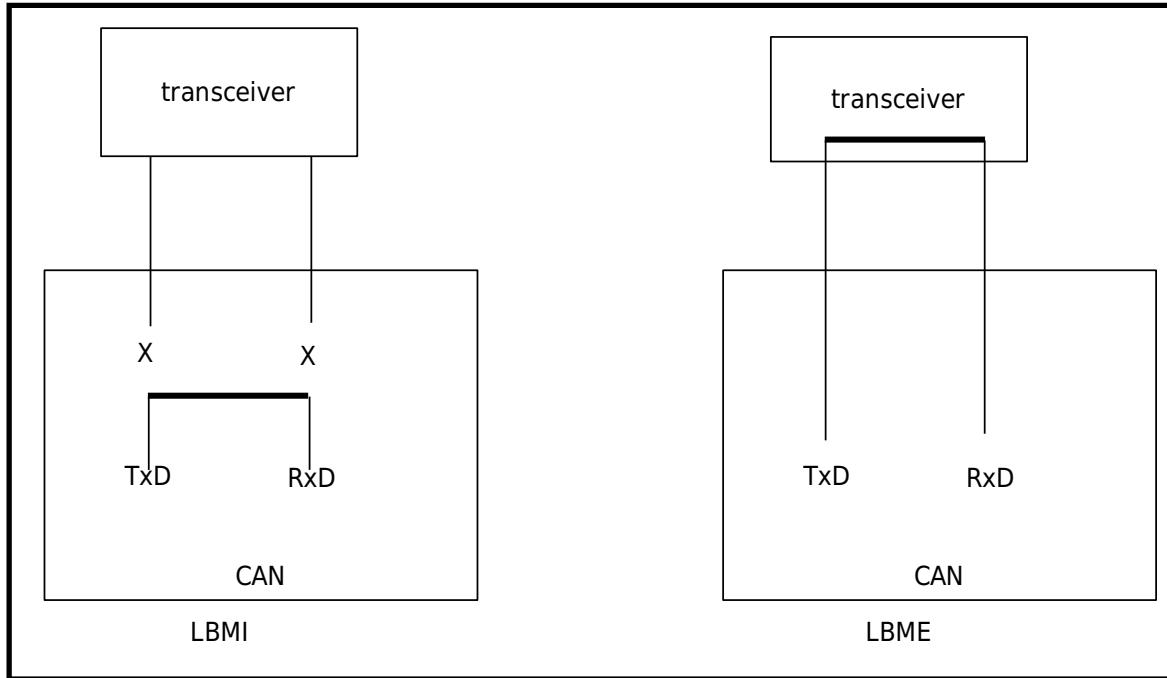


Figure 30-6 Schematic Diagram of CAN LBMI and LBME

30.4.13 Silent Mode

Silent mode can be used to monitor CAN network data. In silent mode, data can be received from the CAN bus without sending any data to the bus. Set the LOM in the TCMD register to 1 to make the CAN bus controller enter the silent mode, and clear it to 0 to leave the silent mode.

The external loopback mode can be combined with the silent mode to form an external loopback silent mode. At this time, CAN can be regarded as a quiet receiver, but can send data when necessary. In external loopback silent mode, frames containing self-acknowledgment signals are allowed to be sent, but the node does not generate error flags and overload frames.

30.4.14 Software Reset Function

By setting the RESET bit of the CFG_STAT register to 1, the software reset function is realized. The reset range of the software reset function is shown in the table below.

Table 30-3 Software Reset Range Table

Register Place name	Software Reduction	Remark	Register Place name	Software Reduction	Remark
ACFADR	No	-	EWL	Yes	-
ACODE	No	Can only be written by software reset	KOER	Yes	-
AE_x	No	-	LBME	Yes	-
AFWL	No	-	LBMI	Yes	-
AIF	Yes	-	RACTIVE	Yes	Reception stops immediately and no ACK is generated
ALC	Yes	-	RAFIE	No	-
ALIE	No	-	RAFIF	Yes	-
ALIF	Yes	-	RBALL	Yes	-
AMASK	No	Can only be written by software reset	RBUF	Yes	All RBs are marked as empty, the value is variable
BEIE	No	-	REF_ID	No	-
BEIF	Yes	-	REF_IDE	No	-
BUSOFF	No	Cleared 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	-

Register Place name	Software Reduction	Remark	Register Place name	Software Reduction	Remark
ROV	Yes	-	TSNEXT	Yes	-
RREL	Yes	-	TSONE	Yes	-
PRESC	No	Can only be written by software reset	TPIE	No	-
RSTAT	Yes	-	TPIF	Yes	-
SACK	Yes	-	TPSS	Yes	-
SELMASK	No	-	TSFF	Yes	All STB SLOTS are marked empty
SEG_1	No	Can only be written by software reset	TSIE	No	-
SEG_2	No	Can only be written by software reset	TSIF	Yes	-
SJW	No	Can only be written by software reset	TSSS	Yes	-
TACTIVE	Yes	send immediately stop	TSSTAT	Yes	All STB SLOTS are marked empty
TBE	Yes	-	TTEN	Yes Yes	-
TBF	No	-	TTIF	Yes	-
TBPTR	No	-	TTIE	No	-
TBSEL	Yes	-	TTPTR	No	-
TBUF	Yes	All STBs are marked empty, pointing to PTB	TTTBM	No	-
TECNT	No	Cleared by BUSOFF=1	TTTYPE	No	-
TEIF	Yes	-	TT_TRIG	No	-
TPA	Yes	-	TT_WTRIG	No	-
TPE	Yes	-	T_PRESC	No	-
TSA	Yes	-	WTIE	No	-
TSALL	Yes	-	WTIF	Yes	
TSMODE	No	-			

30.4.15 Upward Compatible with CAN-FD Function

Even if CAN-CTRL receives CAN-FD frames in a CAN-FD network, the receiver will automatically ignore these frames and not return ACK, and wait until the bus is free before sending or receiving the next CAN2.0B frame.

30.4.16 Time-Triggered TTCAN

CAN-CTRL provides partial (level 1) hardware support for the time-triggered communication method specified in ISO11898-4. This chapter introduces TTCAN functions from the following 5 parts.

30.4.16.1 TBUF Behavior in TTCAN Mode

TTTBM=1

When TTTBM=1, PTB and STB SLOT form TB SLOT as well, and send BUF through the TBPTR register. When TBPTR=0, it points to PTB, TBPTR=1 points to STB SLOT1, and so on. The host can mark the sending BUF SLOT through the TPE and TPF registers. At this time, the TBSEL and TSNEXT registers have no meaning and can be ignored.

When TTTBM=1, PTB does not have any special attributes. Like STB SLOT, the transmission completion flag also uses TSIF.

In TTCAN mode, there is no FIFO mode and priority arbitration mode for sending BUF, and only one selected SLOT can send data.

In TTCAN mode, time-triggered mode is required for transmission start, TPE, TSONE, TSALL, TPSS and TPA are fixed to 0 and ignored.

TTTBM=0

When TTTBM=0, use event-driven communication and receive timestamp function in combination. In this mode, the functions of PTB and STB are the same as when TTEN=0, so PTB always has the highest priority, and STB can work in FIFO mode or arbitration mode.

30.4.16.2 TTCAN Function Description

After power on, Time Master needs to be initialized according to the ISO 11898-4 protocol. In a CAN network, there can be up to 8 potential Time Masters. Each Time Master has its own reference message ID (the last 3 digits of the ID). These potential Time Masters send respective reference messages according to their own priorities.

After TTEN=1, the 16-bit counter starts to work. When the reference message is successfully received or the Time Master successfully sends the reference message, the CAN controller copies Sync_Mark to Ref_Mark, and Ref_Mark sets the cycle time to 0. The successful reception of the reference message sets the RIF flag and the successful transmission of the reference message sets

the TPIF flag or the TSIF flag. At this point the host needs to prepare the trigger conditions for the next action.

The trigger condition may be receiving a trigger. This trigger-only interrupt can be used to detect that an expected message has not been received.

The trigger condition can also be a send trigger. This trigger starts sending the data in the TBUF SLOT specified by the TTPTTR register. If the selected TBUF SLOT is marked empty, the transmission is not started, but the interrupt flag is set.

30.4.16.3 TTCAN Timing

CAN_CTRL supports ISO11898-4 level 1. A 16-bit counter is included operating at the bit times defined for PRESC, SEG_1, SET_2. If TTEN=1, there is an additional prescaler T_PRESC.

When SOF of one frame of data, the value of the counter is Sync_Mark. If the frame data is a reference message, copy Sync_Mark to Ref_Mark. The cycle time is equal to the value of the counter minus Ref_Mark. This time is used as a time stamp for receiving messages or as a trigger time reference for sending messages.

30.4.16.4 TTCAN Trigger Mode

The trigger mode of TTCAN is defined through the TTTYPE register, the TTPTTR register specifies sending SLOT, and TT_TRIG specifies the cycle time of the trigger.

Contains the following five trigger methods:

- Trigger immediately
- Time trigger
- Single send trigger
- Send start trigger
- Send stop trigger

All triggers use the TTIF flag except the immediate trigger mode. When TTTBM=1, only time trigger mode is supported.

Trigger Immediately

By writing the high bit of TT_TRIG (don't care about the value written), the trigger is started. In this mode, the data in the TBUF SLOT selected by TTPTTR will be sent immediately. TTIF is not set.

Time Trigger

The time trigger mode only generates interrupts by setting the TTIF flag, and has no other functions. Time triggering can be used if a node expects to receive expected data within a specific time window. If the TT_TRIG value is less than the actual cycle time, TEIF is set and there is no other action.

Single Send Trigger

The one-shot trigger method is used to send data within the execution time window. At this time, the TSSS bit is ignored.

Set the Tick of up to 16 cycle times specified by ISO11898-4 through the TEW bit, and the setting range is 1~16. If data transmission does not start within the specified transmit enable time window, the frame is discarded. The corresponding sending BUF SLOT is marked as empty, and AIF is set, and the data in the corresponding sending BUF 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 there is no other action.

Send start trigger

The sending start trigger mode is used in the arbitration time window to participate in the arbitration. TSSS is used to decide whether to automatically repeat or send a single mode. A send stop trigger can be used to stop the send if the specified message was not sent successfully.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and there is no other action.

Send stop trigger

The send stop trigger method is used to stop the send that has been started by the send start trigger method. If the sending is stopped, the sending frame is discarded, the AIF is set and the selected TBUF SLOT is marked as empty, but the data in the TBUF SLOT will not be rewritten, and it can be sent again by setting the TPF.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and execution stops.

30.4.16.5 TTCAN Trigger Gate Time

The TTCAN trigger watchdog function is similar to the watchdog function and is used when TTTBM=1. Used to watch the time since the last successful reference message received by the gate. The reference message can be received during the cycle time or after an event, and the application should set the appropriate gate time according to the specific situation.

If cycle count is equal to TT_WTRIG, set WTIF. Write 0 by WTIE to disable the gate trigger.

If TT_WTRIG is smaller than the actual cycle time, TEIF is set.

30.4.17 Interrupt

Table 30-4 CAN Interrupt Table

Interrupt logo	Description
RIF	Receive interrupt
ROIF	Receive overflow interrupt
RFIF	Receive BUF full interrupt
RAFIF	Receive BUF will be full interrupt
TPIF	PTB send interrupt
TSIF	STB send interrupt
EIF	error interrupt
AIF	Cancel send interrupt
EPIE	Error passive interrupt
ALIF	Arbitration Lost Interrupt
BEIF	Bus error interrupt
WTIF	Trigger gate interrupt
TEIF	Trigger error interrupt
TTIF	Time triggered interrupt

30.5 Register Description

CAN_BASE_ADDR:0x40070400

Table 30-5 List of CAN registers

Register name	Symbol	Offset address	Bit width	Reset value
CAN receive BUF register	CAN_RBUF	0x00~0x0F	128	0XXXXX XXXX
CAN send BUF register	CAN_TBUF	0x50~0x5F	128	0XXXXX XXXX
CAN Configuration and Status Register	CAN_CFG_STAT	0xA0	8	0x80
CAN command register	CAN_TCMD	0xA1	8	0x00
CAN send 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 Flags Register	CAN_ERRINT	0xA6	8	0x00
CAN Warning Limit Register	CAN_LIMIT	0xA7	8	0x1B
CAN Bit Timing Register	CAN_BT	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_TECNT	0xB3	8	0x00
CAN Filter Group Control Register	CAN_ACFCTRL	0xB4	8	0x00
CAN Filter Group Enable Register	CAN_ACFEN	0xB6	8	0x01
CAN filter group code and mask registers	CAN_ACF	0xB8	32	0XXXXX 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	0XXXXX XXXX
TTCAN trigger configuration register	CAN_TRG_CFG	0xC4	16	0x0000
TTCAN Trigger Time Register	CAN_TT_TRIG	0xC6	16	0x0000
TTCAN Trigger Watchdog Time Register	CAN_TT_WTRIG	0xC8	16	0x0000

Table 30-6 CAN Register BYTE/HALFWORD/WORD Access Arrangement Table

Address	BYTE access	HALFWORD access		WORD access			
0x00~0x0F	CAN_RBUF	CAN_RBUF		CAN_RBUF			
0x50~0x5F	CAN_TBUF	CAN_TBUF		CAN_TBUF			
0xA0	CAN_CFG_STAT	CAN_TCMD	CAN_CFG_STAT	CAN_RCTRL	CAN_TCTRL	CAN_TCMD	CAN_CFG_STAT
0xA1	CAN_TCMD	-		-			
0xA2	CAN_TCTRL	CAN_RCTRL	CAN_TCTRL	-			
0xA3	CAN_RCTRL	-		-			

Address	BYTE access	HALFWORD access		WORD access					
0xA4	CAN_RTIE	CAN_RTIF	CAN_RTIE	CAN_LIMIT	CAN_ERRINT	CAN_RTIF	CAN_RTE		
0xA5	CAN_RTIF			-					
0xA6	CAN_ERRINT	CAN_LIMIT	CAN_ERRINT	-					
0xA7	CAN_LIMIT			-					
0xA8	CAN_BT[7:0]	CAN_BT[15:0]		CAN_BT					
0xA9	CAN_BT[15:8]	-		-					
0xAA	CAN_BT[23:16]	-		-					
0xAB	CAN_BT[31:24]	CAN_BT[31:16]							
0xB0	-	-		CAN_TECNT	CAN_RECNT	-			
0xB1	-	-		-					
0xB2	CAN_RECNT	CAN_TECNT	CAN_RECNT	-					
0xB3	CAN_TECNT	-		-					
0xB4	CAN_ACFCTRL[7:0]	CAN_ACFCTRL		CAN_ACFEN		CAN_ACFCTRL			
0xB5	CAN_ACFCTRL[15:8]	-		-					
0xB6	CAN_ACFEN[7:0]	CAN_ACFEN		-					
0xB7	CAN_ACFEN[15:8]	-		-					
0xB8	CAN_ACF	CAN_ACF		CAN_ACF					
0xBC	-	-		CAN_TTCFG	CAN_TBSLOT	-			
0xBD	-	-		-					
0xBE	CAN_TBSLOT	CAN_TTCFG	CAN_TBSLOT	-					
0xBF	CAN_TTCFG	-		-					
0xC0	CAN_REF_MSG[7:0]	CAN_REF_MSG[15:0]		CAN_REF_MSG					
0xC1	CAN_REF_MSG[15:8]	-		-					
0xC2	CAN_REF_MSG[23:16]	CAN_REF_MSG[31:16]		-					
0xC3	CAN_REF_MSG[31:24]	-		-					
0xC4	CAN_TRG_CFG[7:0]	CAN_TRG_CFG		CAN_TT_TRIGGER		CAN_TRG_CFG			
0xC5	CAN_TRG_CFG[15:8]	-		-					
0xC6	CAN_TT_TRIGGER[7:0]	CAN_TT_TRIGGER		-					
0xC7	CAN_TT_TRIGGER[15:8]	-		-					
0xC8	CAN_TT_WTRIG[7:0]	CAN_TT_WTRIG				CAN_TT_WTRIG			
0xC9	CAN_TT_WTRIG[15:8]								

30.5.1 CAN Receive Buffer Registers (CAN_RBUF)

Offset address: 0x00

Reset value: 0XXXX XXXX

The RBUF register points to the RB SLOT address of the earliest received CAN mailbox, and the RBUF register can be read in any order.

The KOER bit is the register EALCAP.KOER, which is only meaningful when RBALL=1.

The TX bit indicates that the message sent by itself is received in the loopback mode.

The CYCLE_TIME bit is only valid in TTCAN mode, indicating the cycle time when SOF starts.

The data format of the CAN receiving mailbox is as follows:

Table 30-7 Standard Format CAN Receiving Mailbox Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
RBUF	ID[7:0]								ID
RBUF+1	-				ID[10:8]				ID
RBUF+2	-								ID
RBUF+3	-								ID
RBUF+4	IDE=0	RTR	0	0	DLC[3:0]				Control
RBUF+5	KOER[2:0]			TX	-				Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN
RBUF+7	CYCLE_TIME[15:8]								TTCAN
RBUF+8	DATA1								Data
RBUF+9	DATA2								Data
RBUF+10	DATA3								Data
RBUF+11	DATA4								Data
RBUF+12	DATA5								Data
RBUF+13	DATA6								Data
RBUF+14	DATA7								Data
RBUF+15	DATA8								Data

Table 30-8 Extended Format CAN Receive Mailbox Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
RBUF	ID[7:0]								ID
RBUF+1	ID[15:8] ID[10:8]								ID
RBUF+2	ID[23:16]								ID
RBUF+3	-			ID[28:24]					ID
RBUF+4	IDE=1	RTR	0	0	DLC[3:0]				Control
RBUF+5	KOER[2:0] TX			TX	-				Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN
RBUF+7	CYCLE_TIME[15:8]								TTCAN
RBUF+8	DATA1								Data
RBUF+9	DATA2								Data
RBUF+10	DATA3								Data
RBUF+11	DATA4								Data
RBUF+12	DATA5								Data
RBUF+13	DATA6								Data
RBUF+14	DATA7								Data
RBUF+15	DATA8								Data

The 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, the setting range is 0~8, and the corresponding data length is 0Byte~8Byte

30.5.2 CAN Transmit Buffer Registers (CAN_TBUF)

Offset address: 0x50

Reset value: 0xFFFF XXXX

The TBUF register points to the next empty CAN sending BUF SLOT, and the TBUF register can be read in any order. Write 1 to TSNEXT by software to mark that the corresponding TBUF SLOT has written data, thus pointing to the next TBUF SLOT.

TBUF can only be accessed by WORD.

The data format of the CAN sending mailbox is as follows:

Table 30-9 Standard Format CAN Sending Mailbox Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
TBUF	ID[7:0]								ID
TBUF+1	-				ID[10:8]				ID
TBUF+2	-								ID
TBUF+3	-								ID
TBUF+4	IDE=0	RTR	0	0	DLC[3:0]				Control
TBUF+5	-								-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
TBUF+11	DATA4								Data
TBUF+12	DATA5								Data
TBUF+13	DATA6								Data
TBUF+14	DATA7								Data
TBUF+15	DATA8								Data

Table 30-10 Extended Format CAN 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	0	0	DLC[3:0]				Control
TBUF+5	-								-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
TBUF+11	DATA4								Data
TBUF+12	DATA5								Data
TBUF+13	DATA6								Data
TBUF+14	DATA7								Data
TBUF+15	DATA8								Data

The 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, the setting range is 0~8, and the corresponding data length is 0Byte~8Byte

30.5.3 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	Marking	Place name	Function	Read and write
b7	RESET	Reset request	reset request bit 0: Do not request a partial reset 1: request 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 when RESET=0.	R/W
b6	LBME	External loopback mode enable bit	External loopback mode enable bit 0: disable external loopback mode 1: Enable external loopback mode Note: It is forbidden to set this bit during communication.	R/W
b5	LBMI	Internal loopback mode enable bit	Internal loopback mode enable bit 0: disable internal loopback mode 1: Enable internal loopback mode Note: It is forbidden to set this bit during communication.	R/W
b4	TPSS	PTB single transfer mode	PTB single transfer mode 0: Disable PTB single transfer mode 1: Enable PTB single transfer mode	R/W
b3	TSSS	STB single transfer mode	STB single transfer mode 0: Disable STB single transfer mode 1: Enable STB single transfer mode	R/W
b2	RACTIVE	Receive status signal	Receive status signal 0: not receiving 1: receiving	R
b1	TACTIVE	Sending status signal	Sending status signal 0: not sending 1: Sending	R
b0	BUSOFF	Bus off state	Bus off state 0: Bus valid state 1: Bus off state NOTE: Writing a 1 clears the TECNT and RECNT registers, but only for debug purposes.	R/W

30.5.4 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	Marking	Place name	Function	Read and write
b7	TBSEL	Send BUF selection bit	Transmit Buffer Select 0: PTB 1: STB When TTEN=1&TTTB=1, TBSEL is reset to reset value. Note: When writing TBUF register or TSNEXT bit, this bit needs to keep a constant value.	R/W
b6	LOM	Silent mode enable bit	Listen Only Mode 0: disable silent mode 1: Enable silent mode Sending 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 it can send data. Note: It is forbidden to set this bit during communication.	R/W
b5	Reserved	-	The reset value must be maintained.	R
b4	TPE	PTB transmit enable bit	Transmit Primary Enable 0: Disable PTB sending 1: Enable PTB transmission After this bit is enabled, the Mailbox in PTB will be sent at the next available sending position. STB transmissions that have already started will continue, but the next pending STB transmission will be delayed until the PTB transmission is complete. After this bit is written to 1, it will remain 1 until the PTB transmission is completed or the transmission is canceled by TPA. Software cannot clear this bit by writing a 0. The TPE is reset to the reset value by hardware in the following cases: - RESET=1 - BUSOFF=1 - LOM=1&LBME=0 - TTEN=1&TTTB=1	R/W

Bit	Marking	Place name	Function	Read and write
b3	TPA	PTB send cancel bit	<p>Transmit Primary Abort 0: do not cancel 1: Cancel the PTB transmission that has been requested by setting TPE to 1 but has not yet started</p> <p>This bit is written 1 by software but cleared by hardware. The TPE bit can be cleared by writing 1, so write 1 at the same time as TPE.</p> <p>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&TTBMM=1 	R/W
b2	TSONE	Send a frame of STB data set bit	<p>Transmit Secondary ONE frame 0: do not send 1: Send a frame of STB data</p> <p>In FIFO mode, the earliest written data is sent, and in priority mode, the highest priority data is sent</p> <p>After this bit is written to 1, it will remain 1 until the STB transmission is completed or the transmission is canceled by TSA. Software cannot clear this bit by writing a 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&TTBMM=1 	R/W
b1	TSALL	Send all STB data set bits	<p>Transmit Secondary ALL frame 0: do not send 1: Send all data in STB</p> <p>After this bit is written to 1, it will remain 1 until the STB transmission is completed or the transmission is canceled by TSA. Software cannot clear this bit by writing a 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&TTBMM=1 	R/W
b0	TSA	STB sends cancel bit	<p>Transmit Secondary Abort 0: do not cancel 1: Cancel the STB transmission that has been requested by TSONE or TSALL but has not yet started</p> <p>This bit is written 1 by software but cleared by hardware. Write 1 to clear TSONE or TSALL bit.</p> <p>TSA is reset to the reset value by hardware in the following cases:</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 	R/W

30.5.5 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	Marking	Place name	Function	Read and write
b7	Reserved	-	The reset value must be maintained.	R
b6	TSNEXT	Next STB SLOT	Transmit buffer Secondary NEXT 0: no action 1: The current STB SLOT is filled, pointing to the next SLOT After the application program finishes writing the data in TBUF, it indicates that the current STB SLOT has been filled by setting the TSNEXT bit, so that the hardware points TBUF to the next STB SLOT. The data in the STB SLOT identified by the TSNEXT bit can be sent through the TSONE or TSALL bit. This bit is written to 1 by the application program and cleared to 0 by hardware. After all STB SLOTS are filled, TSNEXT remains 1 until a STB SLOT is released. Note: This bit is fixed to 0 in TTCAN mode.	R/W
b5	TSMODE	STB send mode	Transmit buffer Secondary operation MODE 0: FIFO mode 1: priority mode FIFO mode is sent according to the order in which data frames are written. The priority mode is automatically judged according to the ID. The smaller the ID, the higher the priority. Regardless of the mode, PTB has the highest priority. Note: The TSMODE bit can only be set when the STB is empty.	R/W
b4	TTTBM	TTCAN BUF mode	TTCAN Transmit Buffer Mode When TTEN=0, TTTBM is ignored. 0: TSMODE decision, PTB and STB 1: Set by TBPTR and TTPTR In TTCAN mode, this bit can be set to 0 when only receiving the time stamp function is required, and TSMODE is used to determine whether to use PTB or STB. Note: The TSMODE bit can only be set when the STB is empty.	R/W
b3~b2	Reserved	-	The reset value must be maintained.	R

Bit	Marking	Place name	Function	Read and write
b1~b0	TSSTAT	STB state	Transmission Secondary STATus bits TTEN=0 or 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 full 10: reserved 11: PTB and STB full	R

30.5.6 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	Marking	Place name	Function	Read and write
b7	SACK	Self-response	Self-ACKnowledge 0: no self-response 1: When LBME=1, enable the self-acknowledgment function	R/W
b6	ROM	Receive BUF overflow mode setting bit	Receive buffer Overflow Mode 0: The earliest received data is overwritten 1: Newly received data is not stored	R/W
b5	ROV	Receive BUF overflow flag bit	Receive buffer OVerflow 0: no overflow 1: overflow, at least one data is lost Cleared by writing RREL to 1.	R
b4	RREL	Release receive BUF	Receive buffer RELease 0: do not release 1: Indicates that the receiving BUF has been read, and the RBUF register points to the next RB SLOT.	R/W
b3	RBALL	Receive BUF data to store all data frames	Receive Buffer stores ALL data frames 0: Normal mode 1: Store all data including error data.	R/W
b2	Reserved	-	The reset value must be maintained.	R
b1~b0	RSTAT	Receive BUF status	Receive buffer STATus 00: RBUF empty 01: RBUF is not empty but less than AFWL programming value 10: RBUF is greater than or equal to the programmed value of AFWL but not full 11: full (keep this value on overflow)	R

30.5.7 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	Marking	Place name	Function	Read and write
b7	RIE	Receive interrupt enable	Receive Interrupt Enable 0: Prohibited 1: Enable	R/W
b6	ROIE	Receive overflow interrupt enable	Receive Overrun Interrupt Enable 0: Prohibited 1: Enable	R/W
b5	RFIE	Receive BUF full interrupt enable	RB Full Interrupt Enable 0: Prohibited 1: Enable	R/W
b4	RAFIE	Receive BUF will be full interrupt enable	RB Almost Full Interrupt Enable 0: Prohibited 1: Enable	R/W
b3	TPIE	PTB transmit interrupt enable	Transmission Primary Interrupt Enable 0: Prohibited 1: Enable	R/W
b2	TSIE	STB transmit interrupt enable	Transmission Secondary Interrupt Enable 0: Prohibited 1: Enable	R/W
b1	EIE	Error interrupt enable	Error Interrupt Enable 0: Prohibited 1: Enable	R/W
b0	TSFF	Send BUF full flag	TTEN=0 or TTTBM=0: Transmit Secondary buffer Full Flag 0: STB SLOT is not fully filled 1: STB SLOT is fully filled TTEN=1 and TTTBM=1: Transmit buffer Full Flag 0: The sending BUF selected by TBPTR is not fully filled 1: The sending BUF selected by TBPTR is fully filled	R

30.5.8 CAN Receive and Transmit Interrupt Status 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	Marking	Place name	Function	Read and write
b7	RIF	Receive interrupt flag	Receive Interrupt Flag 0: No data frame received 1: A valid data frame or remote frame is received Write 1 to clear 0 by application program.	R/W
b6	ROIF	Receive overflow interrupt flag	Receive Overrun Interrupt Flag 0: No RB is overwritten (overwrite) 1: At least one RB is covered Both ROIF and RFIF are set to 1 on overflow. Write 1 to clear 0 by application program.	R/W
b5	RFIF	Receive BUF full interrupt flag	RB Full Interrupt Flag 0: RB FIFO is not full 1: RB FIFO is full Write 1 to clear 0 by application program.	R/W
b4	RAFIF	Receive BUF will be full interrupt flag	RB 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 0 by application program.	R/W
b3	TPIF	PTB send interrupt flag	Transmission Primary Interrupt Flag 0: No PTB send completed 1: The requested PTB send completed successfully Write 1 to clear 0 by application program. Note: In TTCAN mode, TPIF is invalid, only the TSIF flag is applicable	R/W
b2	TSIF	STB send interrupt flag	Transmission Secondary Interrupt Flag 0: No STB transmission completed 1: The requested STB send completed successfully Write 1 to clear 0 by application program. Note: In TTCAN mode, TPIF is invalid, only use TSIF flag	R/W

Bit	Marking	Place 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 error counter value and the ERROR warning limit setting value has not changed.</p> <p>1: The BUSOFF bit changes, or the relative relationship between the value of the error counter and the setting 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 clear 0 by application program.</p>	R/W
b0	AIF	Cancel send interrupt flag	<p>Abort Interrupt Flag</p> <p>0: Send data is not canceled</p> <p>1: The send message requested by TPA and TSA was successfully canceled.</p> <p>Write 1 to clear 0 by application program.</p>	R/W

30.5.9 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	Marking	Place name	Function	Read and write
b7	EWARN	Reached the set ERROR WARNING LIMIT	Error WARNING limit reached 0: RECNT or TECNT is less than the set value of EWL 1: RECNT or TECNT is greater than or equal to the EWL setting value Write 1 to clear 0 by application program.	R
b6	EPASS	Error passive	Error Passive mode active 0: The node is an active error node 1: The node is a passive error node	R
b5	EPIE	Error Passive Interrupt Enable	Error Passive Interrupt Enable 0: Prohibited 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: Change from error active to error passive or error passive to error active Write 1 to clear 0 by application program.	R/W
b3	ALIE	Arbitration failure interrupt enable	Arbitration Lost Interrupt Enable 0: Prohibited 1: Enable	R/W
b2	ALIF	Arbitration Lost Interrupt Flag	Arbitration Lost Interrupt Flag 0: Arbitration is successful 1: Arbitration failure Write 1 to clear 0 by application program.	R/W
b1	BEIE	Bus Error Interrupt Enable	Bus Error Interrupt Enable 0: Prohibited 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 0 by application program.	R/W

30.5.10 CAN Bit Timing Register (CAN_BT)

Offset address: 0xA8

Reset value: 0x0102 0203

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PRESC[7:0]								-	SJW[6:0]						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	SEG_2[6:0]								SEG_1[7:0]						

Bit	Marking	Place name	Function	Read and write
b31~b24	PRESC	Prescaler	Prescaler When setting the value of this register, divide the module BCLK setting bit (PRESC+1) as the clock of TQ.	R/W
b23	Reserved	-	The reset value must be maintained.	R
b22~b16	SJW	Resynchronization compensation width time setting	Bit Timing Segment 2 Resynchronization compensation width time=(SJW+1)*TQ	R/W
b15	Reserved	-	The reset value must be maintained.	R
b14~b8	SEG_2	Bit segment 2 time setting	Bit Timing Segment 2 Segment 2 time=(SEG_2+1)*TQ	R/W
b7~b0	SEG_1	Bit segment 1 time setting	Bit Timing Segment 1 Segment 1 time=(SEG_1+2)*TQ	R/W

30.5.11 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	Marking	Place name	Function	Read and write
b7~b5	KOER	Error category	Kind Of Error 000: no error 001: bit error 010: wrong form 011: Padding error 100: Response error 101: CRC error 110: Other errors 111: reserved The KOER bit is updated when there is an error, and the KOER bit remains unchanged when sending and receiving normally.	R
b4~b0	ALC	Lost Arbitration Position Capture	Arbitration Lost Capture When the arbitration fails, the ALC records the position when the arbitration fails in one frame of data.	R

30.5.12 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	Marking	Place name	Function	Read and write
b7~b4	AFWL	Receive BUF will be full Warning Limit	receive buffer Almost Full Warning Limit The setting range is 1~10. AFWL=0 is meaningless, treat it as AFWL=1.	R/W
b3~b0	EWL	Error Waring Limit programming value	Programmable Error Warning Limit Error Waring Limit= (EWL+1) *8. The value set in this register affects the EIF flag.	R/W

30.5.13 CAN Receive Error CouNT Register (CAN_RECNT)

Offset address: 0xB2

Reset value: 0x00

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

Bit	Marking	Place name	Function	Read and write
b7~b0	RECNT	Receive error counter	Receive Error CouNT The receive error counter increases or decreases according to the error count specified in the CAN protocol. There is no overflow in this counter, and 255 is the maximum value.	R

30.5.14 CAN Transmit Error CouNT Register (CAN_TECNT)

Offset address: 0xB3

Reset value: 0x00

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

Bit	Marking	Place name	Function	Read and write
b7~b0	TECNT	Send error counter	Transmit Error CouNT The sending error counter increases or decreases according to the error count specified in the CAN protocol. There is no overflow in this counter, and 255 is the maximum value.	R

30.5.15 CAN Acceptance Filter Control Register (CAN_ACFCTRL)

Offset address: 0xB4

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-		SELMASK	-			ACFADR	

Bit	Marking	Place name	Function	Read and write
b7~b6	Reserved	-	The reset value must be maintained.	R
b5	SELMASK	Select the mask register for the filter	SElect acceptance MASK 0: ACF points to the filter ID register 1: ACF points to the filter MASK register Select a specific filter register set by ACFADR	R/W
b4	Reserved	-	The reset value must be maintained.	R
b3~b0	ACFADR	Filter address	acceptance filter address ACFADR points to a specific filter, and uses SELMASK to distinguish between ID and MASK. 0000: point to ACF_1 0001: point to ACF_2 0010: point to ACF_3 0011: point to ACF_4 0100: point to ACF_5 0101: point to ACF_6 0110: point to ACF_7 0111~1111: point to ACF_8	R/W

30.5.16 CAN Acceptance Filter Enable Register (CAN_ACFEN)

Offset address: 0xB6

Reset value: 0x01

b7	b6	b5	b4	b3	b2	b1	b0
AE_8	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1

Bit	Marking	Place name	Function	Read and write
b7	AE_8	ACF_8 enable	Acceptance Filter 8 Enable 0: Prohibited 1: Enable	R/W
b6	AE_7	ACF_7 enable	Acceptance Filter 7 Enable 0: Prohibited 1: Enable	R/W
b5	AE_6	ACF_6 enable	Acceptance Filter 6 Enable 0: Prohibited 1: Enable	R/W
b4	AE_5	ACF_5 enable	Acceptance Filter 5 Enable 0: Prohibited 1: Enable	R/W
b3	AE_4	ACF_4 enable	Acceptance Filter 4 Enable 0: Prohibited 1: Enable	R/W
b2	AE_3	ACF_3 enable	Acceptance Filter 3 Enable 0: Prohibited 1: Enable	R/W
b1	AE_2	ACF_2 enable	Acceptance Filter 2 Enable 0: Prohibited 1: Enable	R/W
b0	AE_1	ACF_1 enable	Acceptance Filter 1 Enable 0: Prohibited 1: Enable	R/W

30.5.17 CAN Acceptance Filter Code And Mask Register (CAN_ACF)

Offset address: 0xB8

Reset value: 0xFFFFFFFF

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	Marking	Place name	Function	Read and write
b31	Reserved	-	The readout value is indeterminate.	R
b30	AIDEE	IDE bit compare enable	Acceptance mask IDE bit check enable Valid only when SELMASK=1 0: Filter accepts standard format and extended format frames 1: The filter accepts standard format or extended format frames defined by the AIDE bits	R/W
b29	AIDE	IDE bit MASK	IDE bit MASK 0: filter only accepts standard formats 1: Filter only accepts extended formats	R/W
b28~b0	ACODE/ AMASK	Filter CODE/ Filter MASK	acceptance filter code Point to the specific filter via ACFADR. When SELMASK=0, it means the CODE of the filter. Use bit 10~bit 0 for the standard format, and bit 28~bit 0 for the extended format. acceptance filter mask Point to the specific filter via ACFADR. When SELMASK=1, it indicates the mask of the filter. Use bit 10~bit 0 for the standard format, and bit 28~bit 0 for the extended format.	R/W

30.5.18 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	Marking	Place name	Function	Read and write
b7	TBE	Set TB to empty	set TB slot to "empty" 0: No operation 1: The SLOT selected by TBPTR is marked as empty When SLOT is marked empty and TSFF=0, TBE is automatically reset to 0. If this bit is set to 1, there is data being sent in the selected SLOT, then TBE=1, then TBE is reset to 0 after the sending is completed, the sending is wrong, or the sending is canceled. TBE has higher priority than TBF.	R/W
b6	TBF	Set TB to filled	set TB slot to "Filled" 0: No operation 1: 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
b2~b0	TBPTR	TB SLOT pointer	Pointer to a TB message slot 000: point to PTB 001: point to STB SLOT1 010: point to STB SLOT2 011: point to STB SLOT3 100: point to STB SLOT4 Others: Prohibiting The pointed TB SLOT can be read and written through TBUF, and can be marked whether it has been filled through TBE and TBF. In TTCAN mode, the TBSEL and TSNEXT registers are invalid. Note: This bit can only be written when TSFF=0.	R/W

30.5.19 TTCAN TB Slot Pointer 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	Marking	Place name	Function	Read and write
b7	WTIE	Trigger gate interrupt enable	Watch Trigger Interrupt Enable 0: Prohibited 1: Enable	R/W
b6	WTIF	trigger gatekeeper interrupt flag	Watch Trigger Interrupt Flag When CYCLE COUNT value=TT_WTRIG setting value and WTIE=1, WTIF is set. Write 1 to clear 0 by application program.	R/W
b5	TEIF	Trigger error interrupt flag	Trigger Error Interrupt Flag When the set value of TT_TTIG is less than the actual CYCLE_TIME, TEIF is set. Write 1 to clear 0 by application program.	R/W
b4	TTIE	Time-triggered interrupt enable	Time Trigger Interrupt Enable 0: Prohibited 1: Enable	R/W
b3	TTIF	Time triggered interrupt flag	Time Trigger Interrupt Flag When CYCLE COUNT value = TT_TRIG set value and TTIE = 1, TTIF is set. If TT_TRIG is not updated, TTIF is only set once, and the next basic CYCLE is not set. Write 1 to clear 0 by application program.	R/W
b2~b1	T_PRESC	TTCAN counter prescaler	TTCAN Timer PREScaler 00: Divide by 1 the bit time set by the BT register 01: Divide by 2 the bit time set by the BT register 10: Divide by 4 the bit time set by the BT register 11: Eighth frequency division of the bit time set by the BT register Note: T_PRESC can be written when TTEN=0 or simultaneously when writing TTEN=1.	R/W
b0	TTEN	TTCAN enable	Time Trigger Enable 0: Prohibited 1: Enable TTCAN, the counter starts counting.	R/W

30.5.20 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_IDE 1	-	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	Marking	Place name	Function	Read and write
b31	REF_IDE	IDE bits for reference messages	REFERENCE message IDE bit 0: standard format 1: Extended format	R/W
b30~b29	Reserved	-	The readout value is indeterminate.	R
b28~b0	REF_ID	ID bit of the reference message	REFERENCE message IDentifier REF_IDE=0: REF_ID[10:0] is valid REF_IDE=1: REF_ID[28:0] is valid REF_ID is used to detect reference messages and is applicable for sending and receiving. After the reference message is detected, the Sync_Mark of the current frame becomes Ref_Mark. REF_ID[2:0] is fixed to 0, and its value is not checked, so that up to 8 potential time masters can be supported. After the highest byte of REF_MSG is written, it is necessary to wait for 6 CAN clock cycles to complete the transfer of REF_MSG to the CAN clock domain.	R/W

30.5.21 TTCAN Reference Message 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	Marking	Place name	Function	Read and write
b15~b12	TEW	Send enable window	Transmit Enable Window For the Single Shot Transmit Trigger mode (Single Shot Transmit Trigger) of TTCAN, a window of TEW+1 cycle time can be set, and transmission is only allowed within this window.	R/W
b11	Reserved	-	The reset value must be maintained.	R
b10~b8	TTYPE	Trigger type	Trigger Type 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: reserved 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: point to PTB 001: point to STB SLOT1 010: point to STB SLOT2 011: point to STB SLOT3 100: point to STB SLOT4 Others: Prohibiting If the pointed TB SLOT is marked empty, TEIF is set when the trigger time is reached.	R/W

30.5.22 TTCAN Reference Message 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	Marking	Place name	Function	Read and write
b15~b0	TT_TRIG	Trigger time	Trigger Time It is used to specify the cycle time of the trigger. For the sending trigger, the sending SOF time is about TT_TRIG setting value +1 When the highest byte of TT_TRIG is written, the TT_TRIG value begins to be transferred to the CAN clock domain. Therefore, if BYTE is operated, it is necessary to write the low byte first and then the high byte.	R/W

30.5.23 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	Marking	Place name	Function	Read and write
b15~b0	TT_WTRIG	Trigger time	Trigger Time Used to specify the cycle time of the gate trigger. When the highest byte of TT_WTRIG is written, the TT_WTRIG value begins to be transferred to the CAN clock domain. Therefore, if BYTE is operated, it is necessary to write the low byte first and then the high byte.	R/W

30.6 Precautions for Use

30.6.1 CAN Bus Anti-Jamming Measures

The CAN bus is widely used in automobiles, 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, and grid incoming lines, which will cause a large amount of communication noise on the CAN bus 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, efforts should be made to eliminate noise interference and ensure the stable operation of the CAN bus network.

The following are several commonly used CAN bus anti-interference measures (including but not limited to)

- 1) Increase electrical isolation of CAN bus interface
- 2) Common transceiver signal ground
- 3) Use shielded twisted pair cables and ground them properly
- 4) Improve twisted pair degree of CAN transmission line
- 5) Add signal protector
- 6) Improve network topology
- 7) Application layer software anti-jamming mechanism

30.6.2 CAN Controller Noise Constraints

In the CAN bus network, it should be ensured that the bit time of the communication meets the requirements of the standard protocol. If the noise interference that does not meet the bit time width is introduced, it may cause the abnormal operation of the CAN controller .

31 USB2.0 Full Speed Module (USBFS)

31.1 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 a full-speed PHY is integrated inside the chip. In host mode, the USBFS controller supports full-speed (FS, 12Mb/s) and low-speed (LS, 1.5Mb/s) transceivers, while in device mode it only supports full-speed (FS, 12Mb/s) transceivers . The USBFS controller supports all transfer methods defined by the USB 2.0 protocol (control transfer, bulk transfer, interrupt transfer, and isochronous transfer).

31.2 Main Features of USBFS

There are three main categories: general features, host mode features, and device mode features.

31.2.1 General Features

- Built-in on-chip USB2.0 full-speed PHY
- Support host mode and device mode
- Supports FS SOF and low-speed "Keep-alive" tokens and has the following functions:
 - SOF pulse pin output function
 - The SOF pulse can be used as an event source inside the chip to trigger the work of TIMER, DMA and other modules
 - Configurable frame period
 - Configurable end-of-frame interrupt
- The module is embedded with 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
- 1.25KB of dedicated RAM with advanced FIFO control
 - The RAM space can be divided into different FIFOs for flexible and effective use of RAM
 - Each FIFO can store multiple packets
 - Dynamically allocate memory
 - The size of the FIFO can be configured as a non-power-of-two value, so that the storage unit can be used continuously
- No application intervention is required within one frame to achieve maximum USB bandwidth
- Can automatically determine the host mode or device mode according to the level of the ID line

31.2.2 Host Mode Features

- Host mode supports USB2.0 full speed (FS, 12Mb/s) and low speed (LS, 1.5Mb/s) transmission
- Need to generate VBUS voltage through external power chip
- Up to 12 host channels (pipes): each channel can be dynamically reconfigured to support any type of USB transfer
- Built-in hardware scheduler that can:
 - Store up to 8 interrupt plus isochronous transfer requests in a periodic hardware queue
 - Store up to 8 control plus bulk transfer requests in an aperiodic hardware queue
- Manages a shared RX FIFO, a periodic TX FIFO, and an aperiodic TX FIFO for efficient use of USB data RAM

31.2.3 Device Mode Properties

- The slave mode supports USB2.0 full speed (FS, 12Mb/s) transmission.
- 1 bidirectional control endpoint 0
- 5 OUT endpoints that can be configured to support bulk transfers, interrupt transfers, or isochronous transfers
- 5 IN endpoints that can be configured to support bulk transfers, interrupt transfers, or isochronous transfers
- Contains 6 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.

31.3 USBFS System Block Diagram

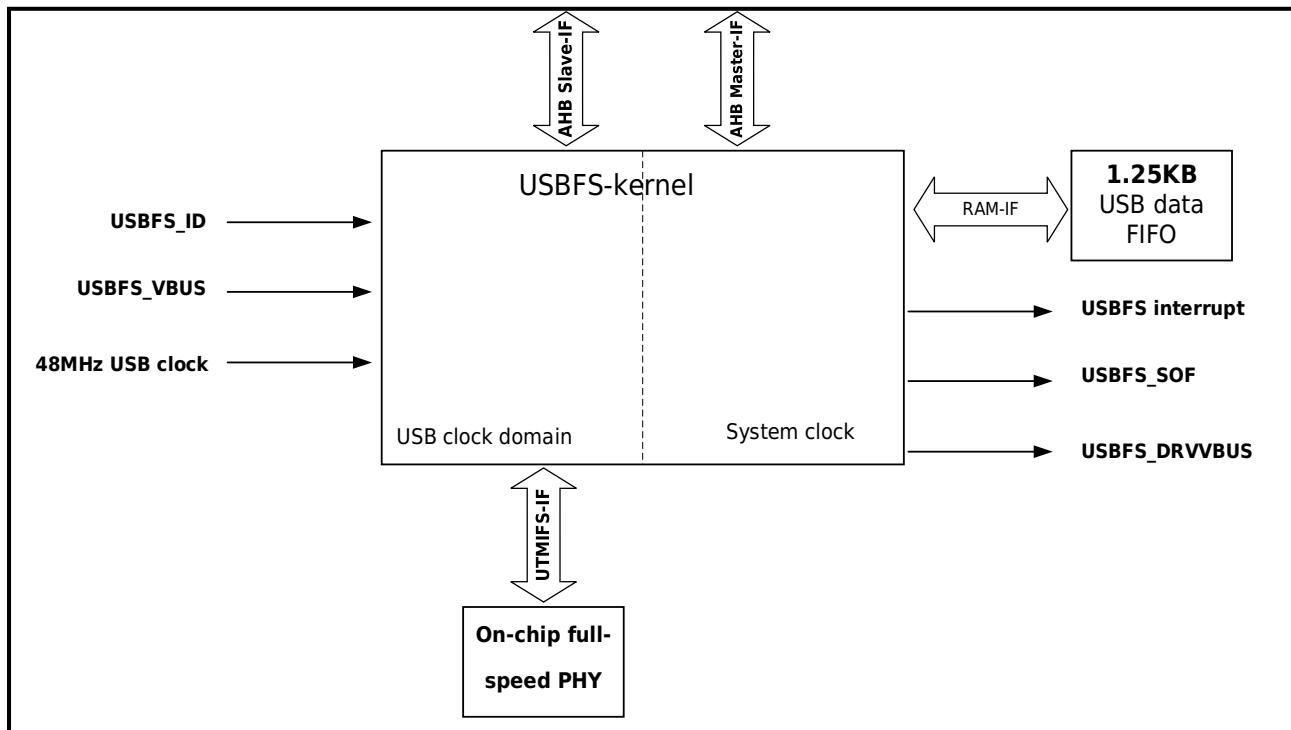


Figure 31-1 USBFS System Block Diagram

31.4 USBFS Pin Description

Table 31-1 USBFS Pin Descriptions

Pin name	Direction	Functional description
USBFS_VBUS	Input	Power port, 5V withstand voltage
USBFS_DP	Input/Output	Differential data D+ signal
USBFS_DM	Input/Output	Differential Data D-Signal
USBFS_DRVVBUS	Output	External power chip enable signal
USBFS_ID	Input	USB AB device identification signal
USBFS_SOF	Output	SOF output pulse signal

Since the USBFS_DP and USBFS_DM pins are multiplexed with general GPIO, when using USBFS, it is recommended to turn off the digital function of the corresponding pins, please refer to chapter General IO (GPIO) for details. In addition, when the USBFS function is not in use, when the digital function pins corresponding to the USBFS_DP and USBFS_DM pins are flipped, additional current consumption will be generated.

31.5 USBFS Function Description

31.5.1 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 a device and includes an on-chip full-speed PHY.

The on-chip full-speed PHY integrates pull-up and pull-down resistors, and USBFS can be automatically selected according to the current mode and connection status.

When USBFS is working, the VCC voltage range is 3.0~3.6V.

31.5.2 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 the USBFS_ID line is detected to be at a high level, the module will work in the device mode. When the USBFS_ID line is detected to be at a low level, the module will work 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, the module ignores the level of the USBFS_ID line and is forced to work in the device or host mode.

31.5.3 USBFS Host Function

31.5.3.1 Introduction to Host Functions

When USBFS works in the host mode, VBUS is the 5V power supply pin stipulated 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 supply chip, and the overcurrent detection of the external power supply chip can be realized through the external interrupt IRQ of this MCU. USB_VBUS can be used as GPIO in host mode.

A typical USB host mode system construction diagram is as follows:

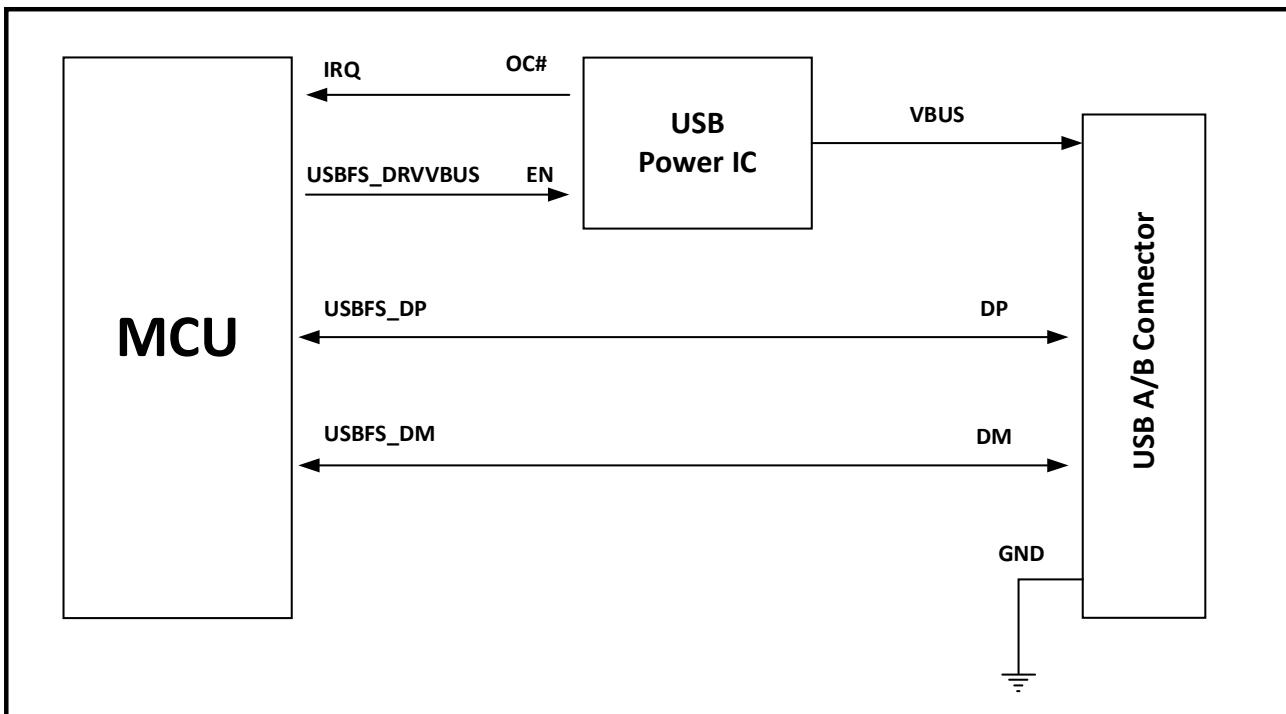


Figure 31-2 USBFS Host Mode System Construction Diagram

31.5.3.2 Host Port Power Supply

This MCU cannot output 5V to provide VBUS. To do this, a USB power chip or a basic power switch (such as an application board providing 5V power) 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 GPIO to control external devices to provide VBUS, the port power bit (PPWR bit in USBFS_HPRT) in the host port control and status register must still be set to 1.

31.5.3.3 Host Detects Device Connection And Disconnection

The USB device will be detected immediately after connection. The USBFS module will signal a host port interrupt, which is triggered by the device connection bit (PCDET bit in USBFS_HPRT) in the host port control and status register.

A device disconnect event will trigger a disconnect detection interrupt (DISCINT bit in USBFS_GINTSTS).

31.5.3.4 Host Enumeration

After a device connection is detected, if a new device is connected, the host must initiate the enumeration process by sending a USB reset and configuration command to the new device.

The application drives the USB reset signal (single-ended zero) over 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 10 ms and a maximum of 20 ms. The application calculates the duration of this process and then clears the port reset bit.

Immediately after the completion of the USB reset sequence, the port enable/disable change bit (PENCHNG bit in USBFS_HPRT) triggers a host port interrupt, which in turn notifies the application that the port speed field in the host port control and status register (in USBFS_HPRT PSPD) reads 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.

31.5.3.5 Host Hang

The application suspends the USB bus by setting the Port Suspend bit in the Host Port Control and Status Register (PSUSP in USBFS_HPRT). USBFS module stops sending SOF and enters suspend state.

The bus can be taken out of suspend by autonomous activity (remote wake-up) by a remote device. In this case, the remote wake-up signal will trigger the remote wake-up interrupt (the WKUPINT bit in USBFS_GINTSTS), and the hardware will automatically reset the port recovery bit in the host port control and status register (the PRES bit in USBFS_HPRT), and automatically drive the recovery through USB Signal. The application must time the resume window, then clear the port resume bit to exit suspend and restart SOF.

If the exit from suspend is initiated by the host, the application must set the port resume bit to initiate resume signaling on the host port, time the resume window and finally clear the port resume bit.

31.5.3.6 Host Channel

The USBFS module implements 12 host channels. Each host channel can be used for USB host transfers (USB pipes). The host can handle up to 8 transfer requests at the same time. If the application has more than 8 transfer requests pending, after the channel is released from the previous task (i.e. after receiving the transfer complete and channel stop interrupts), the host controller driver (HCD) must reallocate channels for outstanding transfer requests.

Each host channel can be configured to support input/output and periodic/aperiodic transactions. Each master channel uses dedicated control (HCCHARx) registers, transfer configuration (HCTSIZx) registers/interrupt (HCINTx) registers, and its associated interrupt mask register (HCINTMSKx).

Host Channel Control

The application can control the host channel as follows 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
- Sets the transfer direction on this channel: IN/OUT
- Set the type of USB transfer on this channel: control/bulk/interrupt/isochronous
- Set the maximum packet length for device endpoints communicating with this channel
- Set the frame to be transmitted periodically: odd-numbered frame/even frame

Host Channel Transport

The host channel transfer size registers (HCTSIZx) allow the application to program the transfer size parameters and read the transfer status. Setting this register must be done before setting the channel enable bit in the host channel characteristics register. After the endpoint is enabled, the packet count field immediately becomes read-only, and the USBFS module updates this field according to the current transmission 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 status of the endpoint on USB and AHB related events. 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 first read the host collective channel interrupt (HCAINT) register to obtain the channel number of the host channel x interrupt register. The application must clear the corresponding bits in this register 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 the data transfer
- Channel stopped due to transfer complete, USB transaction error, or application issued a prohibit command
- The associated transmit FIFO is half empty or fully 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 incorrect EOPs
- Crosstalk error
- Frame overflow
- Toggle bit error for data synchronization

31.5.3.7 Host Scheduler

The host module has a built-in hardware scheduler, which can independently reorder and manage the USB transaction requests sent by the application. At the beginning of each frame, the host performs periodic (synchronous and interrupt) transactions followed by aperiodic (control and bulk) transactions to comply with the USB specification's guarantee of high priority for isochronous and interrupt transfers.

The host handles 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 program but has not yet been responded to, and stores the number of the IN or OUT channel used to execute the USB transaction, as well as other related 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 the aperiodic request queue. If an isochronous or interrupt-type USB transfer transaction request scheduled for 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 USBFS_GINTSTS). The USBFS module is responsible for the management of periodic and aperiodic request queues. Both the periodic transmit FIFO and queue status register (HPTXSTS) and the aperiodic transmit FIFO and queue status register (HNPTXSTS) are read-only registers, and applications can use them to read the status of each request queue, including:

- The number of free entries currently available in the periodic (non-periodic) request queue (up to 8)
- Free space currently available in the periodic (aperiodic) TxFIFO (OUT transaction)

- IN/OUT tokens, host channel number, 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 completes.

To issue a transaction request to the host scheduler (queue), the application must read the PTXQSAV bit in the USBFS_HNPTXSTS register or the NPTQXSAV bit in the USBFS_HNPTXSTS register, ensuring that at least one of the periodic (non-periodic) request queues has space available to store the current ask.

31.5.4 USBFS Device Capabilities

31.5.4.1 Introduction to Device Functions

When USBFS works in device mode, VBUS is the 5V power supply pin stipulated by the USB protocol, and it is a 5V withstand voltage pin. This module always detects the level status of the VBUS line to connect or disconnect the device.

A typical USB device mode system construction diagram is as follows:

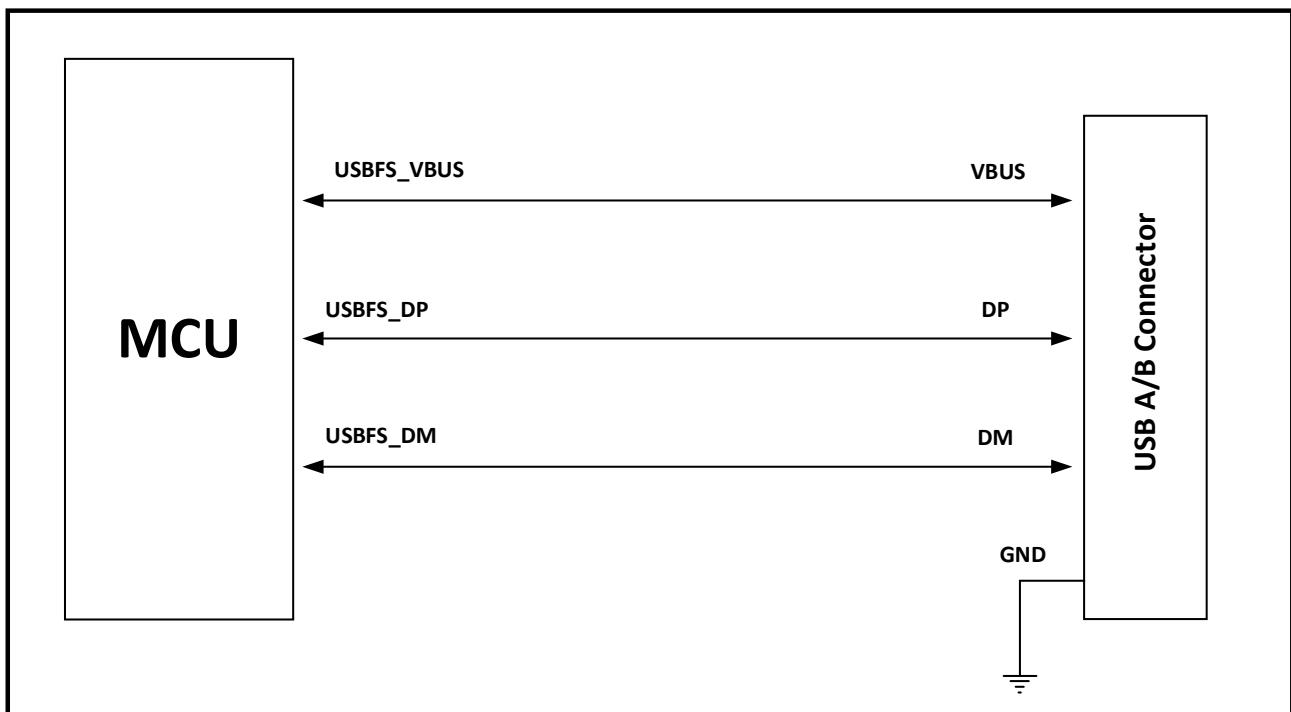


Figure 31-3 USBFS Device Mode System Construction Diagram

31.5.4.2 Device Power Status

When the module detects that USBFS_VBUS is high level, 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 to enter the power supply state.

In addition, 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, caused by power disturbance or host port shutdown).

In the powered state, USBFS expects to receive a reset signal from the host. Other USB operations cannot be performed. As soon as the reset signal is received, a reset detected interrupt (USBRST in USBFS_GINTST) is generated. After the reset signal ends, an enumeration complete interrupt (ENUMDNE bit in USBFS_GINTSTS) will be generated, and USBFS will then enter the default state.

31.5.4.3 Device Default State

By default, USBFS expects to receive 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 answer host transactions with the configured USB address.

31.5.4.4 Device Suspend State

The USBFS device continuously monitors USB activity. After the USB idle time reaches 3ms, an early suspend interrupt (ESUSP bit in USBFS_GINTSTS) will be issued, and the device will enter the suspend state confirmed by the suspend interrupt (USBSUSP bit in USBFS_GINTSTS) after 3ms. Then, the device suspend bit in the device status register (SUSPSTS bit in USBFS_DSTS) is automatically set to 1, and USBFS then enters the suspend state.

Suspend 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) to 1, and clear it to 0 within 1ms to 15m.

However, 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 will be automatically cleared.

31.5.4.5 Device Soft Disconnect

The powered state can be exited by software with the soft-disconnect function. The DP pull-up resistor can be removed by setting the soft disconnect bit (SDIS bit in USBFS_DCTL) in the device control register to 1. At this time, although the USB cable is not actually pulled out from the host port, the device disconnection will still occur on the host side Detect interruption.

31.5.4.6 Device Endpoint

Endpoint Class

The USBFS module implements the following USB endpoints:

- Control endpoint 0:
 - Bi-directional and only handles control messages
 - Use a separate set of registers for IN and OUT transactions
 - Dedicated Control (USBFS_DIEPCTL0/USBFS_DOEPCTL0) Register, Transport Configuration (USBFS_DIEPTSIZ0/USBFS_DOEPTSIZ0) Register, and Status Interrupt (USBFS_DIEPINT0/USBFS_DOEPINT0) Register. The set of bits available in the control and transfer size registers is slightly different from other endpoints
- 5 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_DIEPTSIz) registers, and status interrupt (USBFS_DIEPINTx) registers
- The device IN endpoint generic interrupt mask register (USBFS_DIEPMSK) can be used to enable/disable the same class of endpoint interrupt sources on all IN endpoints (including EP0)
- Supports incomplete isochronous IN transfer interrupt (IISOIXFR bit in USBFS_GINTSTS), which will trigger when at least one transfer on an isochronous IN endpoint is incomplete in the current frame. This interrupt is triggered together with the periodic frame interrupt (USBFS_GINTSTS/EOPF)
- 5 OUT endpoints
 - Each endpoint can be configured to support isochronous, bulk, or interrupt transfer types
 - Each endpoint has dedicated control (USBFS_DOEPCTLx) registers, transfer configuration (USBFS_DOEPTSIz) registers, and status interrupt (USBFS_DOEPINTx) registers
 - The device OUT endpoint general interrupt mask register (USBFS_DOEPMASK) can be used to enable/disable the same type of endpoint interrupt source on all OUT endpoints (including EP0)
 - Support for incomplete isochronous OUT transfer interrupts (INCOMPISOOUT bit in USBFS_GINTSTS), which will trigger when at least one transfer on an isochronous OUT endpoint is incomplete in the current frame. This interrupt is triggered together with the periodic frame interrupt (USBFS_GINTSTS/EOPF)

endpoint control

The application can take the following control over the endpoint through the device endpoint x IN/OUT control registers (DIEPCTLx/DOEPCTLx):

- Endpoint enable/disable
- Activate the endpoint under the current configuration
- Set USB transfer type (isochronous, bulk and interrupt)
- Set the supported packet size
- Sets the Tx-FIFO number associated with the IN endpoint
- Set the data0/data1 PID you want to receive or use when sending (bulk/interrupt transfers only)
- Set the odd/even frame for receiving or sending transactions (only for synchronous transfers)
- The NAK bit can be set, so that regardless of the state of the FIFO at this time, a NAK is replied to the host's request
- The STALL bit can be set so that the host's tokens for this endpoint are all replied to STALL by the hardware
- The OUT endpoint can be set to listen mode, that is, no CRC check is performed on the received data

Endpoint Transport

The Device Endpoint x Transfer Length Registers (DIEPTSI $_Z$ x/DOEPTSI $_Z$ x) allow the application to program the transfer length parameters and read the transfer status. Setting this register must be done before setting the Endpoint Enable bit in the Endpoint Control register. After the endpoint is enabled, these fields become read-only immediately, and the USBFS module updates these fields according to the current transmission status.

The following transmission parameters need to be configured:

- The length of a single transfer in bytes
- The number of packets that make up the entire transfer

Endpoint Status/Status

The device endpoint x interrupt registers (DIEPINTx/DOEPINTx) indicate the status of the endpoint on USB and AHB related events. When the OUT endpoint interrupt bit or the IN endpoint interrupt bit (OEPINT bit in USBFS_GINTSTS or IEPINT bit in USBFS_GINTSTS, respectively) is set in the module interrupt register, the application must read these registers for detailed information. Before an application can read these registers, it must first read the Device Overall Endpoint Interrupt (USBFS_DAI T) register to obtain the 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 DAI T and GINTSTS registers.

- The module provides the following status check and interrupt functions:
- Transfer complete interrupt, indicating to the application that both the 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 fully empty (IN endpoint)
- A NAK reply has been sent to the host (IN endpoint for isochronous transfers only)
- IN token received while TxFIFO is empty (only for IN endpoints of bulk and interrupt transfer types)
- OUT token received when the endpoint has not been enabled
- babble error detected
- Application shutdown endpoint takes effect
- The application sets NAK on the endpoint to take effect (only for the IN endpoint of the synchronous transmission type)
- Received more than 3 consecutive setup packets (only for control type OUT endpoints)
- A timeout condition was detected (only for IN endpoints of control transfer type)
- Packets of isochronous transfer type are lost without generating an interrupt

31.5.5 USBFS SOF Pulse Pin Output Function

USBFS can monitor, track and configure SOF frames in both host and device modes 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.

31.5.5.1 Host SOF

In host mode, the number of PHY clocks that occur between 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. An interrupt is 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 generated at the same time as 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 triggering DMA transfer, TIMER counting and other external modules.

31.5.5.2 Device SOF

In device mode, each time the USB receives a SOF token, it will trigger a start-of-frame interrupt (SOF bit in USBFS_GINTSTS). 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, you can also generate a SOF pulse signal with a width of 16 system clock cycles, and make the signal output on the USBFS_SOF pin for external availability.

In addition, the SOF pulse can also work as an internal event triggering DMA transfer, TIMER counting and other external modules.

The periodic end-of-frame interrupt (GINTSTS/EOPF) is used to notify the application when 80%, 85%, 90%, or 95% of the frame interval time has elapsed, depending on the periodic frame interval field in the device configuration register (USBFS_DCFG PFIVL bit in). This function can be used to determine if all isochronous communications for that frame are complete.

31.5.6 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.

Power reduction techniques can be used in the USB suspend state when the USB module is used but the device USB session is not started or the device is not connected.

- Stop the PHY clock (STPPCLK bit in USBFS_GCCTL)

Most of the 48 MHz internal clock domains of the USBFS full-speed module are clock-gated off when the Stop PHY Clock bit in the Clock-Gating Control Register is set to 1. Even if the application still provides the clock input, it will save the dynamic power consumption of the module due to the flipping of the clock signal. It will also turn off most of the units of the transceiver, and only the part responsible for detecting an asynchronous recovery event or a remote wake-up event will remain active. .

- HCLK gating (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 clock-gated off. Only the register read and write interfaces remain active. Even if the application still provides the clock input, it will save the module's dynamic power consumption due to the clock signal flipping.

To save dynamic power, the USB data FIFO is only clocked when it is accessed by the USBFS module.

31.5.7 USBFS Dynamically Updates the USBFS_HFIR Register

In 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 changes within the current SOF frame, the SOF period will be corrected accordingly in the next frame, see Figure 31-4 for details.

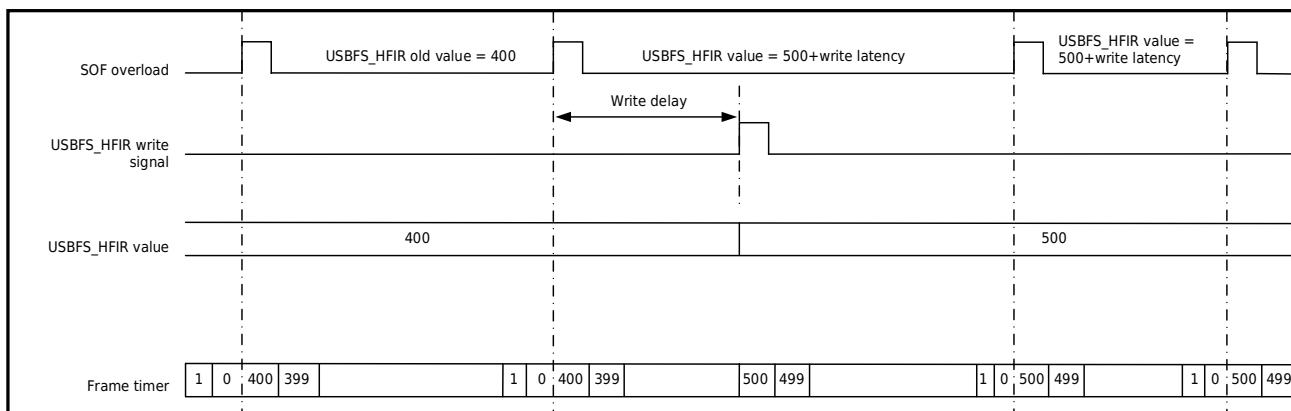


Figure 31-4 Schematic Diagram Of USBFS Dynamically Updating The USBFS_HFIR Register

31.5.8 USBFS Data FIFO

The USBFS system has 1.25KB dedicated RAM and adopts an efficient FIFO control mechanism. The packet FIFO controller module in the USBFS module divides the RAM space into multiple TxFIFOs (before USB transmission, the application program pushes data into it for short-term storage) and a single Rx FIFO (before the data received from the USB is read by the application program), where transient storage takes place).

The number and organization of FIFOs built in RAM depends on the role of the device. In device mode, one Tx FIFO is configured for each active IN endpoint. The sizes of the FIFOs are all software configurable to better meet application requirements.

31.5.9 USBFS Host FIFO Architecture

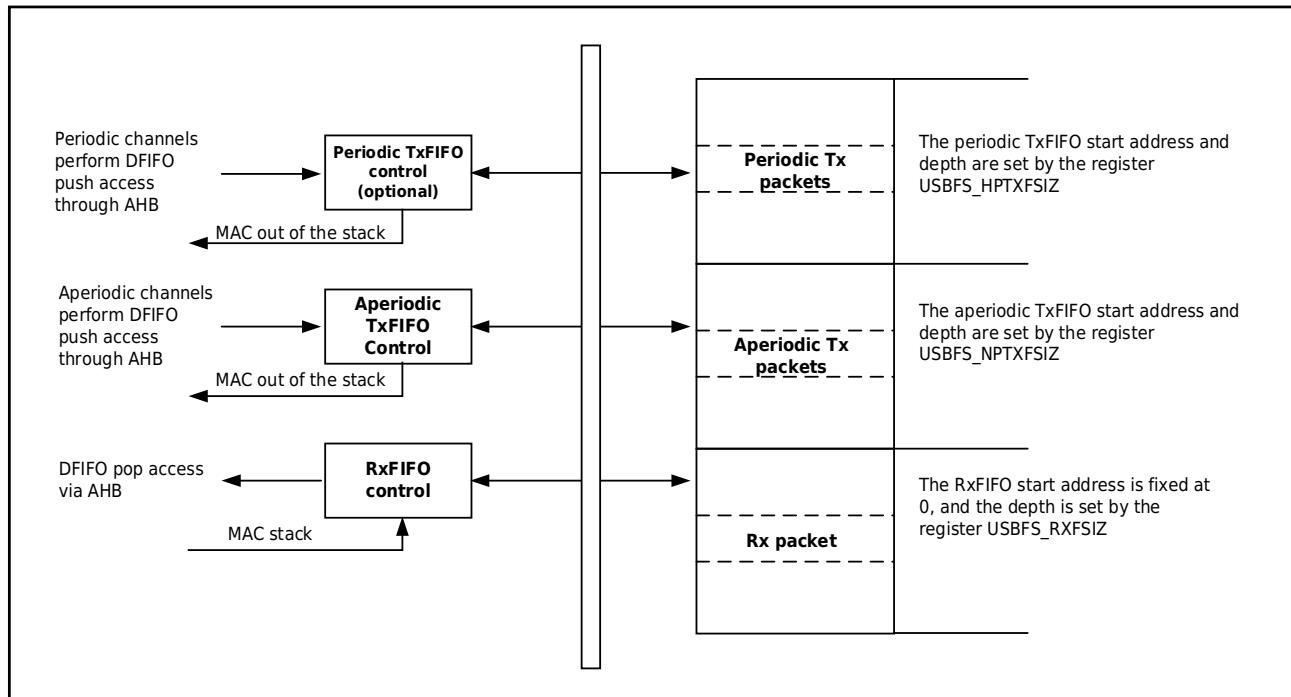


Figure 31-5 Schematic Diagram Of FIFO Architecture In USBFS Host Mode

31.5.9.1 Host RxFIFO

The host uses a receive FIFO for all periodic and aperiodic transactions. The FIFO is used as a receive buffer to hold data received from the USB (data portion of a received packet) until it is transferred to system memory. Packets from the IN endpoint of the device are received and stored one by one as long as there is room in the FIFO. The status of each packet received (including host target channel, byte count, data PID, and checksum on the received data) is also stored in the FIFO. The size of the receive FIFO is configured in the receive FIFO size register (GRXFSIZ).

A 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)
- The USBFS module can fill the receive FIFO to the limit for any sequence of IN tokens driven by host software

The application receives an Rx FIFO not empty interrupt as soon as at least one packet is available for reading in the RxFIFO. The application program reads the packet information from the receive status read and pop registers, and finally reads the data from the RxFIFO.

31.5.9.2 Host TxFIFO

The host uses one transmit FIFO for all aperiodic (control and bulk) OUT transactions and the other transmit FIFO for all periodic (synchronous and interrupt) OUT transactions. The FIFO is used as a transmit buffer to hold data to be sent over USB (transmit packets). The periodic (aperiodic) Tx FIFO size is configured in the host periodic (aperiodic) transmit FIFO size (HPTXFSIZ/HNPTXFSIZ) register.

The two Tx FIFOs are operated according to the priority, and the priority of the periodic communication is higher, so the periodic communication is performed first in 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 the aperiodic request queue.

The architecture of two transmit FIFOs enables the USB host to optimize the management of periodic and aperiodic transmit data buffers:

- All host channels configured to support periodic (aperiodic) OUT transactions share the same RAM buffer (shared FIFO)
- For arbitrary sequences 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 fully 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 of 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 fully empty, depending on the aperiodic TxFIFO empty level bit in the AHB configuration register (TXFELVL in USBFS_GAHBCFG bit). 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 of both can be known by reading the host aperiodic transmit FIFO and queue status register (HNPTXSTS).

31.5.10 USBFS Device FIFO Architecture

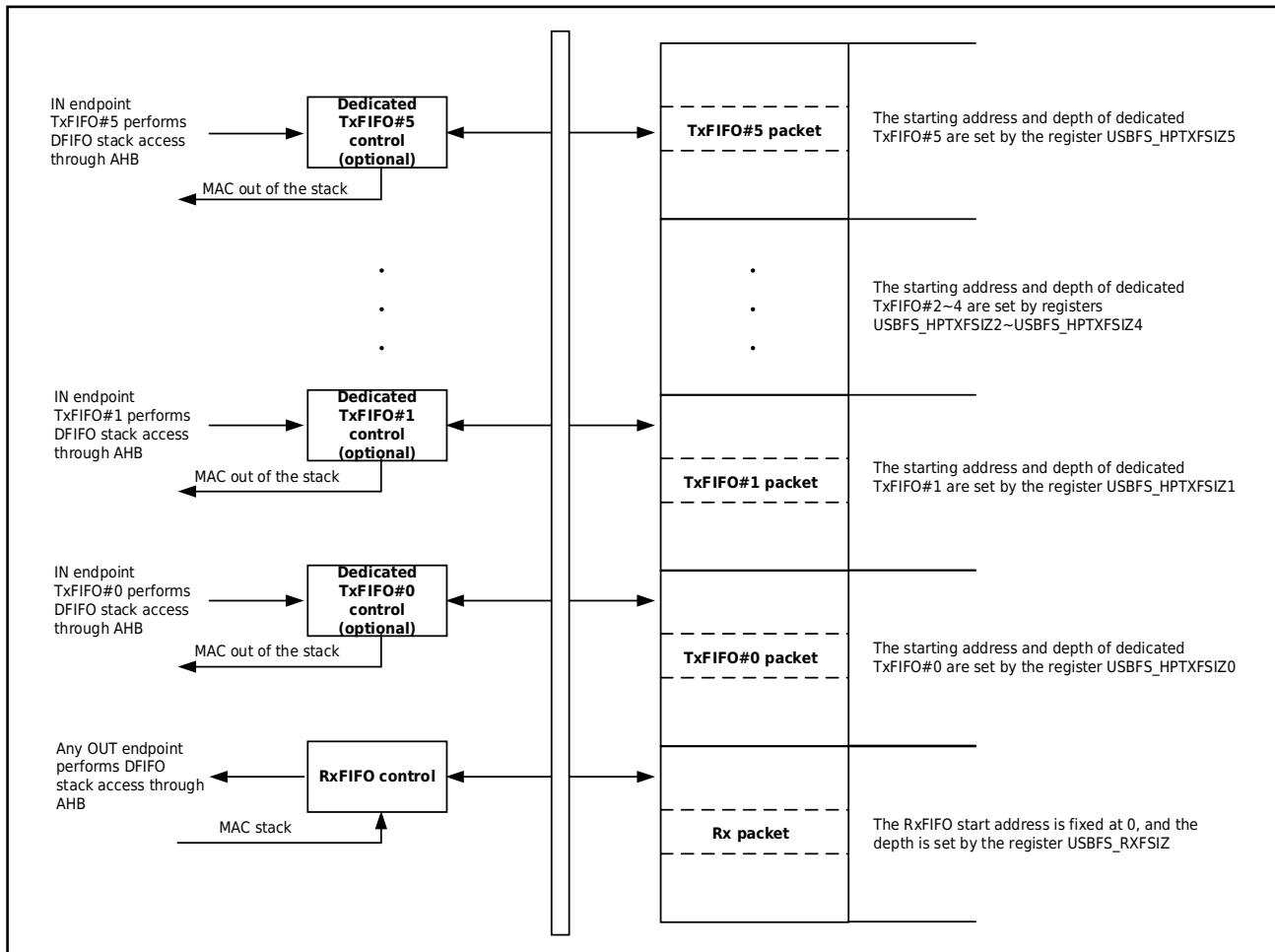


Figure 31-6 Schematic Diagram of FIFO Architecture in USBFS Device Mode

31.5.10.1 Device RxFIFO

USBFS devices use a single receive FIFO to receive data sent to all OUT endpoints. As long as there is free space in the Rx FIFO, the received data packets are filled into the Rx FIFO one by one. In addition to valid data, received packet status (containing 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 transaction NAK to the host 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 enables the USB device to more efficiently fill the receive RAM buffer:

- All OUT endpoints share the same RAM buffer (shared FIFO)
- The USBFS module can fill the receive FIFO to the limit for any host serial OUT token

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 stack register (GRXSTSP), and finally reads the corresponding data from the receive FIFO by reading the stack address related to the endpoint.

31.5.10.2 DeviceTxFIFO

The module provides a dedicated FIFO for each IN endpoint. The application configures the FIFO size for IN endpoint 0 through the aperiodic transmit FIFO size register (USBFS_DIEPTSIZE0); configures the FIFO size for IN endpoint x through the device IN endpoint transmit FIFOx register (DIEPTSIZEx).

31.5.11 USBFS FIFO RAM Allocation

31.5.11.1 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 received packets. If multiple isochronous channels are enabled, the space allocated for receiving consecutive packets must be at least twice (maximum packet size / 4) + 1. Usually, the recommended space is twice the size of (maximum packet/4 + 1), so that when the previous packet is sent to the CPU, the USB can receive subsequent packets at the same time.

The transfer completion status information is written to the FIFO along with the last packet received by this endpoint. So a place has to be allocated for this.

When running in DMA mode, the DMA address registers of each host channel will be stored in the FIFO, so a place needs to be reserved in the FIFO for each channel to store its address registers.

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.

Usually, the recommended space is twice the maximum packet size, so that the AHB can fill the transmit FIFO with the next packet while the USB is sending the current packet.

The minimum RAM required for the host aperiodic transmit FIFO is the size of the largest packet transmitted on all supported aperiodic OUT channels. If there is at least one isochronous OUT endpoint, the space must be at least twice the size of the largest packet in that channel.

When running in DMA mode, the DMA address registers of each host channel will be stored in the FIFO, so a place needs to be reserved in the FIFO for each channel to store its address registers.

31.5.11.2 Device Mode

Receive FIFO RAM Allocation

The application should allocate RAM for SETUP packets: 11 slots 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 received packets. If multiple isochronous endpoints are enabled, the space allocated for receiving consecutive packets must be at least twice (maximum packet size/4)+1. Usually, the recommended space is twice the size of (maximum packet/4 + 1), so that when the previous packet is sent to the CPU, the USB can receive subsequent packets at the same time.

The transfer completion status information is pushed into the FIFO along with the last packet received by this endpoint. In general, it is recommended to assign a location for each OUT endpoint.

Transmit FIFO RAM Allocation

The minimum RAM space required for each IN endpoint's transmit FIFO is the maximum packet size for that particular IN endpoint.

31.5.12 USBFS System Performance

With large RAM buffer, highly configurable FIFO size, fast 32-bit FIFO access via AHB push/pop registers, and especially advanced FIFO control mechanism for optimal USB and system performance. In fact, regardless of the current USB sequence, USBFS can efficiently fill the available RAM space through this mechanism. With these features:

- The application has enough margin to calculate and correct the CPU load to optimize CPU bandwidth utilization:
 - The application program can accumulate a large amount of sending data first, and then send it out via USB
 - Allows sufficient time margin to read data from the receive FIFO
- The USB module can keep working at full speed, that is, to provide the maximum full-speed bandwidth (as much hardware as possible to run automatically, and as little software as possible to participate)
 - The USB module can accumulate a large amount of sending data in advance at its disposal, so that the USB data sending can be autonomously managed
 - There is a lot of empty space in the receive buffer, which can be automatically filled with data from the USB

Because the USBFS module can efficiently fill the 1.25KB RAM buffer and 1.25KB send/receive data is enough to meet the amount of data that can be accommodated in a full-speed frame, the USB

system can reach the maximum USB bandwidth within one frame without application program intervention.

31.5.13 USBFS Interrupt and Events

The global interrupt is the mainterrupt that the software needs to handle, and the flag bit of the global interrupt can be read in the USBFS_GINTSTS register.

Table 31-2 USBFS 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	Disconnect interrupt	Host	-
CIDSCHG	Connector ID Line Status Change Interrupt	Host or device	-
PTXFE	Periodic TxFIFO empty interrupt	Host	-
HCINT	Host channel interrupt	Host	-
HPRTINT	Host port interrupt	Host	-
DATAFSUSP	Data fetch pending	Equipment	-
IPXFR/INCOMPISOOUT	Incomplete periodic transmission / outstanding OUT synchronous 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 hang interrupt	Equipment	-
GONAKEFF	Global OUT NAK active interrupt	Equipment	-
GINAKEFF	Global aperiodic IN NAK active 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	-

31.6 USBFS Programming Model

31.6.1 USBFS Module Initialization

The application must perform the module initialization sequence.

Please refer to [31.5.2 USBFS Mode Decision] for the mode determination method.

This section describes the initialization process after the USBFS controller is powered on. Whether working in host mode or device mode, an application must follow an initialization sequence. Initialize all module global registers according to the module configuration:

1. Program the following fields in the USBFS_GAHBCFG register:
 - Global interrupt mask bit GINTMSK = 1
 - RxFIFO is 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.

31.6.2 USBFS Host Initialization

To initialize a module as a host, an application must perform the following steps:

1. Program the HPRTIINT in the USBFS_GINTMSK register to unmask.
2. Program the USBFS_HCFG register to select a full-speed host.
3. Program the PPWR 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. Programming the PRST bit in USBFS_HPRT to 1 issues a reset signal 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 bits in USBFS_HPRT to get the enumeration speed.
10. With 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 changes, a port reset must be performed.
12. Program the USBFS_GRXFSIZ register to select the receive FIFO size.
13. Program the USBFS_HNPTXFSIZ register to select the size and starting address of the

aperiodic transmit FIFO used for aperiodic communication transactions.

14. Program the USBFS_HPTXFSIZ register to select the size and starting address of the periodic communication transmit FIFO for periodic transactions.

To communicate with the device, system software must initialize and enable at least one channel.

31.6.3 USBFS Device Initialization

During power-up or after switching from host mode to device mode, the application must perform the following steps to initialize the module as a device.

1. Program the following fields in the USBFS_DCFG register:
 - Device speed
 - Non-zero length status OUT handshake signal
2. Program the USBFS_GINTMSK register to unmask the following interrupts:
 - USB reset
 - Enumeration complete
 - Early hang
 - USB suspend
 - SOF
3. Wait for the VBUSVINT interrupt in USBFS_GINTSTS, which means entering 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 about 10ms from the reception of this interrupt.
5. Wait for ENUMDNE interrupt in USBFS_GINTSTS. This interrupt signals 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 on Enumeration Completion.

At this point, the device is ready to accept SOF packets and perform control transfers on control endpoint 0.

31.6.4 USBFS DMA Mode

USB uses the AHB master interface for sending packet data (AHB to USB) and receiving 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 data buffers.

31.6.5 USBFS Host Programming Model

31.6.5.1 Channel Initialization

An application must initialize one or more channels before it can communicate with a connected device.

To initialize and enable a channel, the application must perform the following steps:

1. Program the USBFS_GINTMSK register to unmask the interrupt on the following bits:
 - The aperiodic transmit FIFO used for OUT transactions is empty (applicable when working at the pipeline transaction level and the packet count field is programmed with a value greater than 1).
 - The aperiodic transmit FIFO used for OUT transactions is half empty (applicable when working at the pipeline transaction level and the packet count field is programmed with a value greater than 1).
2. Program the USBFS_HAINTMSK register to enable 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 for 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 (used for the first OUT transaction or expected from the first IN transaction).
5. Program the USBFS_HCCHARx registers for the selected channel, specifying the device's endpoint characteristics such as type, speed, direction, etc. (A channel can be enabled by setting the channel enable bit only when the application is ready to send or receive packets).

31.6.5.2 Channel Stop

The application can disable any channel by programming the USBFS_HCCHARx registers by setting the CHDIS and CHENA bits to 1. This causes the USBFS host to clear any previous requests made 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 will not interrupt communication transactions already initiated on the USB.

Before disabling a channel, the application must ensure that there is free space in at least one of the aperiodic request queue (when disabling aperiodic channels) or the periodic request queue (when disabling periodic channels). The application can clear the request queue by programming the USBFS_HCCHARx register to set the CHDIS bit to 1 and clear the CHENA bit to 0 when the

request queue is full (before disabling the channel). The application will ban 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) on the same channel before receiving the channel stop signal.
2. Received a DISCINT (disconnect device) interrupt in USBFS_GINTSTS. (Application will disable all enabled channels).
3. The application aborted the transfer before it could complete normally.

In DMA mode, the application cannot stop the indivisible periodic transfer by overwriting the register.

31.6.6 USBFS Device Programming Model

31.6.6.1 Endpoint Initialization on USB Reset

1. Set the NAK bit to 1 for all OUT endpoints
 - In USBFS_DOEPCCTLx, 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 the status of the control OUT packet) + 10 words (for the SETUP packet).
 - 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 1 maximum packet size for Control Endpoint 0.
4. Program the following fields in the endpoint-associated registers to control OUT endpoint 0 to receive SETUP packets
 - STUPCNT=3 in USBFS_DOEPTSIZ0 (receive up to 3 consecutive SETUP packets)

At this point, all initialization required to receive the SETUP packet is complete.

31.6.6.2 Endpoint Initialization on USB Reset

1. In the enumeration complete interrupt (ENUMDNE in USBFS_GINTSTS), read the USBFS_DSTS register 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 on control endpoint 0.

31.6.6.3 Endpoint Initialization When a 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 issue IN packets for the status phase

31.6.6.4 Endpoint Initialization When a Setconfiguration/Setinterface Command is Received

This section describes what an application must do when it receives a SetConfiguration or SetInterface command in a SETUP packet.

1. When the SetConfiguration command is received, the application must program the endpoint registers to configure them with the characteristics of valid endpoints in the new configuration.
2. When a SetInterface command is received, the application must program the endpoint registers for the endpoint specified by the command.
3. An endpoint that was valid in a previous configuration or other setup is not valid in a new configuration or other setup. 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 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 is ready to receive and send any type of data packet

31.6.6.5 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 endpoints) or the USBFS_DOEPCTLx

- register (for OUT or bidirectional endpoints).
- Maximum Packet Size
 - USB active endpoint location 1
 - Endpoint initial data sync bit (for interrupt and bulk endpoints)
 - Endpoint type
 - TxFIFO number
2. Once the endpoint is activated, the module starts decoding tokens sent to the endpoint and replies with a valid handshake if the received token is valid.

31.6.6.6 Endpoint Deactivation

This section describes the steps required to decommission an existing endpoint.

1. In the endpoint to be deactivated, clear the USB active endpoint bit in the USBFS_DIEPCTLx register (for IN or bidirectional endpoint) or USBFS_DOEPCTLx register (for OUT or bidirectional endpoint).
2. When an endpoint is deactivated, the module ignores tokens sent to it, causing a USB timeout.

31.6.7 USBFS Operational Model

31.6.7.1 SETUP and OUT Data Transfer

This section describes the internal data flow and application operation steps during data OUT transfers and SETUP transactions.

Packet Reading

This section describes how to read packets (OUT data and SETUP packets) from the receive FIFO.

1. When an RXFNE interrupt is caught (USBFS_GINTSTS register), the application must read the receive status pop 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 out of the receive FIFO.
3. If the received data packet has a non-zero byte count, it is popped from the received 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 state of the data packet read from the receive FIFO has the following states:
 - Global OUT NAK:
PKTSTS = global OUT NAK, BCNT = 0x000, the values of EPNUM and DPID don't matter.
These data indicate that the global OUT NAK bit is in effect.
 - SETUP packet:
PKTSTS=SETUP, BCNT=0x008, EPNUM=control EP number, DPID=D0. These data indicate

that SETUP packets 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 doesn't matter.

These data indicate that the build phase for the specified endpoint is complete and the data phase has started. After this status entry has been popped from the receive FIFO, the module will generate a setup interrupt on this control OUT endpoint.

- OUT packet:

PKTSTS=DataOUT, BCNT=the size of the received OUT packet (BCNT:0~1024), EPNUM=the endpoint number of the received data packet, DPID=actual data PID.

- Data transfer complete:

PKTSTS = OUT data transfer complete, BCNT = 0x0, EPNUM = OUT EP number that completed data transfer, DPID value does not matter.

These data indicate the completion of OUT data transfer for the specified OUT endpoint.

After this status entry has been 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 will be repeated each time the application detects an RXFNE interrupt in USBFS_GINTSTS. Reading from 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 applications process SETUP transactions.

Application Requirements:

1. To receive SETUP packets, the Control OUT endpoint STUPCNT field (USBFS_DOEPTSIZ0) 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_DOEPCTL0. Every time the control endpoint receives a SETUP packet, the STUPCNT field will be decremented. If the STUPCNT field is not programmed to an appropriate value prior to 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 to receive during the setup phase of a control transfer .

- In USBFS_DOEPTSIZ0, 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 section of FIFO is only used to store SETUP packets, 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 NAK and STALL bit settings of the endpoint.
- The module will internally set the IN NAK and OUTNAK bits of the control IN/OUT endpoint that received the SETUP packet to 1.
2. For each SETUP packet received on the USB, the module will write 3 words of data into the receive FIFO, and decrement 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 indicating 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 interrupt the application with a STUP interrupt (USBFS_DOEPINTx), 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_DOEPTSIZ0 register.
 - STUPCNT=3

2. Wait for RXFNE interrupt (USBFS_GINTSTS) and read packet from receive FIFO.
3. The triggering of the STUP interrupt (USBFS_DOEPINTx) indicates the successful completion of the 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 received SETUP packet.

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 USB2.0 specification does not limit the number of consecutive SETUP packets that the 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 into the receive FIFO. Regardless of the size of the available space in the receive FIFO, the device will reply NAK to the asynchronous OUT token sent by the host, and ignore the synchronous OUT packet directly.
2. The module writes the global OUT NAK to the receive FIFO. The application must allow enough space for this.
3. The module will set the GONAKEFF interrupt (USBFS_GINTSTS) when the application pops the global OUT NAK word from the receive FIFO.
4. After the application program detects this interrupt, it will consider 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 a 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 GONAKEY interrupt (USBFS_GINTSTS).
- In USBFS_DCTL, CGONAK=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. Before disabling any OUT endpoints, the application must enable global OUT NAK mode in the module.
 - In USBFS_DCTL, SGONAK=1
2. Wait 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 EPDISD interrupt (USBFS_DOEPINTx), which indicates that the OUT endpoint has been completely disabled. When an EPDISD 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

General Asynchronous OUT Data Transfer

This section describes a general asynchronous OUT data transfer (control, bulk or interrupt).

Application Requirements:

1. Before establishing an OUT transfer, the application must allocate a buffer in memory to hold all the data to be received as part of the OUT transfer.
2. For OUT transfers, the transfer size field in the endpoint's transfer size register must be a multiple (and word-aligned) of the endpoint's maximum packet size.
 - Transfer Size[EPNUM] = n × (MPSIZ[EPNUM] + 4 - (MPSIZ[EPNUM] mod 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 valid data in memory. The amount of valid data received may be less than the programmed transfer size.
- Effective data amount in memory = initial transfer amount set by application program - remaining transfer amount after module update
- The number of received USB data packets = the initial data packets set by the application - the remaining data packets after the module is updated

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. Once the NAK bit is cleared, the module starts receiving data and writing 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. Every packet written to the receive FIFO (packets up to the maximum packet size or short packets) decrements the packet count field for this endpoint by 1.
 - If the CRC of the received data packet is invalid, it will be automatically cleared from the receive FIFO.
 - After replying with an ACK for the packet on the USB, the module will discard the non-synchronized OUT packet resent by the host because the ACK cannot 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 is not decremented.
 - If there is no room in the receive FIFO, isochronous or asynchronous packets are ignored and not written to the receive FIFO. Additionally, non-synchronized OUT tokens will receive a NAK handshake response.
 - In all three cases above, the packet count is not decremented because no data is written to the receive FIFO.
3. When the packet count goes to 0 or a short packet is received on an endpoint, the NAK bit for that endpoint will be set. After the NAK bit is set, synchronous or asynchronous packets will be ignored and will not be written to the receive FIFO, while the non-synchronous OUT token will receive a NAK handshake response.
4. After the data is written into 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 data packet is written to the external memory on AHB, the transfer size of the

- endpoint is automatically subtracted from the size of the data packet.
6. The OUT data transfer completion status of the OUT endpoint will be written to the receive FIFO when:
 - transfer size is 0 and packet count is 0
 - The last OUT packet written to the receive FIFO is a short packet
(packet size: 0~max. packet size-1)
 7. When the application program pops up this status entry (OUT data transfer is completed), and generates the transfer completion interrupt of this endpoint, and clears the endpoint enable bit at the same time.

Application programming sequence:

1. Program the USBFS_DOEPTSIZx registers with the transfer size and corresponding number of packets.
2. Program the USBFS_DOEPCTLx registers with the endpoint characteristics and set the EPENA and CNAK bits.
 - In USBFS_DOEPCTLx, EPENA=1
 - In USBFS_DOEPCTLx, CNAK=1
3. Wait for the RXFNE interrupt (in USBFS_GINTSTS) and read the packet from the receive FIFO.
 - This step can be repeated multiple times, depending on the transfer size.
4. An XFRC interrupt (USBFS_DOEPINTx) is triggered to indicate the successful completion of the asynchronous OUT data transfer.
5. Read the USBFS_DOEPTSIZx registers to determine the amount of valid data.

Universal Synchronous OUT Data Transfer

This section describes regular synchronous OUT data transfers.

Application Requirements:

1. All application requirements for asynchronous OUT data transfers apply to synchronous 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 largest packet size that can be received in a single frame. OUT data transfer transactions of synchronous type must be completed within one frame.
3. The application must read all isochronous OUT packets (data entries and status entries) from the receive FIFO before the end of the periodic frame (EOPF interrupt in USBFS_GINTSTS).

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 a non-synchronous OUT endpoint, but there are some differences.
2. When a synchronous OUT endpoint is enabled by setting the endpoint enable bit and clearing the NAK bit, the even/odd-numbered frame bit must be set accordingly. The module will receive data in a particular frame on the synchronous OUT endpoint only if the following conditions are met:
 - EONUM (in USBFS_DOEPCTLx) = FNSOF[0] (in USBFS_DSTS)
 - 3. When the application completely reads a synchronous OUT packet (data and status) from the receive FIFO, the module will update the RXDPID field in USBFS_DOEPTSIZx according to the data PID of the last synchronous 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 with endpoint characteristics and set the endpoint enable bit, clear NAK bit, and odd/even frame bit.
 - EPENA=1
 - CNAK=1
 - EONUM=(0: Even/1: Odd)
3. Wait for the RXFNE interrupt (in USBFS_GINTSTS) and read the packet from the receive FIFO.
 - This step can be repeated multiple times, depending on the transfer size.
4. XFRC interrupt (in USBFS_DOEPINTx) indicates completion of synchronous OUT data transfer. This interrupt does not necessarily mean that the data in memory is valid.
5. For synchronous OUT transfers, the application may not always detect this interrupt. Instead, the application may detect the IISOXXFRM interrupt in USBFS_GINTSTS.
6. Read the USBFS_DOEPTSIZx registers to determine the size of the transfer received and to determine the validity of the data received in the frame. The application must consider data received in memory as valid data only if one of the following conditions is true:
 - RXDPID=D0 (in USBFS_DOEPTSIZx) and the number of USB packets receiving this valid data=1
 - RXDPID=D1 (in USBFS_DOEPTSIZx) and the number of USB packets receiving this valid data=2

- RXDPID=D2 (in USBFS_DOEPTSIZx) and the number of USB packets receiving this valid data=3

The number of USB data packets receiving the valid data = the initial data packet number of application programming - the remaining data packet number after the module is updated.

Applications can drop invalid packets.

Incomplete Synchronous OUT Data Transfer

This section describes the application programming sequence when a sync OUT packet is lost.

Internal data flow:

1. For synchronous OUT endpoints, XFRC interrupts (in USBFS_DOEPINTx) may not always be raised. If the module drops isochronous OUT packets, the application may not detect XFRC interrupts (USBFS_DOEPINTx) in the following cases:
 - When the receive FIFO cannot hold the complete ISO OUT packet, the module will discard the received ISO OUT data
 - Received sync OUT packet has CRC error
 - The sync OUT token received by the module is corrupted
 - Application reads data from receive FIFO very slowly
2. If the module detects the periodic end of frame before the transmission of all synchronous OUT endpoints is completed, it will trigger the incomplete synchronous OUT data interrupt (IISOXXFRM in USBFS_GINTSTS), indicating that at least one synchronous OUT endpoint has not triggered the XFRC interrupt (in USBFS_DOEPINTx). At this time, the endpoint that has not completed the transfer is still enabled, but on this endpoint of the USB, there is no valid transfer in progress.

Application programming sequence:

1. The hardware triggers the IISOXXFRM interrupt (USBFS_GINTSTS) to indicate that at least one isochronous OUT endpoint has an outstanding transfer in the current frame.
2. If this occurs because the isochronous OUT data was not fully read from the endpoint, the application must ensure that all isochronous OUT data (including data entries and status entries) is read from the receive FIFO before continuing processing.
 - Once all data has been read from the receive FIFO, the application can detect an XFRC interrupt (USBFS_DOEPINTx). In this case, the application must re-enable the endpoint to receive isochronous OUT data in the next frame.
3. When the application receives an IISOXXFRM interrupt (in USBFS_GINTSTS), the application must read the control registers (USBFS_DOEPCTLx) of all synchronous OUT endpoints to determine which endpoints have incomplete transfers in the current frame. An endpoint transfer is incomplete when both of the following conditions are met:

- EONUM bit (in USBFS_DOEPCTLx) = FNSOF[0] (in USBFS_DSTS)
 - EPENA=1 (in USBFS_DOEPCTLx)
4. Before the SOF interrupt is detected (in USBFS_GINTSTS), the previous step must be performed to ensure that the current frame number has not changed.
 5. For isochronous OUT endpoints with incomplete transfers, the application must discard data in memory and disable the endpoint by setting the EPDIS bit in USBFS_DOEPCTLx.
 6. Wait for an 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 inhibit the endpoint, the application may not be able to receive data in the next frame after receiving invalid sync data.

Stop an Asynchronous OUT Endpoint

This section describes how an application can stop an asynchronous endpoint.

1. Puts the module in global OUT NAK mode.
2. Disable 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. The STALL bit (in USBFS_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 the endpoint due to receiving the 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 the control endpoint.

31.6.7.2 IN Data Transmission

Packet Write

This section describes how the application writes packets to the endpoint FIFO when the dedicated transmit FIFO has been 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.
 - Interrupt mode, the application waits for the TXFE interrupt (in USBFS_DIEPINTx), 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 room in the data FIFO to hold the entire packet.
 - To write a zero-length packet, the application must not look into the FIFO space.
2. If one of the above methods is used, when the application determines that there is enough space to write the transmit data packet, the application must first write the appropriate write operation to the endpoint control register, and then write the data to the data FIFO. Typically, the application must perform a read-modify-write operation on the USBFS_DIEPCTLx register to avoid modifying other contents in the register while setting the endpoint enable bit to 1.

The application can write multiple packets for the same endpoint to the transmit FIFO if there is enough space. For periodic IN endpoints, applications can only write multiple packets within a frame at a time. The application writes all the packets to be sent in the next frame only after the transmission of the previous frame's communication transaction is complete.

Set IN endpoint NAK to 1

Internal data flow:

1. When the application program sets the IN NAK of a specific endpoint, the module will stop the data transmission on the endpoint regardless of whether the data in the endpoint transmit FIFO is available or not.
2. The asynchronous endpoint receives the IN token and replies with a NAK handshake response.
 - The synchronous endpoint receives the IN token and returns a 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. Once the application detects this interruption, it considers the endpoint to be 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 particular 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 INEPNE interrupt trigger in USBFS_DIEPINTx. This interrupt indicates that the module has stopped sending data on the endpoint.
3. The module can send valid IN data on the endpoint when the application program sets the NAK bit to 1 but the "NAK Valid" interrupt has not yet triggered.
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 action also clears INEPNE interrupts (in USBFS_DIEPINTx).
 - In USBFS_DIEPCTLx, CNAK=1
6. If the application has masked this interrupt, it must be unmasked as follows:
 - In DIEPMSK, INEPNEM=1

Disable IN Endpoints

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. The triggering of the EPDISD interrupt in USBFS_DIEPINTx indicates that the module has completely disabled the specified endpoint. At the same time 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 on the USB on the endpoint.
7. The application must clear the data in 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 Acyclic IN Data Transfer

Application Requirements:

1. Before establishing an IN transfer, the application must ensure that each packet that makes up an IN transfer fits in a single buffer.
 2. For IN transmission, the transfer size field in the endpoint transfer size register indicates the effective data volume 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 transfer.
- To send multiple packets of maximum packet size plus a short packet at the end of the transfer:

$$\text{Transfer Size[EPNUM]} = x \times \text{MPSIZ[EPNUM]} + \text{sp}$$

If ($\text{sp} > 0$), $\text{Packet Count[EPNUM]} = x + 1$.

Otherwise, $\text{Packet Count[EPNUM]} = x$

- To send a single zero-length packet:

$$\text{transfer_size[EPNUM]} = 0$$

$\text{Packet Count[EPNUM]} = 1$

- To send multiple packets of the maximum packet size plus 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 the maximum packet size, the second part sends only zero-length packets.

First transfer: $\text{transfer size[EPNUM]} = x \times \text{MPSIZ[epnum]}$; $\text{packet count} = n$;

Second transfer: $\text{transfer size[EPNUM]} = 0$; $\text{packet count} = 1$;

3. After enabling an endpoint for data transfer, the module updates the transfer size register. At the end of an IN transfer, the application must read the transfer size register to determine how much of the data that was put into the transmit FIFO has been sent out over the USB.

4. Amount of data put into transmit FIFO = initial transfer size programmed by application - final transfer size after module update

- Amount of data sent over USB = $(\text{initial packet count from application programming} - \text{final packet count after module update}) \times \text{MPSIZ[EPNUM]}$

- Amount of data remaining to be sent over USB = (Application programmed initial transfer size - 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 for a specific endpoint and enable that endpoint to send data.
2. The application must also write the necessary data to the endpoint's transmit FIFO.
3. Every time the application writes a packet to the send FIFO, the transfer size of the endpoint is automatically subtracted from the packet size. The application continues fetching 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 "Number of packets in FIFO" count will increase (this is a 3-bit count, which is maintained internally by the module, and each IN endpoint sends a FIFO corresponding to one. In the IN endpoint FIFO, the maximum number of packets maintained by the module is always eight). Each FIFO has a separate flag for zero-length packets, and there is no data in the FIFO.
4. After the data is written into the send FIFO, the module will send the data when it receives the IN token. After each data packet is sent out and the reply ACK handshake signal is received, the data packet count of the endpoint will be decremented by 1 until the data packet count becomes 0. The packet count is not decremented when a timeout occurs.
5. For a zero-length packet (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 that received the IN token, and the packet count field for that endpoint is zero, the module generates an "IN Token Received While TxFIFO Empty" (ITTXFE) interrupt for that endpoint (provided that the NAK bit for that endpoint is not set). The module replies with a NAK handshake signal on the asynchronous endpoint.
7. The module resets the FIFO pointers to the beginning internally and does 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 with 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.

General Periodic IN Data Transmission

This section describes typical periodic IN data transfers.

Application Requirements:

1. Application requirements 1, 2, 3, 4 for general acyclic IN data transmission are also applicable for cyclic IN data transmission (with a slight modification to requirement 2).

- The application can only send a number of packets of the maximum packet size or a number of packets of the maximum packet size, plus a short packet at the end of the transfer.

To send multiple packets of the maximum packet size plus a short packet at the end of the transfer, 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 the range of sp is 0 ~ MPSIZ[EPNUM]-1)

if (sp > 0), Packet Count[EPNUM] = x + 1

else, PacketCount[EPNUM] = x;

MCNT[EPNUM] = Packet Count[EPNUM]

- Applications cannot send zero-length packets at the end of a 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. Applications can only schedule data transfers for one frame at a time.

- $(\text{MCNT} - 1) \times \text{MPSIZ} < \text{XFERSIZ} \leq \text{MCNT} \times \text{MPSIZ}$

- PKTCNT = MCNT (in USBFS_DIEPTSIZx)

- If XFERSIZ < MCNT × MPSIZ, the last packet transmitted is a short packet

- Please note: MCNT is located in USBFS_DIEPTSIZx, MPSIZ is located in USBFS_DIEPCTLx, PKTCNT is located in USBFS_DIEPTSIZx, XFERSIZ is located in USBFS_DIEPTSIZx

3. Before receiving the IN token, the application must write the complete data to be sent in the frame into the transmit FIFO. When the IN token is received, even if the data to be sent in the frame in the sending FIFO is only one double word short of writing, 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

- NAK handshake signal will be replied on interrupt endpoint

Internal data flow:

1. The application must set the transfer size and packet count fields in the registers for 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 this endpoint.
3. Every time the application writes a packet to the send FIFO, the transfer size of the endpoint is automatically subtracted from the packet size. The application continues fetching 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 there is no complete packet of data to be sent for the frame in the FIFO, the module will generate an "IN token received while TxFIFO is empty" interrupt for the endpoint.
- A zero-length packet will be replied on the isochronous endpoint
- A NAK handshake signal will be replied on the interrupt endpoint
5. The packet count for an endpoint is decremented by 1 when:
 - For isochronous endpoints, when sending a zero-length or non-zero-length packet
 - For interrupt endpoints, decrements when sending ACK handshake
 - When the transfer size and packet count are both 0, a transfer complete interrupt for this endpoint is generated and the endpoint enable bit is 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 with the endpoint characteristics and set the CNAK and EPENA bits.
2. Write the data 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 yet written all the data that needs to be sent into the transmit FIFO.
4. If the interrupt endpoint is enabled before the interrupt is detected, the interrupt will be ignored. If the interrupt endpoint is not enabled, enable it so that data can be sent on the next IN token received.

5. If the hardware triggers the XFRC interrupt (in USBFS_DIEPINTx), if no ITTXFE interrupt is generated in USBFS_DIEPINTx, it means that the synchronous IN transfer is successfully completed. Reading the USBFS_DIEPTSIZx registers always gives transfer size = 0 and packet count = 0, which means all data has been sent over USB.
6. Regardless of whether an ITTXFE interrupt (in USBFS_DIEPINTx) is generated when the XFRC interrupt (in USBFS_DIEPINTx) is set, it indicates the successful completion of the interrupt IN transfer. Reading the USBFS_DIEPTSIZx registers always gets transfer size = 0 and packet count = 0, which means all data has been sent over USB.
7. If none of the aforementioned interrupts are generated when the incomplete synchronous IN transfer (IISOIXFR) interrupt is set in USBFS_GINTSTS, it means that the module has not received at least 1 periodic IN token in the current frame.

Synchronous IN Data Transfer Not Completed

This section describes what an application must do for outstanding isochronous 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 corrupt sync IN token on at least one sync IN endpoint. At this point, the application detects an incomplete isochronous IN transfer interrupt (IISOIXFR bit in USBFS_GINTSTS).
 - b) The application is writing data to the transmit FIFO too slowly, and the IN token is received before the complete data has been 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 eventually this will generate an incomplete isochronous IN transfer interrupt (IISOIXFR bit in USBFS_GINTSTS) at the end of a 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 NAK bit and disable bit of the endpoint.
4. The module disables the endpoint, clears the disable bit and triggers an "Endpoint Disabled" interrupt for the endpoint.

Application programming sequence:

1. Applications can ignore the "IN token received while TxFIFO is empty" interrupt in USBFS_DIEPINTx on any isochronous IN endpoint, as eventually this will generate an incomplete isochronous IN transfer interrupt (in USBFS_GINTSTS).
2. A hardware triggered outstanding isochronous IN transfer interrupt (in USBFS_GINTSTS) indicates that there is an outstanding isochronous IN transfer on at least one isochronous

IN endpoint.

3. The application must read the "Endpoint Control" register of all synchronous IN endpoints to detect endpoints with outstanding IN data transfers.
4. The application must stop writing to the "Periodic Transmit FIFO" associated with these endpoints.
5. Program the following fields in the USBFS_DIEPCTLx registers to disable the endpoint:
 - SNAK=1 in USBFS_DIEPCTLx
 - EPDIS=1 in USBFS_DIEPCTLx
6. The hardware triggers the "Endpoint Disabled" interrupt in USBFS_DIEPINTx to indicate that the module has disabled the endpoint.
- At this point, the application must empty the data in the associated transmit FIFO, or overwrite the existing data in the FIFO by enabling the endpoint for a new transmission in the next frame. To refresh data, the application must use the USBFS_GRSTCTL register.

Stop The Asynchronous IN Endpoint

This section describes how an application can stop an asynchronous endpoint.

Application programming sequence:

1. Inhibit IN endpoints to stop. Also set the STALL bit to 1.
2. EPDIS=1 in USBFS_DIEPCTLx (when the endpoint is enabled)
 - STALL=1 in USBFS_DIEPCTLx
 - The STALL bit always has priority over the NAK bit
3. The hardware triggers an "endpoint disabled" interrupt (in USBFS_DIEPINTx) to let 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 non-periodic endpoints, the application must re-enable another non-stop non-periodic endpoint to send data.
5. When the application is ready to end the STALL handshake for this endpoint, it must clear the STALL bit of USBFS_DIEPCTLx to 0.
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, it must set or clear the STALL bit before the state phase transfer of that control endpoint.

Special case: stop controlling the OUT endpoint

If, during the data phase of a control transfer, the host sends more IN/OUT tokens than the value specified in the SETUP packet, the module must reply STALL to these excess IN/OUT tokens. In this

case, the application must enable the ITTXFE interrupt for USBFS_DIEPINTx and the OTEPDIS interrupt for USBFS_DOEPINTx during the data phase of the control transfer (after the module has completed 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.

31.7 Register Description

The application program controls the USBFS module by reading and writing the control and status registers through the AHB slave interface. All registers of the USBFS module are 32-bit registers, and their addresses are aligned by 32 bits, so they can only be accessed in 32-bit mode.

The control and status registers are divided into the following categories:

- USBFS System Control Register
- Module global registers
- Host Mode Register
- Device Mode Register
- Power and Clock Gating Control Registers

The base address of the USBFS system control register is different from other registers. This register is independent of the USBFS module to control the related settings of the USBFS module.

Only module global registers, power and clock gating control registers can be accessed in host and device modes. When the USBFS module is in one mode (host or device), the application cannot access registers in another role mode, such as host mode, accessing device mode registers. If an illegal access occurs, a mode mismatch interrupt will be generated and reflected in the module interrupt register USBFS.GINTSTS.NMIS. 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.

For the USBFS module register list and base address, please refer to Table 31-3~Table 31-4USBFS Register List.

USBFS system control register base address: 0x40055400

Table 31-3 USBFS System Control Register List

(USBFS system control register) register name	Offset address	Reset value
USBFS System Control Register (USBFS_SYCTLREG)	0x00	0x0000 0000

USBFS module register base address: 0x400C0000

Table 31-4 USBFS System Control Register List

(USBFS global register) register name	Offset address	Reset value
USBFS VBUS Control Register (USBFS_GVBUSCFG)	0x00	0x0000 0000
USBFS AHB Control Register (USBFS_GAHBCFG)	0x08	0x0000 0000
USBFS USB configuration register (USBFS_GUSBCFG)	0x0c	0x0000 0A00
USBFS Reset Register (USBFS_GRSTCTL)	0x10	0x8000 0000
USBFS Global Interrupt Register (USBFS_GINTSTS)	0x14	0x1400 0020
USBFS Global Interrupt Mask Register (USBFS_GINTMSK)	0x18	0x0000 0000
USBFS Receive Status Debug Read Register (USBFS_GRXSTS)	0x1c	0x0000 0000
USBFS receive status read and pop register (USBFS_GRXSTSP)	0x20	0x0000 0000
USBFS receive FIFO size register (USBFS_GRXFISZ)	0x24	0x0000 0140
USBFS Host Aperiodic Transmit FIFO Size Register (USBFS_HNPTXFISZ)/Endpoint 0 Transmit FIFO Size Register (USBFS_DIEPRXF0)	0x28	0x0200 0140
USBFS Aperiodic Transmit FIFO/Queue Status Register (USBFS_HNPTXSTS)	0x2c	0x0008 0100
USBFS module ID register (USBFS_CID)	0x3c	0x1234 5678
The USBFS host periodically transmits the FIFO size register (USBFS_HPTXFISZ)	0x100	0x0000 0000
USBFS device IN endpoint x transmit FIFO size register (USBFS_DIEPTXFx)	0x100+x*4(x=1~5)	0x0100 0240+(x-1)*0x100

(USBFS host control and status register) register name	Offset address	Reset value
USBFS Host Configuration Register (USBFS_HCFG)	0x400	0x0000 0000
USBFS Host Frame Interval Register (USBFS_HFIR)	0x404	0x0000 EA60
USBFS Host Frame Number/Frame Remaining Interval Register (USBFS_HFNUM)	0x408	0x0000 3FFF
USBFS Host Periodic Transmit FIFO/Queue Status Register (USBFS_HPTXSTS)	0x410	0x0008 0100
USBFS host overall channel interrupt register (USBFS_HAIINT)	0x414	0x0000 0000
USBFS host all channel interrupt mask register (USBFS_HAIINTMSK)	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=0~11)	0x0000 0000
USBFS Host Channel x Interrupt Register (USBFS_HCINTx)	0x508+x*0x20(x=0~11)	0x0000 0000
USBFS Host Channel x Interrupt Mask Register (USBFS_HCINTx)	0x50c+x*0x20(x=0~11)	0x0000 0000
USBFS Host Channel x Transfer Size Register (USBFS_HCTSIZx)	0x510+x*0x20(x=0~11)	0x0000 0000
USBFS Host Channel x DMA Address Register (USBFS_HCDMAx)	0x514+x*0x20(x=0~11)	0xFFFF XXXX

(USBFS Device Control and Status Register) Register Name	Offset address	Reset value
USBFS Device Configuration Register (USBFS_DCFG)	0x800	0x0820 0000
USBFS Device Control Register (USBFS_DCTL)	0x804	0x0000 0000
USBFS Device Status Register (USBFS_DSTS)	0x808	0x0000 0002
USBFS device IN endpoint general interrupt mask register (USBFS_DIEPMSK)	0x810	0x1400 0000
USBFS device OUT endpoint general interrupt mask register (USBFS_DOEPMSK)	0x814	0x0000 0000
USBFS device overall endpoint interrupt register (USBFS_DAINT)	0x818	0x0000 0000
USBFS device overall 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 x Control Register (USBFS_DIEPCTLx)	0x900+x*0x20(n=1~5)	0x0000 0000
USBFS Device IN Endpoint x Interrupt Register (USBFS_DIEPINTx)	0x908+x*0x20(n=0~5)	0x0000 0000
USBFS Device IN Endpoint x Transfer Size Register (USBFS_DIEPSIZx)	0x910+x*0x20(n=0~5)	0x0000 0000
USBFS device IN endpoint x DMA address register (USBFS_DIEPDMAx)	0x914+x*0x20(n=0~5)	0x0000 0000
USBFS Device IN Endpoint x Transmit FIFO Status Register (USBFS_DTXFSTSx)	0x918+x*0x20(n=0~5)	0x0000 0000
USBFS device OUT endpoint 0 control register (USBFS_DOEPCTL0)	0xb00	0x0000 8000
USBFS Device OUT Endpoint x Control Register (USBFS_DOEPCTLx)	0xb00+x*0x20(n=1~5)	0x0000 0000
USBFS Device OUT Endpoint x Interrupt Register (USBFS_DOEPINTx)	0xb08+x*0x20(n=0~5)	0x0000 0000
USBFS device OUT endpoint x transfer size register (USBFS_DOEPSIZx)	0xb10+x*0x20(n=0~5)	0x0000 0000
USBFS device OUT endpoint x DMA address register (USBFS_DOEPDMAx)	0xb14+x*0x20(n=0~5)	0xXXXXXXXX

(USBFS power supply and clock free control register) register name	Offset address	Reset value
USBFS Power and Gating Clock Control Register (USBFS_PCGCCTL)	0xe00	0x0000 0000

31.7.1 USBFS System Control Register

31.7.1.1 USBFS System Control Register (USBFS_SYCTLREG)

Offset address: 0x00

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16					
Reserved																				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0					
Reserved																				
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">SOFEN</td><td style="padding: 2px;">DFB</td></tr> </table>																SOFEN	DFB			
SOFEN	DFB																			
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">Bit</td><td style="padding: 2px;">Marking</td><td style="padding: 2px;">Place name</td><td style="padding: 2px;">Function</td><td style="padding: 2px;">Read and write</td></tr> </table>																Bit	Marking	Place name	Function	Read and write
Bit	Marking	Place name	Function	Read and write																
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">b31~b2</td><td style="padding: 2px;">Reserved</td><td style="padding: 2px;">-</td><td style="padding: 2px;">The reset value must be maintained.</td><td style="padding: 2px;">R</td></tr> </table>																b31~b2	Reserved	-	The reset value must be maintained.	R
b31~b2	Reserved	-	The reset value must be maintained.	R																
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">b1</td><td style="padding: 2px;">SOFEN</td><td style="padding: 2px;">SOF pulse output enable bit</td><td style="padding: 2px;">When the host sends SOF or the device successfully receives SOF, the 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.</td><td style="padding: 2px;">R/W</td></tr> </table>																b1	SOFEN	SOF pulse output enable bit	When the host sends SOF or the device successfully receives SOF, the 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
b1	SOFEN	SOF pulse output enable bit	When the host sends SOF or the device successfully receives SOF, the 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																
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">b0</td><td style="padding: 2px;">DFB</td><td style="padding: 2px;">VBUS/ID pin module internal debounce filter bypass enable bit</td><td style="padding: 2px;">0: The internal debounce filter of the module is valid 1: Bypass module internal debounce filter Note: Accessible in both device mode and host mode.</td><td style="padding: 2px;">R/W</td></tr> </table>																b0	DFB	VBUS/ID pin module internal debounce filter bypass enable bit	0: The internal debounce filter of the module is valid 1: Bypass module internal debounce filter Note: Accessible in both device mode and host mode.	R/W
b0	DFB	VBUS/ID pin module internal debounce filter bypass enable bit	0: The internal debounce filter of the module is valid 1: Bypass module internal debounce filter Note: Accessible in both device mode and host mode.	R/W																

31.7.2 USBFS Global Registers

These registers are available in both host and device modes and do not need to be reprogrammed when switching between the two modes. Bit values in register descriptions are in binary unless otherwise noted.

31.7.2.1 VBUS Configuration Register (USBFS_GVBUSCFG)

Offset address: 0x00

Reset value: 0x0000 0000

This register can be used to set the VBUS value regardless of 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															
Bit	Marking	Place name	Function										Read and write		
b31~b8	Reserved	-	The reset value must be maintained.										R		
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) It is used to reflect the value set by VBUSVAL to the status of USBFS CORE. The value of VBUSVAL is valid only when this bit is set to 1. NOTE: Only accessible in device mode.										R/W		
b5~b0	Reserved	-	The reset value must be maintained.										R		

31.7.2.2 AHB Configuration Register (USBFS_GAHBCFG)

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								PTXF ELVL	TXFE LVL	Reser ved	DMA EN	HBSTLEN[3:0]			GINT MSK
Bit	Marking	Place name	Function												Read and write
b31~b9	Reserved	-	The reset value must be maintained.												R
b8	PTXFELVL	Periodic TxFIFO Empty Threshold	Periodic TxFIFO empty level Indicates when to trigger the periodic TxFIFO empty interrupt bit in the module interrupt register (PTXFE bit in USBFS_GINTSTS). 0: PTXFE (at USBFS_GINTSTS) interrupt indicates periodic TxFIFO is half-empty 1: PTXFE (at USBFS_GINTSTS) interrupt indicates that the periodic TxFIFO is completely empty NOTE: Only accessible in host mode.												R/W
b7	TXFELVL	Device TxFIFO Empty Threshold	TxFIFO empty level In device mode, this bit indicates when the IN endpoint transmit FIFO empty interrupt (TXFE in USBFS_DIEPINTx) is triggered. 0: TXFE (at 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 host mode.												R/W
b6	Reserved	-	The reset value must be maintained.												R
b5	DMAEN	DMA enable	DMA enable 0: Module operates in slave mode 1: The module operates in DMA mode Note: Accessible in both device mode and host mode.												R/W
b4~b1	HBSTLEN	Batch length/type	Burst length/type 0000b: single 0001b: INCR 0011b:INCR4 0101b:INCR8 0111b:INCR16 Other values: reserved Note: Accessible in both device mode and host mode.												R
b0	GINTMSK	Global Interrupt 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: Mask interrupts triggered by the application 1: Unmask application-triggered interrupts Note: Accessible in both device mode and host mode.												R/W

31.7.2.3 USBFS USB Configuration Register (USBFS_GUSBCFG)

Offset address: 0x00C

Reset value: 0x0000 0A00

This register can be used to configure the module after power-up or changing role modes. 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
Reser ved	FDM OD	FHM OD													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Reserved TRDT[3:0] Reserved PHYS EL Reserved TOCAL[2:0]															
Bit	Marking	Place name	Function												Read and write
b31	Reserved	-	The reset value must be maintained.												R
b30	FDMOD	Force device mode	Force device mode Writing a 1 to this bit forces the module into device mode regardless of 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	forced host mode	Force host mode Writing a 1 to this bit forces the module into host mode regardless of 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
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: $TRDT = 4 \times AHB\ clock + 1\ PHY\ clock$ For example: 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: Accessible in both device mode and host mode.												R/W
b9~b7	Reserved	-	The reset value must be maintained.												R
b6	PHYS EL	Full-Speed Family Transceiver Selection	Full Speed serial transceiver select This bit is write-only and is always 1.												W
b5~b3	Reserved	-	The reset value must be maintained.												R
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 has different effects on the state of the data line. The USB standard timeout value for full speed operation is 16 to 18 single digit times, inclusive. The application must program this field according to the enumerated 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

31.7.2.4 USBFS reset register (USBFS_GRSTCTL)

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 LSH	Reser ved	FCRS T	HSRS T	CSRS T					
Bit	Marking	Place name	Function												Read and write					
b31	AHBIDL	AHB Master 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 Idle	DMA request signal This bit indicates that a DMA request is in progress. Used for debugging. Note: Accessible in both device mode and host mode.												R					
b29~b11	Reserved	-	Read as "0", write as "0"												R					
b10~b6	TXFNUM	TxFIFO number	TxFIFO number FIFO number for FIFO flushing using the TxFIFO flush bit. This field should only be changed after the module has cleared 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: — Refresh periodic TxFIFO in master mode — Refresh 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 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 confirming that the module is not currently reading or writing to the TxFIFO. Confirm with the following registers: <ul style="list-style-type: none"> — Read: NAK active interrupt ensures that the module is not currently performing a read operation on the FIFO — Write: The AHBIDL bit in USBFS_GRSTCTL ensures that the module is not currently doing any writes to the FIFO Note: Accessible in both device mode and host mode.												R/W					
b4	RXFFLSH	RxFIFO refresh	RxFIFO flush The application can use this bit to flush the entire RxFIFO, but must first ensure that the module is not currently 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. Usually there is a wait of 8 clock cycles (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					
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					
b1	HSRST	HCLK domain logic soft reset	HCLK soft reset Applications use this bit to refresh the control logic in the AHB clock domain. Only the AHB clock domain pipeline is reset. FIFO is not flushed by this bit. After the transaction on the AHB is terminated according to the protocol, all state machines in the AHB clock domain are reset to the idle state. 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												R/W					

			state machine and used to control the status of the interrupt needs to be cleared. Since the interrupt status bit is not cleared, the application can get the status of all module events that occurred after this bit was set. This bit is self-clearing and the module will clear it when all necessary logic in it is reset. This process takes several clocks, depending on the current state of the module. Note: Accessible in both device mode and host mode.	
b0	CSRST	Module soft reset	<p>Core soft reset Reset the HCLK and PCLK domains as follows: Clear individual interrupt and all CSR register bits except:</p> <ul style="list-style-type: none"> — RSTPDMODL bit in USBFS_PCGCCTL — GAYEHCLK bit in USBFS_PCGCCTL — PWRCLMP bit in USBFS_PCGCCTL — STPPCLK bit in USBFS_PCGCCTL — FSLSPCS bit in USBFS_HCFG — DSPD bit in USBFS_DCFG <p>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 final data phase of the AHB transfer. Immediately terminates all transactions on the USB.</p> <p>The application can write to this bit anytime it needs to reset the module. This bit is self-clearing and the module will clear it after all necessary logic in it is reset, which 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 (synchronous delay). In addition, software must determine that bit 31 in this register is set (AHB Master Idle) before starting to run.</p> <p>Software reset is usually used in two situations, one is during software development, and the other is after the user dynamically changes the PHY selection 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.</p> <p>Note: Accessible in both device mode and host mode.</p>	R/W

31.7.2.5 USBFS Interrupt Status Register (USBFS_GINTSTS)

Offset address: 0x14

Reset value: 0x14000020

This register is used to interrupt the application with system-level events in the current mode (device or host).

Some bits in this register are valid only in host mode, while others are valid only in device mode. Additionally, this register indicates the current mode.

The FIFO status interrupts are read-only; if software reads or writes to the FIFO while these interrupts are being processed, 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	Reserved	PTXF	HCINT	HPRTINT	Reserved	DATAFSUSP	IPXFR / INCOMPIOSOUT	IISOIXFR	OEPINT	IEPIN	Reserved	Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EOPF	ISOODRP	ENUMDNE	USBRST	USBSUSP	ESUSP	Reserved	Reserved	GONAKEFF	GINAKEFF	NPTXFE	RXFNE	SOF	Reserved	MMIS	CMOD

Bit	Marking	Place name	Function	Read and write
b31	WKUINT	Resume/Remote Wake 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. Write 1 to this bit to clear it to 0 by software. Note: Accessible in both device mode and host mode.	R/W
b30	VBUSVINT	VBUS active interrupt	VBUS valid interrupt In device mode, when the USBFS_VBUS pin is detected to change from low to high, the interrupt will be triggered. Write 1 to this bit to clear it to 0 by software. Note: Only accessible in device mode.	R/W
b29	DISCINT	Disconnect interrupt detected	Disconnect detected interrupt This interrupt is triggered when a device disconnection is detected. Write 1 to this bit to clear it to 0 by software. Note: Only accessible in host mode.	R/W
b28	CIDSCHG	Connector ID Line Status Change Interrupt	Connector ID status change The module sets this bit to 1 when the state of the Connector ID line changes. Write 1 to this bit to clear it to 0 by software. Note: Accessible in both device mode and host mode.	R/W
b27	Reserved	-	The reset value must be maintained.	R
b26	PTXFE	Periodic TxFIFO empty interrupt	Periodic TxFIFO empty interrupt This interrupt is triggered when the periodic transmit FIFO is half empty or fully 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). Note: Only accessible in host mode.	R
b25	HCINT	Host channel interrupt	Host channels interrupt When the module sets this bit, it indicates that there is an interrupt pending on one of the channels in the module (in master mode). The application must read the host USBFS_HAINT register to determine the exact number of the channel on which the interrupt occurred, and then	R

			read the corresponding USBFS_HCINTx registers to determine the exact cause of the interrupt. The application must clear the corresponding status bit of the USBFS_HCINTx register before clearing this bit. NOTE: Only accessible in host mode.	
b24	HPRTINT	Host port interrupt	Host port interrupt When the module sets this bit to 1, it indicates that the state of the USBFS controller port in host mode changes. The application must read the USBFS_HPRT register to determine the exact event that raised this interrupt. The application must clear the corresponding status bit of the USBFS_HPRT register before clearing this bit. NOTE: Only accessible in host mode.	R
b23	Reserved	-	The reset value must be maintained.	R
b22	DATAFSUSP	Data fetch pending	<p>Data fetch suspended This interrupt is only valid in DMA mode. This interrupt indicates that the module stopped fetching data for the IN endpoint due to unavailable TxFIFO space or request queue space. 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 based on the IN token sequence learning queue — re-enable the 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 at the same time, clear the global aperiodic IN NAK handshake signal: the module will generate "FIFO is empty when receiving IN token" interrupt. USBFS then sends a NAK response to the host. To avoid this situation, the application can check the DATAFSUSP interrupt in USBFS_GINTSTS, which can ensure that the global NAK handshake signal is cleared after the FIFO is full. Alternatively, the application can mask the "IN token received while FIFO empty interrupt" while clearing the global IN NAK handshake signal. <p>Write 1 to this bit to clear it to 0 by software. NOTE: Only accessible in device mode.</p>	R/W
b21	IPXFR/ INCOMPISOOUT	outstanding periodic transfers/ Incomplete OUT synchronous transfer	<p>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>Write 1 to this bit to clear it to 0 by software. NOTE: Only accessible in host mode.</p> <p>INCOMPISOOUT: Incomplete isochronous OUT transfer In device mode, when the module asserts this interrupt, it indicates that at least one transfer on the isochronous OUT endpoint is not complete in the current frame. This interrupt is triggered with the Periodic End of Frame Interrupt (EOPF) bit in this register.</p> <p>Write 1 to this bit to clear it to 0 by software. 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 the transmission on at least one synchronous IN endpoint in the current frame is not completed. This interrupt is triggered with the Periodic End of Frame Interrupt (EOPF) bit in this register.</p> <p>Write 1 to this bit to clear it to 0 by software. 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 an interrupt pending on an OUT endpoint in the module (in device mode). 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 register to determine the exact cause of the interrupt. The application must clear the corresponding status bit of 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 an interrupt pending on an IN endpoint in the module (in device mode). 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 convert the corresponding</p> <p>The corresponding status bit of the USBFS_DIEPINTx register must be cleared before the bit can be cleared.</p> <p>NOTE: Only accessible in device mode.</p>	R

b17~b16	Reserved	-	The reset value must be maintained.	R
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 (PFIVL bit in USBFS_DCFG) in the USBFS_DCFG register. Write 1 to this bit to clear it to 0 by software. NOTE: Only accessible in device mode.</p>	R/W
b14	ISOODRP	Drop sync OUT packet interrupt	<p>Isochronous OUT packet dropped interrupt The module sets this bit to 1 if the module cannot write a sync OUT packet to the RxFIFO due to insufficient space in the RxFIFO to accommodate the largest packet for the sync OUT endpoint. Write 1 to this bit to clear it to 0 by software. NOTE: Only accessible in device mode.</p>	R/W
b13	ENUMDNE	Enum complete interrupt	<p>Enumeration done interrupt When the module sets this bit to 1, it indicates that the velocity enumeration is complete. The application must read the USBFS_DSTS register to obtain the enumerated speed. Write 1 to this bit to clear it to 0 by software. NOTE: Only accessible in device mode.</p>	R/W
b12	USBRST	USB reset interrupt	<p>USB reset interrupt When the module sets this bit, it indicates that a reset signal was detected on the USB. Write 1 to this bit to clear it to 0 by software. 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 condition has been detected on the USB. When the idle state on the USB bus remains for 3ms, the module will enter the suspend state. Write 1 to this bit to clear it to 0 by software. NOTE: Only accessible in device mode.</p>	R/W
b10	ESUSP	Early hang interrupt	<p>Early suspend interrupt When the module sets this bit to 1, it indicates that the USB has been detected to be idle for 3ms. NOTE: Only accessible in device mode.</p>	R/W
b9~b8	Reserved	-	The reset value must be maintained.	R
b7	GONAKEFF	Global OUT NAK active interrupt	<p>Global OUT NAK effective interrupt Indicates that the "set global OUT NAK" bit (SGONAK bit in USBFS_DCTL) set by the application in the USBFS_DCTL register has taken effect in the module. This bit is cleared by writing to the "Clear Global OUT NAK" bit in the USBFS_DCTL register (CGONAK bit in USBFS_DCTL). NOTE: Only accessible in device mode.</p>	R
b6	GINAKEFF	Global aperiodic IN NAK active interrupt	<p>Global IN nonperiodic NAK effective interrupt Indicates that the "Set Global Aperiodic IN NAK" bit (SGINAK bit in USBFS_DCTL) set by the application in the USBFS_DCTL register 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 is 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 has been sent on the USB. The STALL bit has priority over the NAK bit. 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 or fully empty and there is room in the aperiodic transmit request queue to write at least one entry. 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). NOTE: Only accessible 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. 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, the module, when set to 1, indicates that a SOF (FS) or Keep-Alive (LS) signal has been sent on the USB. The application must set this bit to 1 to clear the interrupt. In device mode, the module, when set to 1, indicates that a SOF token has been received on the USB. The application program can get the current frame number by reading the device status register. This interrupt occurs only when the module is running in FS mode. Write 1 to this bit to clear it to 0 by software. Note: Accessible in both host mode and device mode.</p>	R/W
b2	Reserved	-	The reset value must be maintained.	R

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"> — Module running in device mode accessing 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>Write 1 to this bit to clear it to 0 by software. Note: Accessible in both host mode and device mode.</p>	R/W
b0	CMOD	Current working mode	<p>Current mode of operation Indicates the current mode.</p> <p>0: device mode 1: Host mode</p> <p>Note: Accessible in both host mode and device mode.</p>	R

31.7.2.6 USBFS Interrupt Mask Register (USBFS_GINTMSK)

Offset address: 0x18

Reset value: 0x00000000

This register is used in conjunction with the module interrupt register to interrupt the application. If an interrupt bit is masked, the interrupt associated with that bit will not be generated.

However, the module interrupt (USBFS_GINTSTS) 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
WKUIM	VBUSSVIM	DISCIM	CIDSCHGM	Reser ved	PTXFEM	HCIM	HPRTIM	Reser ved	DATAFSUSPM	IPXFRM/INCO MPISOOUTM	IISOIXFRM	OEPI M	IEPIM	Reser ved	Reser ved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EOPFM	ISOODRPM	ENUMDNE M	USBRSTM	USBSUSPM	ESUSPM	Reser ved	Reser ved	GONAKEFFM	GINAKEFFM	NPTXFEM	RXFNEM	SOFM	Reser ved	MMISM	Reser ved

Bit	Marking	Place name	Function	Read and write
b31	WKUIM	Resume/Remote Wakeup Detected Interrupt Mask	Resume/remote wakeup detected interrupt mask 0: mask interrupt 1: enable interrupt Note: Accessible in both host mode and device mode.	R/W
b30	VBUSSVIM	VBUS active interrupt mask	VBUS valid interrupt mask 0: mask interrupt 1: enable interrupt Note: Only accessible in device mode.	R/W
b29	DISCIM	Disconnect detected interrupt mask	Disconnect detected interrupt interrupt mask 0: mask interrupt 1: enable interrupt Note: Only accessible in host mode.	R/W
b28	CIDSCHGM	Interrupt Connector ID Line Status Change Interrupt Shield	Connector ID status change interrupt mask 0: mask interrupt 1: enable interrupt Note: Accessible in both device mode and host mode.	R/W
b27	Reserved	-	The reset value must be maintained.	R
b26	PTXFEM	Periodic TxFIFO empty interrupt mask	Periodic TxFIFO empty interrupt mask 0: mask interrupt 1: enable interrupt Note: Only accessible in host mode.	R/W
b25	HCIM	Host channel interrupt mask	Host channels interrupt mask 0: mask interrupt 1: enable interrupt Note: Only accessible in host mode.	R/W
b24	HPRTIM	Host port interrupt mask	Host port interrupt mask 0: mask interrupt 1: enable interrupt Note: Only accessible in host mode.	R/W
b23	Reserved	-	The reset value must be maintained.	R
b22	DATAFSUSPM	Data Acquisition Pending Interrupt Mask	Data fetch suspended interrupt mask 0: mask interrupt 1: enable interrupt Note: Only accessible in device mode.	R/W
b21	IPXFRM/INCOMPISOOUTM	Incomplete Periodic Transfer Interrupt Mask/ Incomplete OUT synchronous transfer interrupt mask	Incomplete periodic transfer interrupt mask 0: mask interrupt 1: enable interrupt Note: Only accessible in host mode. INCOMPISOOUT: Incomplete isochronous OUT transfer interrupt mask 0: mask interrupt 1: enable interrupt	R/W

			NOTE: Only accessible in device mode.	
b20	IISOIXFRM	Incomplete IN synchronous 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
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	Discard 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	ENUMDDEM	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 Suspend 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
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: Only accessible in host mode.	R/W
b4	RXFNEM	RxFIFO Not 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
b1	MMISM	Pattern Mismatch Interrupt Interrupt Mask	Mode mismatch interrupt mask 0: mask interrupt 1: enable interrupt Note: Accessible in both host mode and device mode.	R/W
b0	Reserved	-	The reset value must be maintained.	R

31.7.2.7 USBFS Receive Status Debug Read/USBFS Status Read And Pop Registers (USBFS_GRXSTSR/USBFS_GRXSTSP)

Read offset address: 0x01C

The offset address of 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 from the top of the RxFIFO. The receive status content is interpreted differently in host mode and device mode.

When the receive FIFO is empty, the module will ignore the read or stack operation of the register and return 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[11:0]										CHNUM[3:0]				
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b21	Reserved	-	The reset value must be maintained.										R		
b20~b17	PKTSTS	Packet status	Packet status Indicates the status of received packets 0010: IN packet received 0011: IN transfer complete (trigger interrupt) 0101: Data synchronization error (trigger interrupt) 0111: Pause channel (trigger interrupt) Other values: reserved										R		
b16~b15	DPID	Data PID	Data PID Indicates the data PID of the received packet 00: DATA0 10: DATA1 01: DATA2 11: MDATA										R		
b14~b4	BCNT	Byte count	Byte count Indicates the number of bytes of the IN packet 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
b20~b17	PKTSTS	Packet status	Packet status Indicates the status of received packets 0001: Global OUT NAK (trigger interrupt) 0010: OUT packet received 0011: OUT transfer completed (interrupt triggered) 0100: SETUP transaction completed (interrupt triggered) 0110: SETUP packet received Other values: reserved	R
b16~b15	DPID	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 <u>Indicates the number of bytes of the packet received.</u>	R
b3~b0	EPNUM	Endpoint number	Endpoint number Indicates the endpoint number to which the currently received packet belongs.	R

31.7.2.8 USBFS Receive FIFO Size Register (USBFS_GRXFSIZ)

Offset address: 0x024

Reset value: 0x0000 0140

This application can program the size of RAM that must be allocated to the RxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								RXFD[10:0]							
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b11	Reserved	-	The reset value must be maintained.										R		
b10~b0	RXFD	RxFifo depth	RXFD: RxFIFO depth In units of 32-bit words. Minimum value is 16 The maximum value is 256 Power-on reset value is the maximum Rx data FIFO depth.										R/W		

31.7.2.9 USBFS Host Non-Periodic Transmit FIFO Size Register (USBFS_HNPTXFSIZ)/ Device Endpoint0 Transmit FIFO Size Register (USBFS_DIEPTXF0)

Offset address: 0x028

Reset value: 0x02000140

This application can program the size of RAM that must be allocated to the TxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
NPTXFD[15:0]/TX0FD[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
NPTXFSA[15:0]/TX0FSA[15:0]															
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 units of 32-bit words. Minimum value is 16 The maximum value is 256 Device Mode: TX0FD Endpoint 0 TxFIFO depth In units of 32-bit words. Minimum value is 16 The maximum value is 256												R/W
b15~b0	NPTXFSA/ TX0FSA	Aperiodically send RAM start address/endpoint 0 send 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

31.7.2.10 USBFS Host Non-Periodic Transmit FIFO Size Register/Device Endpoint Transmit FIFO Size Register (USBFS_HNPTXSTS)

Offset address: 0x02C

Reset value: 0x00080100

This read-only register contains free space information for the aperiodic TxFIFO and aperiodic transmit request queues.

This register is valid only in host mode, not in device mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reser ved	NPTXQTOP[6:0]								NPTQXS AV[7:0]						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
NPTXFSAV[15:0]															
<hr/>															
Bit	Marking	Place name	Function								Read and write				
b31	Reserved	-	The reset value must be maintained.								R				
b30~b24	NPTXQTOP	Aperiodic send request queue top	Top of the non-periodic transmit request queue Entries currently being processed by the MAC in the aperiodic send request queue. Bit 30:27:Channel/endpoint number Bits 26:25: — 00: IN/OUT token — 01: send packet with zero length — 11: Channel stop command Bit 24:Terminate (last entry for selected channel)								R				
b23~b16	NPTQXS AV	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. 00: Aperiodic send request queue is full 01: 1 location available 10: 2 location available bxn: n positions are available (among them, n range: 0~8) Other values: reserved								R				
b15~b0	NPTXFSAV	Aperiodic send request queue top	Non-periodic Tx FIFO space available Indicates the amount of free space available in the aperiodic Tx FIFO. In units of 32-bit words. 00: Aperiodic Tx FIFO is full 01: 1 word available 10: 2 word available 0xn: n words are available (among them, n range: 0~256) Other values: reserved								R				

31.7.2.11 USBFS Core ID Register (USBFS_CID)

Offset address: 0x03C

Reset value: 0x12345678

This register is a 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]																									
<table border="1" style="width: 100%;"><thead><tr><th>Bit</th><th>Marking</th><th>Place name</th><th>Function</th><th>Read and write</th></tr></thead><tbody><tr><td>b31~b0</td><td>PRODUCT_ID</td><td>Product ID field</td><td>Product ID field Application-programmable ID field.</td><td>R/W</td></tr></tbody></table>																Bit	Marking	Place name	Function	Read and write	b31~b0	PRODUCT_ID	Product ID field	Product ID field Application-programmable ID field.	R/W
Bit	Marking	Place name	Function	Read and write																					
b31~b0	PRODUCT_ID	Product ID field	Product ID field Application-programmable ID field.	R/W																					

31.7.2.12 USBFS Host Periodic Transmit FIFO Size Register (USBFS_HPTXFSIZ)

Offset address: 0x100

Reset value: 0x01400280

This application can program the size of RAM that must be allocated to the cycle TxFIF.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved					PTXFD[10:0]										
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					PTXSA[11:0]										
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b27	Reserved	-	The reset value must be maintained.										R		
b26~b16	PTXFD	Host Periodic TxFIFO Depth	Host periodic TxFIFO depth In units of 32-bit words. Minimum value is 16 The maximum value is 256										R/W		
b15~b12	Reserved	-	The reset value must be maintained.										R		
b11~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		

31.7.2.13 USBFS Device IN Endpoint Transmit FIFO Size Register (USBFS_DIEPTXF_x) (_x = 1..5)

Offset address: 0x104+(x-1)*0x4

Reset value: 0x01000240+(x-1)*0x100

This application can program the size that must be allocated to the device TxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								INEPTXFD[9:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								INEPTXSA[11:0]							
Bit	Marking	Place name	Function								Read and write				
b31~b26	Reserved	-	The reset value must be maintained.								R				
b25~b16	INEPTXFD	IN endpoint TxFIFO depth	Device IN endpoint TxFIFO depth In units of 32-bit words. Minimum value is 16 The maximum value is 256								R/W				
b15~b12	Reserved	-	The reset value must be maintained.								R				
b11~b0	INEPTXSA	IN endpoint TxFIFOx RAM starting address	IN endpoint FIFOx transmit RAM start address This field contains the memory start address of the IN endpoint transmit FIFOx. This address must be aligned to a 32-bit memory location.								R/W				

31.7.3 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, as the results produced are ambiguous.

Bit values in register descriptions are in binary unless otherwise noted.

31.7.3.1 USBFS Host Configuration Register (USBFS_HCFG)

Offset address: 0x400

Reset value: 0x00000000

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		
b2	FSLSS	Only FS and LS are supported	Applications use this bit to control the enumeration speed of modules. Using this bit, the application can make the module work as FS master even if the connected device supports HS communication. Do not change this field after initial programming. 1: FS/LS only, even if the connected device can support HS										R/W		
b1~b0	FSLSPCS	FS/LS PHY clock selection	FS/LS PHY clock select When the module is in FS host mode 01: PHY clock runs at 48 MHz Other values: reserved When the module is in LS master 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		

31.7.3.2 USBFS Host Frame Interval Register (USBFS_HFIR)

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	-	Must hold the reset value	R											
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. The application can write a value to this register only after setting the port enable bit of the host port control and status register (PENA bit of USBFS_HPRT). If no value is programmed, the module will calculate 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 bits can be modified whenever the application needs to change the frame interval time.	R/W											

31.7.3.3 USBFS Host Frame Interval Register (USBFS_HFNUM)

Offset address: 0x408

Reset value: 0x0000 3FFF

This register is used to indicate the current frame number. It also indicates the remaining time of the current frame in number of PHY clocks.

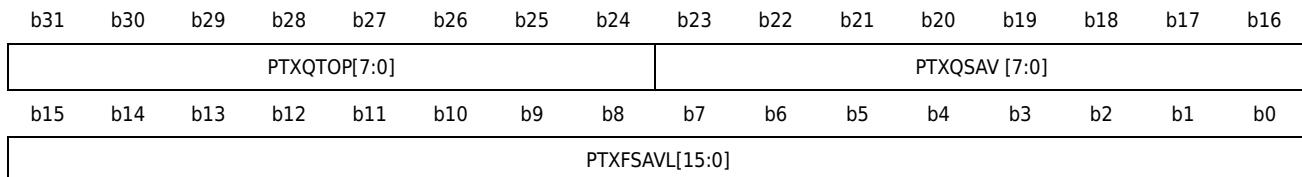
b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
FTREM[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FRNUM[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	FTREM	frame time remaining	Frame time remaining Indicates the remaining time (in PHY clocks) of the current frame. This field is decremented by 1 every 1 PHY clock elapsed. When the value reaches zero, this field is reloaded with the value in the Interframe Register and a new SOF is sent by the module on the USB.	R											
b15~b0	FRNUM	frame number	Frame number The value of this field will be incremented by 1 when a new SOF is sent on the USB, and will be cleared to 0 when 0x3FFF is reached.	R											

31.7.3.4 USBFS Host Periodic Transmit FIFO/Queue Status Register (USBFS_HPTXSTS)

Offset address: 0x410

Reset value: 0x00080100

This read-only register contains free space information for the periodic TxFIFO and periodic transmit request queues.



Bit	Marking	Place name	Function	Read and write
b31~b24	PTXQTOP	Periodically send the request to the top of the queue	Top of the periodic transmit request queue Indicates the item in the periodic Tx request queue that the MAC is currently processing. This register is used for debugging. Bit 31: Odd/Even frame — 0: send in even frames — 1: send in odd-numbered frames Bit 30:27:Channel number Bit 26:25: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	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 requests and OUT requests. 00: Periodic send request queue is full 01: 1 position available 10: 2 position available Bxn: n positions are available (among them, n range: 0~8) Other values: reserved	R
b15~b0	PTXFSAVL	Periodically send data FIFO free space	Periodic transmit data FIFO space available Indicates the number of free slots of the periodic TxFIFO available for writing. in units of 32-bit words 0000: Periodic TxFIFO is full 0001: 1 word available 0010: 2 word available bxn: n words are available (among them, n range: 0~PTXFD) Other values: reserved	R

31.7.3.5 USBFS Host All Channels Interrupt Register (USBFS_HAINT)

Offset address: 0x414

Reset value: 0x0000 0000

When an event occurs on the channel, the host overall channel interrupt register will use the host channel interrupt bit (HCINT bit in USBFS_GINTSTS) in the module interrupt register to interrupt the application. Each channel corresponds to 1 interrupt bit, up to 12 bits.

Bits in this register are also cleared when the application clears an interrupt via the corresponding host channel x interrupt register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	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				HAINT[11:0]											
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b12	Reserved	-	The reset value must be maintained.										R		
b11~b0	HAINT	Channel interruption	Channel interrupt One bit per channel: bit 0 for channel 0 and bit 11 for channel 11.										R/W		

31.7.3.6 USBFS Host All Channels Interrupt Mask Register (USBFS_HAINTMASK)

Offset address: 0x418

Reset value: 0x0000 0000

The Host All Channels Interrupt Mask Register is used in conjunction with the Host All Channels Interrupt Register to interrupt the application when an event occurs on the channel.

Each channel corresponds to 1 interrupt mask bit, up to 12 bits.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				HAINTM[11:0]											
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b12	Reserved	-	Read as "0", write as "0"										R		
b11~b0	HAINTM	Channel interrupt mask	Channel interrupt mask 0: mask interrupt 1: enable interrupt One bit per channel: bit 0 for channel 0 and bit 11 for channel 11.										R/W		

31.7.3.7 USBFS Host Port Control And Status Register (USBFS_HPRT)

Offset address: 0x440

Reset value: 0x0000 0000

This register is only available in host mode. Currently, the USBFS host only supports one port.

This register contains information related to the USB port, such as USB reset, enable, suspend, resume, connection status. The PENCHNG/PCDET bit in this register can trigger an application interrupt through 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]		Reser ved	PRST	PSUS P	PRES	Reserved		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				
b18~b17	PSPD	Port speed	Port speed Indicates the speed of the device connected to the port. 00/11: Reserved 01: full speed 10: low speed										R				
b16~b13	Reserved	-	The reset value must be maintained.										R				
b12	PPWR	Port power	Port power Applications use this field to control power to this port. Since the built-in PHY of this USBFS does not have power supply capability, when this bit is set to 1, the external USB power supply chip can be powered by USBFS_DRVVBUS. 0: power down 1: power on										R/W				
b11~b10	PLSTS	Port line status	Indicates the current logic level of the USB data line Bit 11: Logic level of USBFS_DM Bit 10: Logic level of USBFS_DP										R				
b9	Reserved	-	The reset value must be maintained.										R				
b8	PRST	Port reset	Port reset When an application sets this bit, it initiates a reset sequence on that port. The application must time the reset cycle and clear this bit after the reset sequence is complete. 0: Port is 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				
b7	PSUSP	Port hang	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. A read value of this bit reflects the current pending state of the port. After the 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 bit in the module interrupt register. The open connection detection interrupt bit (respectively WKUINT or DISCINT in USBFS_GINTSTS) is set to 1, and the module can also clear this bit. 0: Port is not in suspend mode 1: Port is in suspend mode										R/W				

			Port resume The application sets this bit to drive a resume signal on this port. The module will continue to drive the resume signal until the application program clears this bit.	
b6	PRES	Port recovery	As indicated by the port resume/remote wakeup detect interrupt bit in the module interrupt register (WKUINT bit in USBFS_GINTSTS), if the module detects a USB remote wakeup sequence, it starts driving the resume signal without intervention from the application; if the module detects a disconnect If connected, clear this bit to 0. The read value of this bit indicates whether the current module The recovery signal is being driven. 0: Do not drive recovery signal 1: drive recovery signal	R/W
b5~b4	Reserved	-	The reset value must be maintained.	R
b3	PENCHNG	Port enable/disable change	Port enable/disable change The module sets this bit to 1 when the port enable bit 2 in this register changes state. Write 1 to this bit to clear it to 0 by software.	R/W
b2	PENA	Port enable	Port enable After the port performs a reset sequence, it can only be enabled by the module and can be disabled by an overcurrent condition, a disconnect condition, or by clearing this bit by the application. The application cannot set this bit by writing to the register. The port can only be disabled by clearing this bit. Manipulation of this bit will not trigger any interrupt to the application. 0: port disabled 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 1 to trigger an interrupt to the application using the host port interrupt bit in the module interrupt register (HPRTINT bit in USBFS_GINTSTS). The application must set this bit to 1 to clear the 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 the device	R

31.7.3.8 USBFS Host Channel-X Characteristics Register (USBFS_HCCCHARx) (x = 0..11)

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]										
<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 USBFS host hardware. 0: Channel disabled 1: enable channel										R/W		
b30	CHDIS	Channel prohibited	Channel disable The application sets this bit to 1 to stop sending/receiving data through the channel, even if the transmission through the channel is not completed, the stop operation still takes effect. The application must wait for the disabled channel interrupt to confirm that the channel has been disabled.										R/W		
b29	ODDFRM	Odd-numbered 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 (synchronous and interrupt) transactions. 0: even frame 1: odd-numbered 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	Reserved	-	The reset value must be maintained.										R		
b19~b18	EPTYP	Endpoint type	Endpoint type Indicates the selected transport type. 00: control 01: synchronous 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		
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 for device endpoints communicating with this host channel.										R/W		

31.7.3.9 USBFS Host Channel-X Interrupt Register (USBFS_HCINTx) ($x = 0..11$)

Offset address: 0x508 + (channel number \times 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 (HCINT bit in USBFS_GINTSTS) is set in the module interrupt register. Before performing a read operation on the register, the application must first read the Host All Channel 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	Reser ved	CHH	XFRC
<hr/>																
Bit	Marking	Place name	Function	Read and write												
b31~b11	Reserved	-	The reset value must be maintained.	R												
b10	DTERR	Data switching error	Data toggle error The application needs to write 1 to clear this bit.	R/W												
b9	FRMOR	Frame overflow error	Frame overrun The application needs to write 1 to clear this bit.	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 size. The application needs to write 1 to clear this bit.	R/W												
b7	TXERR	Communication transaction error	Transaction error Indicates that one of the following errors occurred on the USB: CRC check failed Time-out Bit stuffing error Wrong EOP The application needs to write 1 to clear this bit.	R/W												
b6	Reserved	-	The reset value must be maintained.	R												
b5	ACK	ACK response received/sent	ACK response received/transmitted interrupt The application needs to write 1 to clear this bit.	R/W												
b4	NAK	NAK response received	NAK response received interrupt The application needs to write 1 to clear this bit.	R/W												
b3	STALL	STALL response received	STALL response received interrupt	R/W												
b2	Reserved	-	Read as "0", write as "0"	R												
b1	CHH	Channel stop	Channel halted The transfer ended abnormally due to any USB transaction error or in response to an application inhibit request. The application needs to write 1 to clear this bit.	R/W												
b0	XFRC	Transmission completion	Transfer completed The transfer completed normally without any errors. The application needs to write 1 to clear this bit.	R/W												

31.7.3.10 USBFS Host Channel-X Interrupt Mask Register (USBFS_HCINTMSKx) (x = 0..11)

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
b10	DTERRM	Data switching error interrupt mask	Data toggle error mask 0: mask interrupt 1: enable interrupt													R/W
b9	FRMORM	Frame overflow error interrupt mask	Frame overrun mask 0: mask interrupt 1: enable interrupt													R/W
b8	BBERRM	Crosstalk Error Interrupt Masking	Babble error mask 0: mask interrupt 1: enable interrupt													R/W
b7	TXERRM	Communication transaction error interrupt mask	Transaction error mask 0: mask interrupt 1: enable interrupt													R/W
b6	Reserved	-	The reset value must be maintained.													R
b5	ACKM	Received/sent ACK response interrupt mask	ACK response received/transmitted interrupt mask 0: mask interrupt 1: enable interrupt													R/W
b4	NAKM	NAK response interrupt mask received	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	Reserved	-	The reset value must be maintained.													R
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

31.7.3.11 USBFS Host Channel-X Transfer Size Register (USBFS_HCTSIZx) (x = 0..11)

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
b30~b29	DPID	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 throughout the 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 value each time a packet is successfully sent or received. After this value reaches 0, the application will be 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 data bytes sent by the host during the transfer. For IN operations, this field is the buffer size reserved by the application for transfers. For IN transactions (periodic and aperiodic), the application will program this field to an integer multiple of the maximum packet size.	R/W

31.7.3.12 USBFS Host Channel-X DMA Address Register (USBFS_HCDMAx) (x = 0..11)

Offset address: 0x514 + (channel number × 0x20)

Reset value: 0xXXXX 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]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b0	DMAADDR	DMA address	DMA address This field stores the address of the memory used by the host for DMA transfers to obtain data from or send data to the device endpoint. This register is incremented at the end of each AHB transfer.										R/W		

31.7.4 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 results are ambiguous.

Bit values in register descriptions are in binary unless otherwise noted.

31.7.4.1 USBFS Device Configuration Register (USBFS_DCFG)

Offset address: 0x800

Reset value: 0x0820 0000

This register configures the module into device mode after power-up, certain control commands, or enumeration. Do not change this register after initial programming.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved	PFIVL[1:0]			DAD[6:0]								Reserved	NZLSOHSK	DSPD[1:0]	
<hr/>															
Bit	Marking	Place name	Function												Read and write
b31~b13	Reserved	-	The reset value must be maintained.												R
b12~b11	PFIVL	periodic frame interval	Periodic frame interval Indicates the point within a frame at which the application must be notified using a periodic frame interrupt. This function can be used to determine if all isochronous communications for that 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
b2	NZLSOHSK	Non-zero length status OUT handshake signal	Non-zero-length status OUT handshake During the OUT transaction of the control transfer status phase, the application can use this field to select the handshake signal to be sent after the module receives a non-zero length packet. 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 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 the chirp sequence is completed, 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

31.7.4.2 USBFS Device Control Register (USBFS_DCTL)

Offset address: 0x804

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16			
Reserved																		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
Reserved				POPR GDN E	CGO NAK	SGO NAK	CGIN AK	SGIN AK	Reserved			GON STS	GINS TS	SDIS	RWU SIG			
<hr/>																		
Bit	Marking	Place name	Function												Read and write			
b31~b12	Reserved	-	The reset value must be maintained.												R			
b11	POPRGDNE	Power-on programming complete	Power-on programming done The application uses this bit to indicate that the register has finished programming after waking up from power-down mode.												R/W			
b10	CGONAK	Clear global OUT NAK	Clear global OUT NAK Writing to this bit will clear the global OUT NAK.												W			
b9	SGONAK	Set global OUT NAK	Set global OUT NAK Writing to this bit sets the global OUT NAK. Applications use this bit to send NAK handshake signals on all OUT endpoints. The application program can only set this bit to 1 when it is sure that the global OUT NAK valid bit (GONAKEFF bit in USBFS_GINTSTS) in the module interrupt register has been cleared.												W			
b8	CGINAK	Clear global IN NAK	Clear global IN NAK Writing to this bit will clear the global IN NAK.												W			
b7	SGINAK	Set global IN NAK	Set global IN NAK Writing to this field sets the global aperiodic IN NAK. Applications use this bit to cause all aperiodic IN endpoints to send NAK handshake signals. The application program can only set this bit to 1 when it is sure that the global IN NAK valid bit (GINAKEFF bit in USBFS_GINTSTS) in the module interrupt register has been cleared.												W			
b6~b4	Reserved	-	The reset value must be maintained.												R			
b3	GONSTS	Global OUT NAK status	Global OUT NAK status 0: Handshake will be sent according to FIFO status and NAK and STALL bit settings. 1: No data is received regardless of whether there is free space in the RxFIFO. Reply NAK handshake to all received packets except SETUP transactions. All OUT packets of type isochronous will be dropped.												R			
b2	GINSTS	Global IN NAK status	Global IN NAK status 0: Handshake will be replied based on data availability in transmit FIFO. 1: Make all aperiodic IN endpoints reply with NAK handshake signals, regardless of the availability of data in the send 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 attached, and the device will not receive signals on the USB. The module remains disconnected until the application program clears this bit. 0: Normal operation. Clearing this bit after a soft disconnect causes the host to receive a device connected event. After reconnecting the device, the USB host restarts device enumeration. 1: Make the host receive a device disconnect event.												R/W			
b0	RWUSIG	Send a remote wakeup signal	At full speed, the minimum time for soft disconnection is specified as follows: Suspended state: the minimum time is 1ms+2.5us Idle state: 2.5us Non-idle or suspend state: 2.5us												R/W			
<hr/>																		

suspended state. According to the USB 2.0 specification, the application must clear this bit within 1 ms to 15 ms of setting it.

31.7.4.3 USBFS Device Status Register (USBFS_DSTS)

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 interrupted endpoint information must be read from the device overall interrupt (USBFS_DAINT) register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16		
Reserved										FNSOF[13:8]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
FNSOF[7:0]										Reserved		EERR	ENUMSPD[1:0]	SUSP STS			
<hr/>																	
Bit	Marking	Place name	Function										Read and write				
b31~b22	Reserved	-	The reset value must be maintained.										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				
b3	EERR	Indeterminate error	Erratic error The module sets this bit to report any pending errors. Due to an indeterminate error, the USBFS controller enters the suspend state and generates an interrupt to the early suspend bit of the USBFS_GINTSTS register (ESUSP bit in USBFS_GINTSTS). If the early hang 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 that the USBFS controller is enumerated after detecting the speed through the chirp sequence. 01: reserved 10: reserved 11: Full speed (PHY clock running at 48 MHz) Other values: reserved										R				
b0	SUSPSTS	Suspended state	Suspend status In device mode, this bit is set whenever a suspend condition is detected on the USB. When the idle state on the USB bus is kept for 3ms, the module will enter the suspend state. A module exits the suspended state when: — Activity on the USB cable — The application writes to the remote wakeup signal bit of the USBFS_DCTL register (RWUSIG bit in USBFS_DCTL).										R				

31.7.4.4 USBFS Device IN Endpoint Common Interrupt Mask Register (USBFS_DIEPMSK)

Offset address: 0x810

Reset value: 0x0000 0000

This register works in conjunction with the individual USBFS_DIEPINTx registers for all endpoints to generate interrupts on each IN endpoint. IN endpoint interrupts in the USBFS_DIEPINTx registers can be masked by writing to the corresponding bit in this register. By default, status interrupts are masked.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved																
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved										INEP NEM	INEP NMM	ITTXF EMSK	TOM	Reser ved	EPDM	XFR M
Bit	Marking	Place name	Function										Read and write			
b31~b7	Reserved	-	The reset value must be maintained.										R			
b6	INEPNEM	IN endpoint NAK effective interrupt mask	IN endpoint NAK effective mask 0: mask interrupt 1: enable interrupt										R/W			
b5	INEPNMM	IN token interrupt mask received on EP mismatch	IN token received with EP mismatch mask 0: mask interrupt 1: enable interrupt										R/W			
b4	ITTXFEMSK	IN token interrupt mask received while 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			
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			

31.7.4.5 USBFS Device OUT Endpoint Common Interrupt Mask Register (USBFS_DOEPMSK)

Offset address: 0x814

Reset value: 0x0000 0000

This register works in conjunction with the individual USBFS_DOEPINTx registers for all endpoints to generate interrupts on each OUT endpoint. OUT endpoint interrupts in the USBFS_DOEPINTx registers can be masked by writing to the corresponding bit in this register. By default, status interrupts are masked.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Bit	Marking	Place name	Function										Read and write		
b31~b5	Reserved	-	The reset value must be maintained.										R		
b4	OTEPDM	OUT token interrupt mask received when endpoint disabled	OUT token received when endpoint disabled mask Applies to control OUT endpoints only. 0: mask interrupt 1: enable interrupt										R/W		
b3	STUPM	The SETUP phase completes the interrupt mask	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		
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		

31.7.4.6 USBFS Device OUT Endpoint Common Interrupt Mask Register (USBFS_DAINT)

Offset address: 0x818

Reset value: 0x0000 0000

When a valid event occurs on the endpoint, the USBFS_DAINT register will interrupt the application via 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 up to 16 interrupt bits. Bidirectional endpoints will use the corresponding IN and OUT interrupt bits. When the application sets and clears a bit in the corresponding Device Endpoint x Interrupt Register (USBFS_DIEPINTx/USBFS_DOEPINTx), the corresponding bit in this register is also set and cleared.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										OEPINT[5:0]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										IEPINT[5:0]					

Bit	Marking	Place name	Function	Read and write
b31~b22	Reserved	-	The reset value must be maintained.	R
b21~b16	OEPINT	OUT endpoint interrupt bit	OUT endpoint interrupt bits One bit per OUT endpoint: OUT endpoint 0 corresponds to bit 16, and OUT endpoint 5 corresponds to bit 21.	R/W
b15~b6	Reserved	-	The reset value must be maintained.	R
b5~b0	IEPINT	IN endpoint interrupt bit	IN endpoint interrupt bits One bit per IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 5 corresponds to bit 5.	R/W

31.7.4.7 USBFS Device All Endpoints Interrupt Mask Register (USBFS_DAIINTMSK)

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 the interrupt will still be set.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										OEPINTM[5:0]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										IEPINTM[5:0]					
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b22	Reserved	-	The reset value must be maintained.										R		
b21~b16	OEPINTM	OUT endpoint interrupt mask bit	OUT endpoint interrupt mask bits One bit per OUT endpoint: OUT endpoint 0 corresponds to bit 16, and OUT endpoint 5 corresponds to bit 21. 0: mask interrupt 1: enable interrupt										R/W		
b15~b6	Reserved	-	The reset value must be maintained.										R		
b5~b0	IEPINTM	IN endpoint interrupt mask bit	IN endpoint interrupt mask bits One bit per IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 5 corresponds to bit 5. 0: mask interrupt 1: enable interrupt										R/W		

31.7.4.8 USBFS Device IN Endpoint FIFO Empty Interrupt Mask Register (USBFS_DIEPEMPMSK)

Offset address: 0x834

Reset value: 0x0000 0000

This register is used to control the generation of 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
Reserved															
INEPTXFEM[5:0]															
Bit	Marking	Place name	Function												Read and write
b31~b6	Reserved	-	The reset value must be maintained.												R
b5~b0	INEPTXFEM	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 of an IN endpoint: IN endpoint 0 corresponds to bit 0, while IN endpoint 5 corresponds to bit 5 0: mask interrupt 1: enable interrupt												R/W

31.7.4.9 USBFS Device Control IN Endpoint 0 Control Register (USBFS_DIEPCTL0)

Offset address: 0x900

Reset value: 0x0000 8000

This register is used to control Control Transfer Endpoint 0.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPEN A	EPDIS S	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															MPSIZ[1:0]

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to start a data send on endpoint 0. The module clears this bit before any of the following interrupts are triggered on this endpoint: — Endpoint prohibition — Transmission completion	R/W
b30	EPDIS	Endpoint prohibition	Endpoint disable An application can set this bit to stop sending data on an endpoint even before the transfer on that endpoint is complete. The application must wait for an endpoint disable interrupt to occur before the endpoint is considered disabled. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only if the Endpoint Enable bit for that endpoint is set.	R/W
b29~b28	Reserved	-	The reset value must be maintained.	R
b27	SNAK	Set the NAK bit	Set NAK Writing to this bit will set the NAK bit of the endpoint. With this bit, the application can control the sending of NAK handshake signals on the endpoint. The module can also set this bit of the endpoint to 1 after the endpoint receives the SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK Writing to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number This value is set to the FIFO number assigned to IN endpoint 0. Only TX-FIFO0 can be used.	R/W
b21	STALL	STALL handshake	STALL handshake The application can only set this bit to 1, the module will clear this bit to 0 when the endpoint receives a SETUP token. If the NAK bit, global IN NAK, or global OUT NAK and this bit are both set, the STALL bit takes precedence.	R/W
b20	Reserved	-	The reset value must be maintained.	R
b19~b18	EPTYP	Endpoint type	Endpoint type The hardware is set to '00', indicating a control type endpoint.	R
b17	NAKSTS	NAK state	NAK status Indicates the following results: 0: Module replies to non-NAK handshake according to 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 stops sending data even if there is still data available in the TxFIFO. Regardless of how this bit is set, the module always responds to SETUP packets with an ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R
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
b1~b0	MPSIZ	Maximum Packet Size	Maximum packet size Applications must program this field to the maximum packet size for the current logical endpoint. 00: 64 bytes 01: 32 bytes 10: 16 bytes	R/W

11: 8 bytes

31.7.4.10 USBFS Device IN Endpoint X Control Register (USBFS_DIEPCTLx)(x=1..5)

Offset address: 0x900 + (endpoint number × 0x20)

Reset value: 0x0000 0080

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	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 The application sets this bit to initiate data sending on the endpoint. The module clears this bit before any of the following interrupts are triggered on this endpoint: — SETUP phase completed — Endpoint prohibition — Transmission completion	R/W
b30	EPDIS	Endpoint prohibition	Endpoint disable An application can set this bit to stop sending data on an endpoint even before the transfer on that endpoint is complete. The application must wait for an endpoint disable interrupt to occur before the endpoint is considered disabled. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only if the Endpoint Enable bit for that endpoint is set.	R/W
b29	SODDFRM	set odd-numbered frames	Set odd frame Applies only to synchronous IN and OUT endpoints. Writing to this field sets the Even/odd-numbered frame (EONUM) field to odd-numbered frames.	R/W
b28	SD0PID/ SEVNFRM	set DATA0 PID/ SEVNFRM	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. Writing to this field sets the Even/odd-numbered frames (EONUM) field to even frames.	R/W
b27	SNAK	Set the NAK bit	Set NAK Writing to this bit will set the NAK bit of the endpoint. With this bit, the application can control the sending of NAK handshake signals on the endpoint. The module can also set this bit of the OUT endpoint to 1 when a transfer completion interrupt occurs or after SETUP is received on the endpoint.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK Writing to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number These bits are used to specify the FIFO number associated with this endpoint. A separate FIFO number must be set for each valid IN endpoint. This field is only valid for IN endpoints.	R/W
b21	STALL	STALL handshake	STALL handshake Setting this bit by the application causes the device to reply STALL to 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
b19~b18	EPTYP	Endpoint type	Endpoint type The following are the transport types supported by this logical endpoint. 00: control 01: synchronous	R

			10: Batch 11: interrupt	
b17	NAKSTS	NAK state	<p>NAK status</p> <p>Indicates the following results:</p> <p>0: Module replies to non-NAK handshake according to FIFO status.</p> <p>1: The module replies with a NAK handshake on this endpoint.</p> <p>When an application or module sets this bit:</p> <p>For non-synchronous IN endpoints: The module stops sending any data through the IN endpoint even if there is data available in the TxFIFO.</p> <p>For synchronous IN endpoints: The module sends zero-length packets even if there is data available in the TxFIFO.</p> <p>Regardless of how this bit is set, the module always responds to SETUP packets with an ACK handshake.</p>	R
b16	EONUM/ DPID	Even/odd- numbered frames/ Endpoint data PID	<p>Even/odd frame</p> <p>Applies only to synchronous IN endpoints.</p> <p>Indicates the number of the frame in which the module sends/receives synchronized data for this endpoint. The application must program the even/odd-numbered frame number through the SEVNFRM and SODDFRM fields in this register for this endpoint to send/receive isochronous data.</p> <p>0: even frame 1: odd-numbered frames</p> <p>DPID: Endpoint data PID</p> <p>Applies to interrupt/bulk IN endpoints only.</p> <p>Contains the PID of packets 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 programs the DATA0 or DATA1 PID using the SD0PID register field.</p> <p>0: DATA0 1: DATA1</p>	R/W
b15	USBAEP	USB active endpoint	<p>USB active endpoint</p> <p>Indicates whether this endpoint is active in the current configuration and interface. The module clears this bit to 0 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 register accordingly and set this bit to 1.</p>	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	MPSIZ	Maximum Packet Size	<p>Maximum packet size</p> <p>Applications must program this field to the maximum packet size for the current logical endpoint.</p> <p>This value is in bytes.</p>	

31.7.4.11 USBFS Device IN Endpoint X Interrupt Register (USBFS_DIEPINTx) (x=0..5)

Offset address: 0x908 + (endpoint number × 0x20)

Reset value: 0x0000 0000

This register indicates the status of the endpoint on USB and AHB related events. The application must read this register when the IN endpoint interrupt bit (IEPINT bit in USBFS_GINTSTS) is set in the module interrupt register. Before an application can read this register, it must first read the Device Entire Endpoint Interrupt (USBFS_DAINT) register 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
b7	TXFE	Transmit FIFO is empty	Transmit FIFO empty This interrupt is asserted when the TxFIFO for this endpoint is half or fully empty. Whether the TxFIFO is half empty or fully empty is determined by the TxFIFO blank 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 NAKs the IN endpoint by writing data to the CNAK bit in USBFS_DIEPCTLx. This interrupt indicates that the module has sampled a NAK that was 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 program has acted in the module. This interrupt does not guarantee that a NAK handshake was sent on the USB. The STALL bit has priority over the NAK bit. Software can also write 1 to clear this bit.												R/W
b5	Reserved	-	The reset value must be maintained.												R
b4	TTXFE	IN token received when TxFIFO is empty	IN token received when TxFIFO is empty Applies to acyclic IN endpoints only. When the TxFIFO (periodic/aperiodic) corresponding to this endpoint is empty, an IN token is received and an interrupt is generated. Write 1 to clear by software.												R/W
b3	TOC	Time-out	Timeout condition Applies to control IN endpoints only. Indicates that this endpoint timed out on the most recently received IN token response. Write 1 to clear by software.												R/W
b2	Reserved	-	The reset value must be maintained.												R
b1	EPDSD	Endpoint disable interrupt	Endpoint disabled interrupt This bit indicates that the endpoint has been disabled by the application. Write 1 to clear by software.												R/W
b0	XFRC	Transfer complete interrupt	Transfer completed interrupt This field indicates that transfers set on this endpoint have completed transfers on USB and AHB. Write 1 to clear by software.												R/W

31.7.4.12 USBFS Device IN Endpoint 0 Transfer Size Register (USBFS_DIEPTSIZE0)

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 has cleared the endpoint enable bit.

Non-zero endpoints use the registers of endpoints 1~5.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										PKTCNT[1:0]		Reserved			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										XFRSIZ[6:0]					
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b21	Reserved	-	The reset value must be maintained.										R		
b20~b19	PKTCNT	Packet count	Packet count Indicates the number of data packets included in one data transmission of endpoint 0. This field will be decremented each time a packet (max or short) is read from the TxFIFO.										R/W		
b18~b7	Reserved	-	The reset value must be maintained.										R		
b6~b0	XFRSIZ	transfer size	Transfer size Indicates the amount of data contained in one data transmission of endpoint 0, in bytes. The module interrupts the application only after the application has transferred 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		

31.7.4.13 USBFS Device IN Endpoint X Transfer Size Register (USBFS_DIEPTSIz)(x=1..5)

Offset address: 0x910 + (endpoint number × 0x20)

Reset value: 0x0000 0000

The application must modify this register before enabling the endpoint. After enabling the endpoint through the endpoint enable bit (EPENA bit in USBFS_DIEPCTLx) in the USBFS_DIEPCTLx register, the module modifies this register. The application can read this register only after the module has cleared the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved		PKTCNT[9:0]										XFRSIZ[18:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
XFRSIZ[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	The reset value must be maintained.	R
b28~b19	PKTCNT	Packet count	Packet count Indicates the number of data packets contained in a data transmission on this endpoint. This field will be decremented each time a packet (max size or short) is read from the TxFIFO.	R/W
b18~b0	XFRSIZ	transfer size	Transfer size This field contains the amount of data contained in a data transmission of the current endpoint, in bytes. The module interrupts the application only after the application has transferred 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

31.7.4.14 USBFS Device IN Endpoint X Transfer Size Register (USBFS_DIEPDMAx)(x=0..5)

Offset address: 0x914 + (endpoint number × 0x20)

Reset value: 0x0000 0000

This register is used to set the DMA address in device endpoint 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 start 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 for every AHB transfer. The application must set a double word aligned address.	R/W											

31.7.4.15 USBFS Device IN Endpoint Transmit FIFO Status Register (USBFS_DTXFSTSx)(x=0..5)

Offset address: 0x918 + (endpoint number × 0x20)

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
INEPTFSAV[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	The reset value must be maintained.	R											
b15~b0	INEPTFSAV	IN endpoint TxFIFO free space	IN endpoint TxFIFO space available Indicates the amount of free space available in the endpoint TxFIFO. In 32-bit words: 0x0: Endpoint TxFIFO is full 0x1: 1 word available 0x2: 2 words available 0xn: n words are available	R											

31.7.4.16 USBFS Device Control OUT Endpoint 0 Control Register (USBFS_DOEPCTL0)

Offset address: 0xB00

Reset value: 0x0000 8000

This register is used to control Control Transfer Endpoint 0.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPENA	EPDIS	Reserved	SNAK	CNAK		Reserved		STALL	SNPM	EPTYP[1:0]	NAKSTS	Reserved			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USBAEP						Reserved								MPSIZ[1:0]	

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to 1 to initiate data reception on Endpoint 0. The module clears this bit before any of the following interrupts are triggered on this endpoint: — SETUP phase completed — Endpoint prohibition — Transmission completion	R/W
b30	EPDIS	Endpoint prohibition	Endpoint disable The application cannot disable control of OUT endpoint 0.	R
b29~b28	Reserved	-	The reset value must be maintained.	R
b27	SNAK	Set the NAK bit	Set NAK Writing to this bit will set the NAK bit of the endpoint. With this bit, the application can control the sending of NAK handshake signals on the endpoint. The module can also set this bit of the endpoint to 1 after the endpoint receives the SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK Writing to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	Reserved	-	The reset value must be maintained.	R
b21	STALL	STALL handshake	STALL handshake When this endpoint receives a SETUP token, the application can only set this bit, and the module will clear it. If the NAK bit, the 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 SETUP packets with an ACK handshake.	R/W
b20	SNPM	Monitor mode	Snoop mode This bit is used to configure the endpoint into listen mode. In listen mode, the module does not check the OUT packet for correctness before transferring it to the application memory.	R/W
b19~b18	EPTYP	Endpoint type	Endpoint type The hardware is set to '00', indicating a control type endpoint.	R/W
b17	NAKSTS	NAK state	NAK status Indicates the following results: 0: Module replies to non-NAK handshake according to FIFO status. 1: The module replies with a NAK handshake on this endpoint. When the application or module sets this bit, the module stops receiving data even if there is room in the RxFIFO to continue receiving packets. Regardless of how this bit is set, the module always responds to SETUP packets with an ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R
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
b1~b0	MPSIZ	Maximum Packet Size	Maximum packet size The maximum packet size for Control OUT Endpoint 0 is the same value programmed in Control IN Endpoint 0. 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes	R/W

31.7.4.17 USBFS Device OUT Endpoint X Control Register (USBFS_DOEPCTLx)(x=1..5)

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, cleared by USBFS 0: endpoint disable 1: Endpoint enable	R/W
b30	EPDIS	Endpoint Prohibition	Endpoint disable An application can set this bit to stop data sending/receiving on an endpoint even before the transfer is complete on that endpoint. The application must wait for an endpoint disable interrupt to occur before the endpoint is considered disabled. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only if the Endpoint Enable bit for that endpoint is set.	R/W
b29	SD1PID/ SODDFRM	Set DATA1 PID/ Set Odd Frame	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. Writing to this field sets the Even/odd-numbered frame (EONUM) field to odd-numbered frames.	R
b28	SD0PID/ SEVNFRM	Set DATA0 PID/ SEVNFRM	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. Writing to this field sets the Even/odd-numbered frames (EONUM) field to even frames.	R
b27	SNAK	Set bitNAK bit	Set NAK Writing to this bit will set the NAK bit of the endpoint. With this bit, the application can control the sending of NAK handshake signals on the endpoint. The module can also set this bit of the OUT endpoint to 1 when a transfer complete interrupt occurs or after SETUP is received on the endpoint.	R/W
b26	CNAK	Zero NAK-bit	Clear NAK Writing to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	Reserved	-	The reset value must be maintained.	R
b21	STALL	STALL handshake	STALL handshake When this endpoint receives a SETUP token, the application can only set this bit, and the module will clear it. If the NAK bit, the 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.	R/W
b20	SNPM	Listen mode	Snoop mode This bit is used to configure the endpoint into listen mode. In the monitor mode, the module will no longer check the correctness of the received data.	R/W
b19~b18	EPTYP	Endpoint Type	Endpoint type The following are the transport types supported by this logical endpoint. 00: control 01: synchronous	R/W

			10: Batch 11: interrupt	
b17	NAKSTS	NAK状态	<p>NAK status Indicates the following results: 0: Module replies to non-NAK handshake according to FIFO status. 1: The module replies with a NAK handshake on this endpoint.</p> <p>When an application or module sets this bit: Even if there is room in the RxFIFO to hold the incoming packet, the module stops receiving any data on the OUT endpoint. Regardless of how this bit is set, the module always responds to SETUP packets with an ACK handshake.</p>	R
b16	EONUM/ DPID	Even/odd-numbered frames/ Endpoint data PID	<p>Even/odd frame Applies to synchronous OUT endpoints only. Indicates the number of the frame in which the module sends/receives synchronized data for this endpoint. The application must program the even/odd-numbered frame number through the SEVNFRM and SODDFRM fields in this register so that this endpoint transmits/receives synchronously data. 0: even frame 1: odd-numbered frames</p> <p>DPID: Endpoint data PID Applies to interrupt/bulk OUT endpoints only. Contains the PID of packets 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 programs the DATA0 or DATA1 PID using the SD0PID register field. 0: DATA0 1: DATA1</p>	R/W
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 to 0 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 register accordingly and set this bit to 1.	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	MPSIZ	Maximum Packet Size	Maximum packet size Applications must program this field to the maximum packet size for the current logical endpoint. This value is in bytes.	R/W

31.7.4.18 USBFS Device OUT Endpoint X Interrupt Register (USBFS_DOEPINTx) (x=0..5)

Offset address: 0xb08 + (endpoint number × 0x20)

Reset value: 0x0000 0080

This register indicates the status of the endpoint on USB and AHB related events. The application must read this register when the OUT endpoint interrupt bit (OEPINT bit in USBFS_GINTSTS) in the USBFS_GINTSTS register is set. Before the application can read this register, it must first read the USBFS_DAINT register to get the exact endpoint number of the USBFS_DOEPINTx registers. The application must clear the corresponding bits in this register before clearing 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		
b6	B2BSTUP	Received consecutive SETUP packets	Back-to-back SETUP packets received Applies to control OUT endpoints only. This bit indicates that this 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		
b4	OTEPEDIS	An OUT token was received when the endpoint prohibited	OUT token received when endpoint disabled Applies to control OUT endpoints only. Indicates that an OUT token was received when the endpoint was not yet enabled, generating an interrupt. Write 1 to clear by software.										R/W		
b3	STUP	SETUP stage completed	SETUP phase done Applies to control OUT endpoints only. Indicates that the SETUP phase of the control endpoint is complete, and no consecutive SETUP packets are received in the current control transfer. On this interrupt, the application can decode the received SETUP packet. Write 1 to clear by software.										R/W		
b2	Reserved	-	The reset value must be maintained.										R		
b1	EPDSD	Endpoint disable interrupt	Endpoint disabled interrupt This bit indicates that the endpoint has been disabled by the application. Write 1 to clear by software.										R/W		
b0	XFRC	Transfer complete interrupt	Transfer completed interrupt This field indicates that transfers set on this endpoint have completed transfers on USB and AHB. Write 1 to clear by software.										R/W		

31.7.4.19 USBFS Device OUT Endpoint 0 Transfer Size Register (USBFS_DOEPTSIZ0)

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 has cleared the endpoint enable bit.

Non-zero endpoints use the registers of endpoints 1~5.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved	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			
b30~b29	STUPCNT	SETUP packet count		SETUP packet count This field specifies the number of SETUP packets the endpoint can receive consecutively. 01: 1 packet 10: 2 packet 11: 3 packet								R/W			
b28~b20	Reserved	-		The reset value must be maintained.								R			
b19	PKTCNT	Packet count		Packet count The number of packets that should be received in one transfer. Before the endpoint is enabled, the software sets this bit. After the transmission starts, every time a data packet is received, the value of this field is automatically reduced.								R/W			
b18~b7	Reserved	-		The reset value must be maintained.								R			
b6~b0	XFRSIZ	transfer size		Transfer size Indicates the amount of data contained in one data transmission of endpoint 0, in bytes. The module interrupts the application only after the application has transferred 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			

31.7.4.20 USBFS Device OUT Endpoint X Transfer Size Register (USBFS_DOEPTSIZx)(x=1..5)

Offset address: 0xB10 + (endpoint number × 0x20)

Reset value: 0x0000 0000

The application must modify this register before enabling the endpoint. After enabling the endpoint through the endpoint enable bit (EPENA bit in USBFS_DOEPCTLx) in the USBFS_DOEPCTLx register, the module modifies this register. The application can read this register only after the module has cleared the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved		PKTCNT[9:0]										XFRSIZ[18:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
XFRSIZ[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	The reset value must be maintained.	R
b28~b19	PKTCNT	Packet count	Packet count Indicates the number of data packets contained in a data transmission on this endpoint. This field will be decremented after every packet (max size or short) written to the RxFIFO.	R/W
b18~b0	XFRSIZ	transfer size	Transfer size This field contains the amount of data contained in a data transmission of the current endpoint, in bytes. The module interrupts the application only after the application has transferred 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

31.7.4.21 USBFS Device OUT Endpoint X Transfer Size Register

(USBFS_DOEPDMAx)(x=0..5)

Offset address: 0xB14 + (endpoint number × 0x20)

Reset value: 0xFFFF XXXX

This register is used to set the DMA address in device endpoint 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 start address of the external memory area when using DMA for data transfer 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 for every AHB transfer. The application must set a double word aligned address.	R/W											

31.7.5 USBFS Clock Gating Control Register

Control HCLK and PHY clock by gating the clock control register to reduce power consumption. Bit values in register descriptions are in binary unless otherwise noted.

31.7.5.1 USBFS Clock Gating Control Register (USBFS_GCCTL)

Offset address: 0xE00

Reset value: 0x0000 0000

This register is available in both host and device modes.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16																																																
Reserved																																																															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																
Reserved																																																															
<table border="1" style="width: 100%;"><tr><td style="width: 10px;"></td><td style="width: 10px;"></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>																																																															
Bit	Marking	Place name	Function												Read and write																																																
b31~b2	Reserved	-	The reset value must be maintained.												R																																																
b1	GATEHCLK	Gated HCLK	Gate HCLK When the USB communication is suspended or the session is invalid, the application program will set this bit to stop clocking the modules except the AHB bus slave interface, master interface and wake-up logic. The application clears this bit when USB communication resumes or a new session starts.												R/W																																																
b0	STPPCLK	stop phy clock	Stop PHY clock The application sets this bit to stop the PHY clock when USB communication is suspended, the session is invalidated, or the device is disconnected. The application program clears this bit when USB communication resumes.												R/W																																																

32 Cryptographic Coprocessing Module (CPM)

32.1 Introduction

The encryption co-processing module (CPM) includes three sub-modules: AES encryption and decryption algorithm processor, HASH security hash algorithm, and TRNG true random number generator.

The AES encryption and decryption algorithm processor follows the standard data encryption and decryption standards, and can realize encryption and decryption operations with a key length of 128 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 US National Institute of Standards and Technology, and can process messages with a length of no more than 2^{64} bits. Produces a 256-bit message digest output.

TRNG true random number generator is a random number generator based on continuous analog noise, providing 64bit random numbers.

32.2 Encryption and Decryption Algorithm Processor (AES)

32.2.1 Algorithm Introduction

The AES encryption and decryption algorithm is a key iterative block cipher, which includes the repeated effect of round transformation on the state. The round transformation of the encryption process consists of four operations: SubBytes, ShiftRows, MixColumns, AddRoundKey. Among them, SubBytes is to find 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 Bit; MixColumns linearly transforms each column of the state; AddRoundKey performs bit-by-bit XOR operation on each byte in the state and the round key. Figure 32-1 below is a schematic diagram of the encryption process:

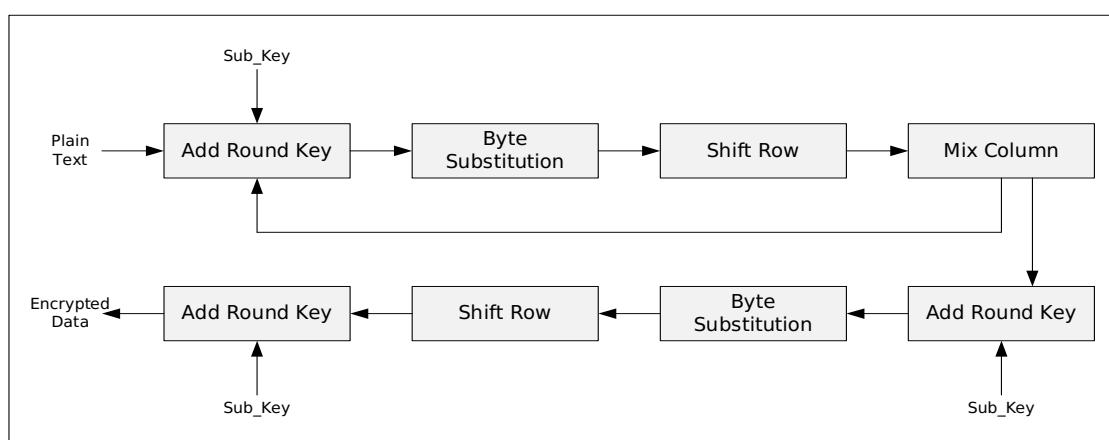


Figure 32-1 AES Encryption Flow Diagram

In the figure above, Sub_Key refers to the subkey of each round. In addition to the initial key for the initial transformation, the subkey used for the subsequent round transformation needs to be expanded from the initial key, and the key expansion process and encryption process are 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 128bits, the number of running rounds is 10; when the key is 192bits, the number of running rounds is 12; when the key is 256bits, the number of running rounds is 14. Except for the lack of MixColumn transformation in the last round, the rest of the rounds perform a complete round transformation operation. The AES on this product only supports the encryption and decryption process with a key length of 128bits.

The decryption process is different from the encryption process. First, the expansion of all keys must be completed. The decryption process is used backwards from the last round of expansion, and then the four operations of the round transformation become the corresponding inverse operations: InvSubByte the s , InvShiftRows , InvMixColumns , AddRoundKey. The modular inverse operation invSubBytes is still maintained, but the affine transformation is changed to inverse transformation; InvShiftRows and InvMixColumns become corresponding inverse transformations; AddRoundKey remains unchanged.

32.2.2 Encryption Operation Process

The encryption operation flow of AES is as follows:

- 1) Write 128bits plaintext into the data register (AES_DR).
- 2) Write the encryption key into the key register (AES_KR).
- 3) Set bits in the control register (AES_CR), including:
 - a) Set CR. Mode to 0
 - b) Write 1 to AES_CR. START to start the module for operation
Note: Steps a and b can be carried out at the same time
- 4) Determine whether the module operation is finished.
Constantly read AES_CR.START, if its value becomes 0, it means the operation is over
- 5) Read the data register (AES_DR) to obtain 128-bit ciphertext.
- 6) If you want to continue to perform new calculations, go back to step 1, otherwise end.

32.2.3 Decryption Operation Process

The AES decryption operation flow is as follows:

- 1) Write the 128bits ciphertext into the data register (AES_DR).
- 2) Write the decryption key into the key register (AES_KR).
- 3) Set bits in the control register (AES_CR), including:
 - a) Set CR. Mode to 1

- b) Write 1 to AES_CR.START to start the module for operation

Note: Steps a and b can be carried out at the same time

- 4) Determine whether the module operation is finished.

Constantly read AES_CR.START, if its value becomes 0, it means the operation is over

- 5) Read the data register (AES_DR) to get 128-bit plaintext.

- 6) If you want to continue to perform new calculations, go back to step 1, otherwise end.

32.2.4 Encryption and Decryption Time

The time required by the AES module from starting an operation (AES_CR.START writes 1) to the end of the operation (AES_CR.START returns to 0) is as follows:

Encryption process	440 CPU clock cycles
Decryption process	580 CPU clock cycles

32.2.5 Operation Precautions

- 1) After power on, the module performs an asynchronous reset operation. The clock needs to be stable and effective before the reset is released, and it should continue to be stable in subsequent operations.
- 2) During the encryption and decryption process, the data register will change, if the data of the next operation operation is the result of this operation, there is no need to rewrite the data.
- 3) In the case of encrypting and decrypting a large amount of data with the same key, there is no need to write the key repeatedly.
- 4) The method of judging the end of the operation of the module: read AES_CR.START continuously, if its value becomes 0, it means the end of the operation.

32.2.6 Register Description

BASE ADDR: 0x4000_8000

Register name	Symbol	Offset	Bit width	Reset value
AES control register	AES_CR	0x0000	32	0x0000_0000
AES data register 0	AES_DR0	0x0010	32	0x0000_0000
AES data register 1	AES_DR1	0x0014	32	0x0000_0000
AES data register 2	AES_DR2	0x0018	32	0x0000_0000
AES data register 3	AES_DR3	0x001C	32	0x0000_0000
AES key register 0	AES_KR0	0x0020	32	0x0000_0000
AES key register 1	AES_KR1	0x0024	32	0x0000_0000
AES key register 2	AES_KR2	0x0028	32	0x0000_0000
AES key register 3	AES_KR3	0x002C	32	0x0000_0000

Note:

- The write operation to all registers is only valid when the module is in the idle state (AES_CR.START=0), otherwise the write operation is ignored; the read operation to registers other than the control register (AES_CR) is only valid when the module is in the idle state (AES_CR .START=0) to read valid data, otherwise unknown data can be read; the read operation of the control register (AES_CR) can be performed at any time, and valid data can be read.

32.2.6.1 AES Control Register (AES_CR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
Bit	Marking	Place name	Function	Read and write											
b31~b2	Reserved	-	Read at "0", write at "0"	R/W											
b1	MODE	Functional selection	0: encryption operation 1: Decryption operation	R/W											
b0	START	Start up	0: The operation of this module is completed or has not been started 1: Start this module for calculation	R/W											

32.2.6.2 AES Data Register (AES_DR)

The data register consists of four 32-bit registers consisting of 128-bit data, which are used to store the plaintext that needs to be encrypted or the ciphertext that needs to be decrypted before the operation of the module, and store 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.

Digits : 128 bit

Offset address: 0x10 - data[31:0]

0x14 - data[63:32]

0x18 - data[95:64]

0x1C - data[127:96]

Reset value : 0x0000_0000 (per 32-bit register)

Data example: 0x00112233445566778899AABBCCDDEEFF

Offset address	Register name	Fill in data
0x10	DR0	0x3322_1100
0x14	DR1	0x7766_5544
0x18	DR2	0xBBAA_9988
0x1C	DR3	0xFFEE_DDCC

32.2.6.3 AES Key Register (AES_KR)

The key register consists of four 32-bit registers to form a 128-bit key, which is used to store the input initial key.

Number of bits : 128 bits

Offset address:0x20 - key[31: 0]

0x24 - key[63: 32]

0x28 - key[95: 64]

0x2C - key[127: 96]

Reset value : 0x0000_0000 (per 32-bit register)

Data example:0x000102030405060708090A0B0C0D0E0F

Offset address	Register name	Fill in data
0x20	KR0	0x0302_0100
0x24	KR1	0x0706_0504
0x28	KR2	0x0B0A_0908
0x2C	KR3	0x0F0E_0D0C

32.3 Secure Hash Algorithm (HASH)

32.3.1 Algorithm Introduction

The steps of the secure hash algorithm are as follows:

First pad the message so that its length is exactly a number that is only 64 bits less than a multiple of 512. The padding method is to attach a 1 to the back of the message, followed by the required multiple 0s, and then attach a 64-bit message length (before padding) to make the message length exactly an integer multiple of 512 bits.

Secondly, 8 32-bit variables A, B, C, D, E, F, G, and H are initialized with hexadecimal. Then start the main loop of the algorithm, process 512-bit messages at a time, 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, cyclic shift, logic operation, modulo 2³² addition, etc. The operation process is shown in Figure 32-2 below. The final output is formed by cascading A, B, C, D, E, F, G, H. Among them, W_t is the temporary value used in the tth step obtained from the 512-bit message, K_t is the constant value used in the tth step, and t (0 ≤ t ≤ 63) is a step in the 64-step cycle.

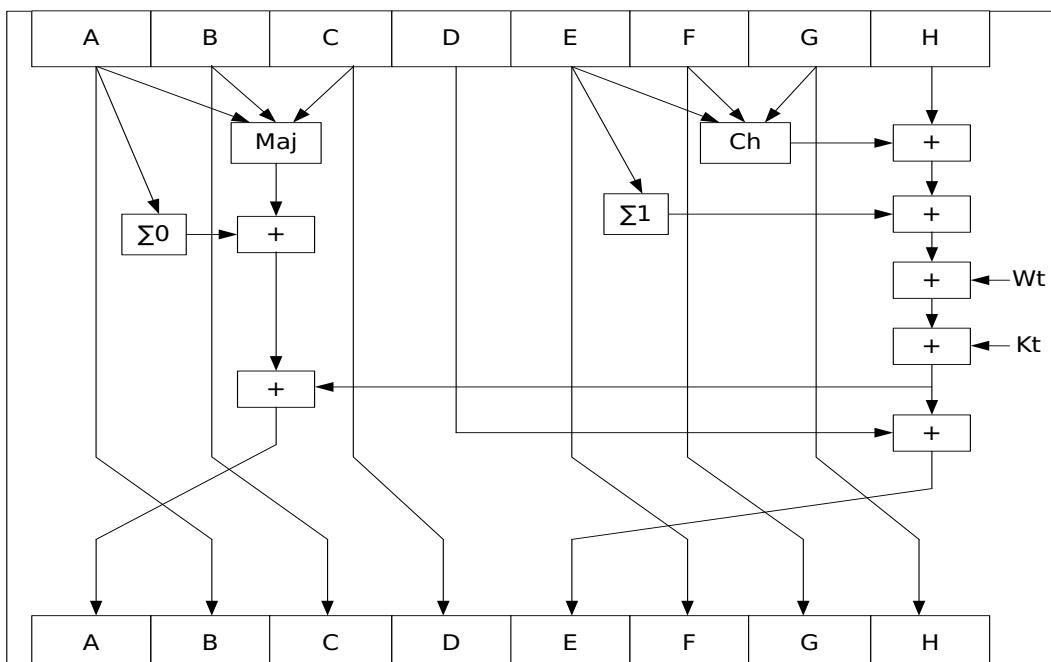


Figure 32-2 HASH Algorithm Flowchart

32.3.2 Operating Procedures

After power on, the module performs an asynchronous reset operation. The clock needs to be stable and effective before the reset is released, and it should continue to be stable in subsequent operations.

The operation process of the HSAH module is as follows:

- 1) The software fills the original data according to the algorithm rules, and groups the filled messages by 512 bits.
- 2) Write the manipulated data into the data register (HASH_DR).
- 3) If this operation is the first group of data in the message group, write 1 to the HASH_CR . FST_GRP bit.
- 4) Write 1 to HASH_CR.START to start the module for calculation.

Note: 3 and 4 above can be done at the same time

- 5) To judge whether the operation of this module is completed, use the following methods:
Keep reading HASH_CR.START until the bit is read as 0, which means the operation is complete
- 6) If this operation is not the last set of data in the message group, return to 2).
- 7) If this operation is the last set of data in the message group, read the summary register (HASH_HR) to obtain the result of this operation. Return to step 1 if further calculations are required.

32.3.3 Message Padding

The padding packet processing steps of SHA-256 are as follows:

1. Raw message grouping

Divide the original message into L groups with a size of 512 bits. Let the total number of bits of the original message be I. If $I \% 512 < 448$, then the grouping number L is $I / 512$; if $I \% 512 \geq 448$, then the grouping number L is $I / 512 + 1$.

2. Add length

① Add padding:

Add padding bits at the end of the $1 / 512$ th group of the message packet: one 1 and several 0s, the number of 0s can be zero. If $I \% 512 < 448$, padding makes the length of the data bits meet the length of $448 \bmod 512$ (the last 64 bits are reserved for the original message length); The 512-bit data block of the group is filled, and the first 448 bits of the L ($L = I / 512 + 1$) group are filled with 0.

② Add raw message length:

A 64bit block, representing the length of the original message, is a 64bit unsigned integer. Add the original message length to the last 64bit of the Lth packet.

An example to illustrate the process of filling a group is as follows:

1) Padding example 1:

The original message is the character string "ABCDE", and its ASCII code represented by a binary bit string is: "01100001 01100010 01100011 01100100 01100101", the steps to add the length are as follows:

- A. Add "1". The filled message is "01100001 01100010 01100011 01100100 01100101 1".
- B. Add "0". Because the length of the original message is 40bit, 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 the 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) Padding example 2:

The original message was the string

"ABCDBCDECDEFDEFGEFGFHIGHIGHIJHIJKIJKLJKLMKLMN LMNOMNOPNOPQ". Each character can be converted to 8bit through its ASCII code, so the length of the message is $l = 56 * 8 = 448$.

- A. Add "1" and "0". The padded message (hexadecimal) is the first message block:

61626364 62636465 63646566 64656667
65666768 66676869 6768696A 68696A6B
696A6B6C 6A6B6C6D 6B6C6D6E 6C6D6E6F
6D6E6F70 6E6F7071 80000000 00000000

- B. Add the original message length. The original message length of 448 is expressed in two 32bit words (hexadecimal): 00000000 000001C0.

The padded message (hexadecimal) is the second message block:

00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 000001C0

32.3.4 Register Description

BASE ADDR: 0x4000_8400

Register name	Symbol	Offset	Bit width	Reset value
HASH control register	HASH_CR	0x0000	32	0x0000_0000
HASH digest register 7	HASH_HR7	0x0010	32	0x0000_0000
HASH digest register 6	HASH_HR6	0x0014	32	0x0000_0000
.....
HASH digest register 1	HASH_HR1	0x0028	32	0x0000_0000
HASH digest register 0	HASH_HR0	0x002C	32	0x0000_0000
HASH data register 15	HASH_DR15	0x0040	32	0x0000_0000
HASH data register 14	HASH_DR14	0x0044	32	0x0000_0000
.....
HASH data register 1	HASH_DR1	0x0078	32	0x0000_0000
HASH data register 0	HASH_DR0	0x007C	32	0x0000_0000

32.3.4.1 HASH Control Register (HASH_CR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved														FST_GRP	START
Bit	Marking	Place name	Function	Read and write											
b31~b2	Reserved	-	Read as "0"	R/W											
b1	FST_GRP	The first group of message groupings	0: This operation is not the first group operation of message grouping 1: This operation is the first group operation of message grouping	R/W											
b0	START	Start up	0: The operation of this module is completed or has not been started 1: Start this module for calculation	R/W											

Note:

- The operation method of the START bit is: after the software writes 1 to this bit, the module will start to run; after the end of this run, the hardware will automatically clear this bit to 0; if the software queries this bit to 0, it means that this run is completed.
- 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.

32.3.4.2 HASH Digest Register (HASH_HR)

Number of bits : 256 bits

Offset address:0x10 - hash[255:224]

- 0x14 - hash[223:192]
- 0x18 - hash[191:160]
- 0x1C - hash[159:128]
- 0x20 - hash[127:96]
- 0x24 - hash[95:64]
- 0x28 - hash[63:32]
- 0x2C - hash[31:0]

Reset value : 0x0000_0000 (per 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 operation of the module is completed, the message summary can be obtained by reading this register				R/W				

Note:

- This register is spliced by 8 32-bit registers. During the access, 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 the write operation 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.

32.3.4.3 HASH Data Register (HASH_DR)

Number of bits : 512 bits

Offset address: 0x040 - data[511: 480]

0x044 - data[479: 448]

0x048 - data[447: 416]

0x04C - data[415: 384]

.....

0x070 - data[127: 96]

0x074 - data[95: 64]

0x078 - data[63: 32]

0x07C - data[31: 0]

Reset value : 0x0000_0000 (per 32-bit register)

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]															
<hr/>															
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.
- 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 operation to this register.
- A read of this register will always return all zeros.

32.4 True Random Number Generator (TRNG)

32.4.1 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 shown in Figure 32-3 below. The random number generator is an analog random number oscillator circuit, which is used to obtain random noise; the algorithm module captures random noise and saves the result to the data module, and outputs it through the bus; the control module controls the mode and start of TRNG.

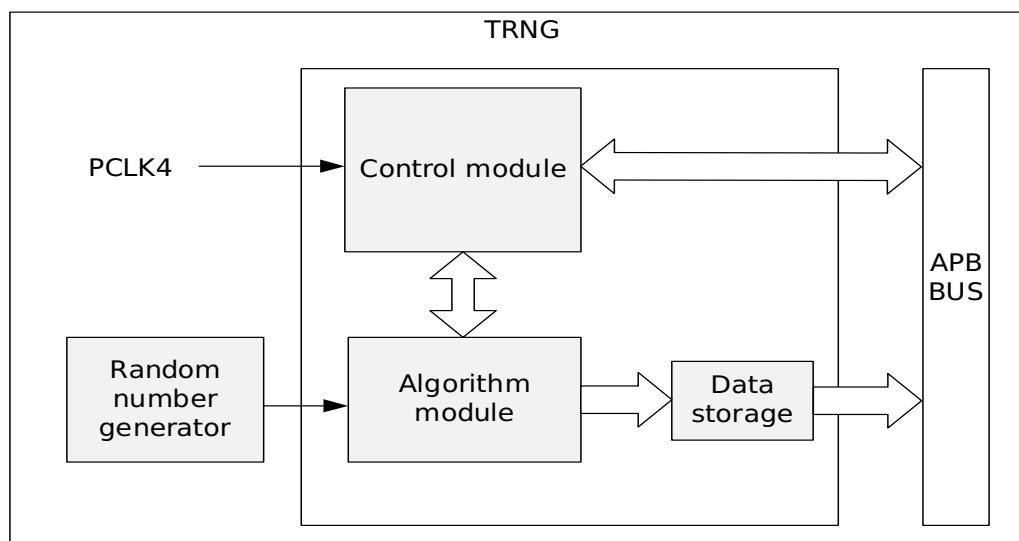


Figure 32-3 TRNG System Block Diagram

32.4.2 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 number (read TRNG_DR).
5. Turn off the random number generator circuit (set the EN bit of TRNG_CR to 0).

32.4.3 Interrupt and Event Output

When the random number generation is finished, the register bit TRNG_CR.RUN is cleared by hardware and generates a random number generation interrupt request (TRNG_END). When the random number generation ends, an event output is also generated, which can trigger the linkage of other modules.

32.4.4 Operation Precautions

In order to obtain a good random number, please set the frequency of the peripheral clock PCLK4 below 1MHz.

32.4.5 Register Description

BASE ADDR: 0x4004_1000

Register name	Symbol	Offset	Bit width	Reset value
TRNG control register	TRNG_CR	0x0000	32	0x0000_0000
TRNG Mode Register	TRNG_MR	0x0004	32	0x0000_0012
TRNG data register 0	TRNG_DR0	0x000C	32	0x0800_0000
TRNG data register 1	TRNG_DR1	0x0010	32	0x0800_0200

32.4.5.1 TRNG Control Register (TRNG_CR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved														RUN	EN
Bit	Marking	Place name	Function										Read and write		
b31~b3	Reserved	-	Read as "0", write as "0"										R		
b2	Reserved	-	Read as "0", write as "0"										R/W		
b1	RUN	Random number operation starts	0: Stop random number operation 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: Turn on the analog random number generator circuit										R/W		

32.4.5.2 TRNG Mode Register (TRNG_MR)

Reset value: 0x0000_0012

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
CNT[2:0]															
-															
Bit	Marking	Place name	Function	Read and write											
b31~b5	Reserved	-	Read as "0", write as "0"	R											
b4~2	CNT[2:0]	Shift count control bit	When capturing random noise, the number of shift control bits 011: Shift 32 times 100: Shift 64 times 101: Shift 128 times 110: Shift 256 times 000~010, 111: Function reserved bits	R/W											
b1	Reserved	-	Read as "1", write as "1"	R/W											
b0	LOAD	Load control bit	Whether the data register is loaded with a new initial value from the random number generator before the random number is generated 0: Do not load new initial values 1: Load new initial values	R/W											

32.4.5.3 TRNG data register (TRNG_DR)

Reset value: DR0: 0x0800_0000

DR1: 0x0800_0200

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DATA[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DATA[15:0]															
-															
Bit	Marking	Place name	Function	Read and write											
b31-b0	DATA[31:0]	Random number	64-bit random number	R											

33 Data Computing Unit (CRC)

33.1 Introduction

The Data Computing Unit (Data Computing Unit) is a module that simply processes data without the help of a CPU. Each DCU unit has 3 data registers, which can add, subtract and compare the size of 2 data, as well as the window comparison function. This product is equipped with 4 DCU units, and each unit can independently complete its own functions.

Functional summary:

- Can carry out 4 kinds of data processing: 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 registers or triggered by other peripheral circuit events.
- The addition and subtraction operations can automatically halve the results once, and put the halved results and the addition and subtraction results into two data registers for use by other modules.
- The comparison mode can compare two data between DATA0 and DATA1 registers, and between DATA0 and DATA2 registers. You can choose to generate interrupts and flags when 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. The event signals are used to start peripheral circuits generated by DCU when other peripheral circuits with hardware trigger start function select DCU as the trigger source Get moving.

33.2 Functional Description

33.2.1 Additive Mode

The addition mode calculates the sum of DATA0 and DATA1, where DATA0 is used as the summand and DATA1 is used as the addend. Every time the DATA1 register is written, a (DATA0+DATA1)/2 operation is performed, the result of DATA0+DATA1 is stored in DATA0, and the result of (DATA0+DATA1)/2 is stored in DATA2. When the result of DATA0+DATA1 exceeds 0xFF (8bit mode) or 0xFFFF (16bit mode) or 0xFFFF_FFFF (32bit mode), a flag bit is generated and an interrupt is generated.

33.2.2 Subtraction Mode

The subtraction mode calculates the difference between DATA0 and DATA1, where DATA0 acts as the minuend and DATA1 acts as the subtrahend. Every time the DATA1 register is written, a (DATA0-DATA1)/2 operation is performed, the result of DATA0-DATA1 is stored in DATA0, and the result of (DATA0-DATA1)/2 is stored in DATA2. When the result of DATA0-DATA1 is less than 0x0 (8bit, 16bit, 32bit mode), a flag bit is generated and an interrupt is generated.

33.2.3 Hardware Triggered Boot Mode

The DCU can trigger and start operations according to events generated by peripheral circuits. When using the hardware trigger start mode, it is necessary to first enable the peripheral circuit trigger function enable bit of the function clock control 0 register (FCG0). 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 start source of the peripheral circuit to be selected in the trigger source selection register (DCU_TRGSELx). When the peripheral circuit event occurs, the event signal will be input to the DCU and trigger the DCU to start computing. The hardware trigger start mode includes trigger plus mode and trigger minus mode. In the trigger plus mode, every time an event trigger occurs, the DCU will start and perform a calculation of (DATA0+DATA1)/2, the result of DATA0+DATA1 is stored in DATA0, and the result of (DATA0+DATA1) /2 is stored in DATA2. When the result of DATA0+DATA1 exceeds 0xFF (8bit mode) or 0xFFFF (16bit mode) or 0xFFFF_FFFF (32bit mode), a flag bit is generated and an interrupt is generated. In the trigger subtraction mode, every time an event trigger occurs, the DCU will start and perform a calculation of (DATA0-DATA1)/2, the result of DATA0-DATA1 is stored in DATA0, and the result of (DATA0-DATA1) /2 is stored in DATA2 . When the result of DATA0-DATA1 is less than 0x0 (8bit, 16bit, 32bit mode), a flag bit is generated and an interrupt is generated.

33.2.4 Compare Mode

Comparing the size 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, DATA0 is equal to DATA2, generate a flag and generate an interrupt. In the comparison mode, you can choose the conditions for data comparison, compare after writing DATA0 or write any data register.

33.2.5 Interrupt and Event Signal Output

DCU has a variety of interrupts and event outputs for triggering and starting other peripheral circuits for users to choose. The control of interrupt and event output is controlled by the interrupt condition selection register (DCU_INTEVTSEL). When an event signal needs to be output, the user needs to enable the corresponding control bit of the interrupt condition selection register (DCU_INTEVTSEL). Each DCU unit outputs a DCU event signal, respectively DCU1/DCU2/DCU3/DCU4 in the event list. When an interrupt neededs to be generated when the corresponding event occurs, the user needs to enable the corresponding control bit of the interrupt condition re re Gister (DCU_INTEVTSEL) and at the Same Time Set the Inten Bit of the Control Register (DCU_CTL) to 1. Each DCU unit outputs a DCU interrupt signal, respectively DCU1/DCU/DCU3/DCU4 in the interrupt list.

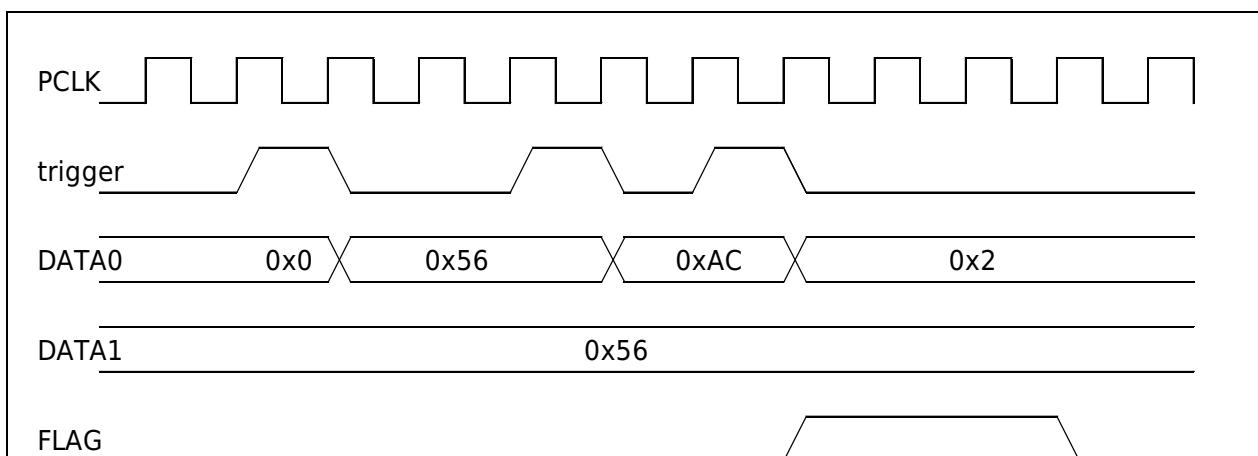
33.3 Application Examples

33.3.1 Additive Mode

Configure the register to achieve: select the addition mode, and the data width is 16bit. Write 0xFF00 and 0x55 in DATA0 and DATA1 respectively, at this time the calculation result is 0xFF55, and the result is saved in DATA0. Continue to write 0xFF in DATA1, at this time, the calculation result overflows, and the result FLAG is generated. Write FLAG register by software to clear FLAG.

33.3.2 Trigger Plus Mode

Configure the register to achieve: select the trigger plus mode, and the data width is 8bit. Write 0x00 and 0x56 in DATA0 and DATA1 respectively, and write the peripheral circuit event number in the trigger source selection register. Make the peripheral circuit act and generate an event, the DCU is triggered by the event and performs an addition operation, and the result is 0x56 and stored in DATA0. After 3 consecutive triggers, the calculation result overflows and the result FLAG is generated. Write FLAG register by software to clear FLAG.



33.3.3 Compare Mode

Configure the register to achieve: select the comparison mode, and the data width is 16bit. The FLAG output condition is DATA0>DATA1, and the comparison starts after writing DATA0. First, write 0xFFFF and 0xAAAA in DATA1 and DATA0 respectively. At this time, FLAG is not generated because DATA0>DATA1 is not satisfied. Then write 0x8888 in DATA1. Although DATA0>DATA1 is satisfied at this time, because it is set to write DATA0 and start comparison, FLAG is not generated. When DATA0 writes 0x9999 and meets the FLAG generation condition, FLAG is generated and reset by writing the FLAG register.

33.4 Register Description

Register overview

Unit 1

Name	Abbreviation	Note	Address
DCU1 Control Register	DCU1_CTL	Configure the action mode of the DCU	0x4005_2000
DCU1 flag register	DCU1_FLAG	DCU Result Identification	0x4005_2004
DCU1 Data Register 0	DCU1_DATA0	Store operation data	0x4005_2008
DCU1 Data Register 1	DCU1_DATA1	Store operation data	0x4005_200C
DCU1 Data Register 2	DCU1_DATA2	Store operation data	0x4005_2010
DCU1 Flag Reset Register	DCU1_FLAGCLR	Clear the result flag of DCU	0x4005_2014
DCU1 interrupt condition selection register	DCU1_INTEVTSEL	Select the conditions for the DCU to generate an interrupt	0x4005_2018

Unit 2

Name	Abbreviation	Note	Address
DCU2 Control Register	DCU2_CTL	Configure the action mode of the DCU	0x4005_2400
DCU2 flag register	DCU2_FLAG	DCU Result Identification	0x4005_2404
DCU2 Data Register 0	DCU2_DATA0	Store operation data	0x4005_2408
DCU2 Data Register 1	DCU2_DATA1	Store operation data	0x4005_240C
DCU2 Data Register 2	DCU2_DATA2	Store operation data	0x4005_2410
DCU2 Flag Reset Register	DCU2_FLAGCLR	Clear the result flag of DCU	0x4005_2414
DCU2 interrupt condition selection register	DCU2_INTEVTSEL	Select the conditions for the DCU to generate an interrupt	0x4005_2418

Unit 3

Name	Abbreviation	Note	Address
DCU3 Control Register	DCU3_CTL	Configure the action mode of the DCU	0x4005_2800
DCU3 flag register	DCU3_FLAG	DCU Result Identification	0x4005_2804
DCU3 data register 0	DCU3_DATA0	Store operation data	0x4005_2808
DCU3 data register 1	DCU3_DATA1	Store operation data	0x4005_280C
DCU3 data register 2	DCU3_DATA2	Store operation data	0x4005_2810
DCU3 Flag Reset Register	DCU3_FLAGCLR	Clear the result flag of DCU	0x4005_2814
DCU3 interrupt condition selection register	DCU3_INTEVTSEL	Select the conditions for the DCU to generate an interrupt	0x4005_2818

Unit 4

Name	Abbreviation	Note	Address
DCU4 Control Register	DCU4_CTL	Configure the action mode of the DCU	0x4005_2C00
DCU4 flag register	DCU4_FLAG	DCU Result Identification	0x4005_2C04
DCU4 Data Register 0	DCU4_DATA0	Store operation data	0x4005_2C08
DCU4 Data Register 1	DCU4_DATA1	Store operation data	0x4005_2C0C
DCU4 Data Register 2	DCU4_DATA2	Store operation data	0x4005_2C10
DCU4 Flag Reset Register	DCU4_FLAGCLR	Clear the result flag of DCU	0x4005_2C14
DCU4 interrupt condition selection register	DCU4_INTEVTSEL	Select the conditions for the DCU to generate an interrupt	0x4005_2C18

33.4.1 DCU Control Register (DCU_CTL)

Register Description: This register is used to configure the action mode of the DCU.

Address: 0x4005_2000, 0x4005_2400, 0x4005_2800, 0x4005_2C00

Reset value: 0x8000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
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[2:0]					

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
b8	COMPTRG	Timing of compare mode trigger comparison	0: compare after writing DATA0 1: compare after writing DATA0 or DATA1 or DATA2	R/W
b7~b5	Reserved	-	Read 0 when reading, write 0 when writing	R
b4~b3	DATASIZE[1:0]	Data size	00: 8bit 01: 16bit 10: 32bit	R/W
b2~b0	MODE[2:0]	Action mode	000: DCU invalid 001: Addition mode, the operation is performed after data is written in the DATA1 register 010: Subtraction mode, the operation is performed after data is written in the DATA1 register 011: Hardware trigger addition mode, triggered by other peripheral circuits to start addition operation 100: The hardware triggers the subtraction mode, which is triggered by other peripheral circuits to start the subtraction operation 101: Compare Mode	R/W

33.4.2 DCU Flag Register (DCU_FLAG)

Register Description: This register generates the result flag of the DCU.

Address: 0x4005_2004, 0x4005_2404, 0x4005_2804, 0x4005_2C04

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16					
Reserved																				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0					
Reserved														FLAG_- GT1	FLAG_- EQ1	FLAG_- LS1	FLAG_- GT2	FLAG_- EQ2	FLAG_- LS2	FLAG_- OP
Bit	Marking	Place name	Function															Read and write		
b31~b7	Reserved	-	Read 0 when reading, write 0 when writing															R		
b6	FLAG_GT1	greater than flag 1	In comparison mode, set when DATA0>DATA1, and clear when the CLR_GT1 bit of the DCU flag reset register DCU_FLAGCLR is written to 1															R		
b5	FLAG_EQ1	equal to flag 1	In comparison mode, set when DATA0=DATA1, and clear when the CLR_EQ1 bit of the DCU flag reset register DCU_FLAGCLR is written to 1															R		
b4	FLAG_LS1	less than flag 1	In comparison mode, set when DATA0<DATA1, and clear when the CLR_LS1 bit of the DCU flag reset register DCU_FLAGCLR is written to 1															R		
b3	FLAG_GT2	greater than flag 2	In comparison mode, set when DATA0>DATA2, and clear when the CLR_GT2 bit of the DCU flag reset register DCU_FLAGCLR is written to 1															R		
b2	FLAG_EQ2	equal to flag 2	In comparison mode, set when DATA0=DATA2, and clear when the CLR_EQ2 bit of the DCU flag reset register DCU_FLAGCLR is written to 1															R		
b1	FLAG_LS2	less than flag 2	In comparison mode, when DATA0<DATA2 is set, it is 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 trigger addition, trigger subtraction mode, set when addition overflows or subtraction underflows, and clear when the CLR_OP bit of the DCU flag reset register DCU_FLAGCLR is written to 1															R		

33.4.3 DCU Data Register (DCU_DATAx) (x=0,1,2)

Register description: used to store operation data. The functions of each data register in each mode are as follows:

	DATA0	DATA1	DATA2
Additive mode	Summand/store result	Addend	Store the halving result
trigger plus mode	Summand/store result	Addend	Store the halving result
Subtraction mode	Minuend/store result	Subtrahend	Store the halving result
Trigger minus mode	Minuend/store result	Subtrahend	Store the halving result
Compare mode	Compared object	Comparison object 1	Comparison object 2
Compare mode (window compare)	Compared object	Window cap	Window lower limit

Address:0x4005_2008, 0x4005_2408, 0x4005_2808, 0x4005_2C08

0x4005_200C, 0x4005_240C, 0x4005_280C, 0x4005_2C0C

0x4005_2010, 0x4005_2410, 0x4005_2810, 0x4005_2C10

Reset value:0x0000_0000

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

Bit	Marking	Place name	Function	Read and write
b31~b0	DAT[31:0]	Operational data	To store calculation data, the actual number of digits used is set according to DCU_CTL.DATASIZE, DATA[7:0] is valid data when DCU_CTL.DATASIZE=00, DATA[15:0] is valid data when DCU_CTL.DATASIZE=01, DATA[31:0] is valid data when DCU_CTL.DATASIZE=10	R/W

33.4.4 DCU Flag Reset Register (DCU_FLAGCLR)

Register Description: This register is used to clear the result flag of DCU.

Address: 0x4005_2014, 0x4005_2414, 0x4005_2814, 0x4005_2C14

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
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	Clear the FLAG_GT1 bit of DCU_FLAG when writing 1, writing 0 has no effect										R/W		
b5	CLR_EQ1	Clear equals flag bit 1	Clear the FLAG_EQ1 bit of DCU_FLAG when writing 1, writing 0 has no effect										R/W		
b4	CLR_LS1	Clear less than flag bit 1	Clear the FLAG_LS1 bit of DCU_FLAG when writing 1, writing 0 has no effect										R/W		
b3	CLR_GT2	Clear greater than flag bit 2	Clear the FLAG_GT2 bit of DCU_FLAG when writing 1, writing 0 has no effect										R/W		
b2	CLR_EQ2	Clear equals flag bit 2	Clear the FLAG_EQ2 bit of DCU_FLAG when writing 1, writing 0 has no effect										R/W		
b1	CLR_LS2	Clear less than flag bit 2	Clear the FLAG_LS2 bit of DCU_FLAG when writing 1, writing 0 has no effect										R/W		
b0	CLR_OP	Clear operation flag	Writing 1 clears the FLAG_OP bit of DCU_FLAG, writing 0 has no effect										R/W		

33.4.5 DCU Interrupt Condition Selection Register (DCU_INTEVTSEL)

Register Description: This register can select the conditions under which the DCU generates an interrupt and outputs an event signal

Address: 0x4005_2018, 0x4005_2418, 0x4005_2818, 0x4005_2C18

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved																
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved																
									SEL_WIN[1:0]	SEL_GT1	SEL_EQ1	SEL_LS1	SEL_GT2	SEL_EQ2	SEL_LS2	SEL_OP
Bit	Marking	Place name	Function													Read and write
b31~b9	Reserved	-	Read 0 when reading, write 0 when writing													R
b8~b7	SEL_WIN[1:0]	Window compare break condition selection	In the comparison mode, when the window comparison condition set by SEL_WIN is met, an interrupt and an output event signal will be generated. When the SEL_WIN setting is valid, an interrupt and an output event signal will not be generated when other comparison conditions are met. 00: Do not generate window comparison interrupt and output event signal. Under this setting, select other interrupt and event signal generation conditions 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: Generate interrupt and output event signal when DATA0 data is outside the window, that is, DATA0>DATA1 or DATA0<DATA2 11: Does not generate any interrupt and event signal in compare mode													R/W
b6	SEL_GT1	Greater than break condition selection 1	In comparison mode and SEL_WIN=00, when DATA0>DATA1 generates interrupt and output event signal, this bit is invalid when SEL_WIN≠00													R/W
b5	SEL_EQ1	Equal to interrupt condition select 1	In comparison mode and SEL_WIN=00, when DATA0=DATA1, an interrupt and output event signal are generated, and this bit is invalid when SEL_WIN≠00													R/W
b4	SEL_LS1	Less than interrupt condition selection 1	In comparison mode and SEL_WIN=00, when DATA0<DATA1, an interrupt and output event signal are generated, and this bit is invalid when SEL_WIN≠00													R/W
b3	SEL_GT2	Greater than break condition selection 2	In comparison mode and SEL_WIN=00, when DATA0>DATA2 generates interrupt and output event signal, this bit is invalid when SEL_WIN≠00													R/W
b2	SEL_EQ2	Equal to interrupt condition select 2	In comparison mode and SEL_WIN=00, when DATA0=DATA2, an interrupt and output event signal are generated, and this bit is invalid when SEL_WIN≠00													R/W
b1	SEL_LS2	Less than interrupt condition selection 2	In comparison mode and SEL_WIN=00, when DATA0<DATA2 generates interrupt and output event signal, this bit is invalid when SEL_WIN≠00													R/W
b0	SEL_OP	Operation flag	Generate interrupt and output event signal when the operation result overflows or underflows in addition and subtraction mode													R/W

34 CRC Operation (CRC)

34.1 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.

34.2 Functional Block Diagram

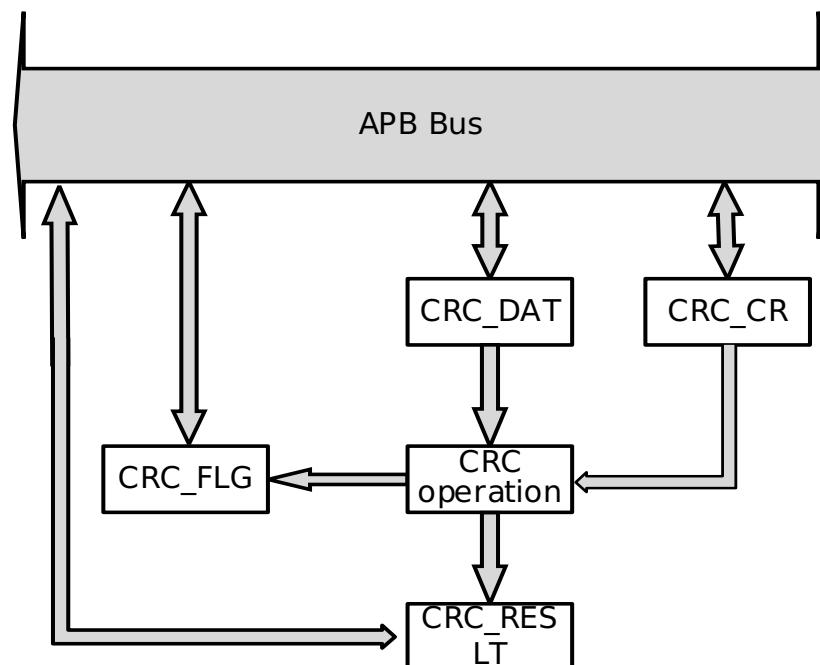


Figure 34-1 CRC Module Block Diagram

34.3 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 CRC32 generating polynomial is $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$. The generating polynomial of CRC16 is $X^{16} + X^{12} + X^5 + 1$.

The functions of this module include CRC code generation and CRC code verification.

34.3.1 CRC Code Generation

CRC encoding is to operate on a string of data to generate a 16/32-bit CRC encoding result.

The operation process is as follows:

1. Set the CRC_CR register bits XOROUT, REFOUT, REFIN, CRC_SEL as required.
2. CRC16(CRC_SEL=0): Write the 16-bit initial value into CRC_REG[15:0] of the CRC_RESLT register.
CRC32(CRC_SEL=1): Write the 32-bit initial value into CRC_REG[31:0] of the CRC_RESLT register.
3. Write the data that needs to be calculated into the CRC_DAT register in turn, and each write operation corresponds to input 1 data (16 bits, 32 bits). For example, if there are 10 data, write 10 times to the CRC_DAT register sequentially.
4. After writing all the data to be calculated into the CRC_DAT register, read the CRC_RESLT register CRC_REG to obtain the 16/32-bit CRC code value.

34.3.2 CRC Check

CRC check is to judge a string of data and 16/32-bit CRC codes to check whether they are correct. It only supports the case where the initial value of 16/32 bits is all 1, and only supports the 16/32 bit CRC checksum.

1. Set CRC_CR as needed, including XOROUT, REFOUT, REFIN, CRC_SEL.
2. CRC16 (CRC_SEL=0): Write the 16-bit initial value into CRC_REG[15:0] of the CRC_RESLT register.
CRC32 (CRC_SEL=1): Write the 32-bit initial value into CRC_REG[31:0] of the CRC_RESLT register.
3. Write the data to be calculated into the CRC_DAT register.
4. Write the 16/32 bit CRC code value to CRC_DATregister.
5. Read the CRC_FLG register, if it is 1, it means the verification is successful, if it is 0, it means the verification fails (for CRC16, you can also read CRC_RESLT[16], CRCFLAG_16 bits to determine whether the verification is successful)

34.3.3 CRC check for XOROUT, REfout, REFIN not all 1

When the options XOROUT, REfout, and REFIN are not all 1, the CRC encoding value following the data input needs to be converted as follows and then written to CRC_DAT for verification:

1. When REfout=0, reverse all BITS of the CRC code value; otherwise, do not reverse.
2. When XOROUT=0, invert the result of (1); otherwise, do not invert;
3. When REFIN=0, completely reverse the BIT of each BYTE in the result of (2); otherwise, do not reverse;
4. Write the result of (3) into the CRC_DAT register for verification.

34.4 Register Description

Table 34-1 Shown is the register list of the CRC module.

CRC_BASE_ADDR: 0x40008C00

Table 34-1 CRC register list

Register name	Symbol	Offset address	Bit width	Reset value
CRC control register	CRC_CR	0x00	32	0x0000_001C
CRC result register	CRC_RESLT	0x04	32	0x0000_0000
CRC flag register	CRC_FLG	0x0C	32	0x0000_0000
CRC data register	CRC_DAT	0x80~0xFC	32	0xFFFF_FFFF

34.4.1 Control Register (CRC _ CR)

Reset value: 0x001C

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

Bit	Marking	Place name	Function	Read and write
b30~b5	Reserved	-	Read as "0", write as "0"	R
b4	XOROUT	Invert the result and output	0: The result is not inverted and output 1: Invert the result and output	R/W
b3	REFOUT	The result is output with all digits reversed	0: All digits of the result are output without inversion 1: All digits of the result are reversed and output	R/W
b2	REFIN	The number of digits in the input data byte is reversed	0: The number of digits in the input data byte is not reversed 1: The number of digits in the input data byte is reversed	R/W
b1	CR	Operation digit selection	0:CRC16 1:CRC32	R/W
b0	Reserved	-	Reserved bit	R/W

34.4.2 Results Register (CRC _ RESLT)

Reset value: 0x0000

Select CRC16: (CRC_SEL=0)

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	CRCFLAG_16
CRC_REG[15:0]																

Bit	Marking	Place name	Function	Read and write
b31~b17	Reserved	-	Read as "0", write as "0"	R
b16	CRCFLAG_16	CRC16 check result	0: CRC16 operation verification error 1: CRC16 operation check is correct	R
b15~b0	CRC_REG[15:0]	Result bit	This 16-bit register is used to update and save each CRC16 calculation result; after the operation, read this register to get the 16-bit CRC encoding result	R/W

Select CRC32: (CRC_SEL=1)

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

Bit	Marking	Place name	Function	Read and write
b31~b0	CRC_REG[31:0]	Result bit	This 32-bit register is used to update and save each CRC32 calculation result; after the operation, read this register to get the 32-bit CRC encoding result	R/W

34.4.3 Flag Register (CRC_FLG)

Reset value: 0x0000

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

Bit	Marking	Place name	Function	Read and write
b31~b1	Reserved	-	Read as "0", write as "0"	R
b0	CRC_FLAG	CRC32 check result	CRC verification result flag; 0: current verification error; 1: current verification is correct	R

34.4.4 Data Register (CRC_DAT)

Reset value: 0x0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CRC_DAT[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
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 the data that needs to be calculated; the address of this register is a range (0x80~0xFC), corresponding to 32 addresses (each address corresponds to 32-bit data). Any one of the 32 addresses is operated, it will be regarded as an operation on this register.	W

35 SDIO Controller (SDIOC)

35.1 Introduction

SDIOC provides an SD host interface and an MMC host interface for communicating with SD cards supporting the SD2.0 protocol, SDIO devices and MMC devices supporting the eMMC4.2 protocol.

The characteristics of SDIOC are as follows:

- Support SDSC, SDHC, SDXC format SD card and SDIO device
- Support one-line (1bit) and four-wire (4bit) SD bus
- Support one-line (1bit), four-wire (4bit) and eight-wire (8bit) MMC bus
- SD clock up to 50MHz
- With card identification and hardware write protection function

This product has 2 SDIO controllers, which can communicate with 2 SD/MMC/SDIO devices at the same time.

35.2 Functional Description

35.2.1 Port Assignment

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 signal
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

According to SD2.0 protocol and eMMC4.51 protocol, SD clock output signal (SDIOx_CK(x=1~2)) is used to output SD clock when SDIOC communicates with SD/MMC device; SD command and response signal (SDIOx_CMD(x=1~2)) is used to output SD/MMC command 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 device Sending and receiving 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 line (4bit) communication, SDIOx_Dy(x=1~2) (y=0~3) is valid, SDIOx_Dy(x=1~2) (y=4~7) keeps high level, adopts eight-wire (8bit)

communication, SDIOx_Dy(x=1~2) (y=0~7) is valid, eight-wire (8bit) communication is limited to MMC device communication.

35.2.2 Basic Access Method

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, writing the command register must be done 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 index), type, response (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 has been received, whether there is an error message, etc. When a command is executed, the user can get the response of the command by reading the response register (RESP0~7).

35.2.3 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 includes sending data, the user will write the data into the data buffer registers (BUF0, BUF1) sequentially, and these data will be buffered in the FIFO of SDIOC first, when the amount 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 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 Registers (BUF0, BUF1) get data. For the data format, please refer to the SD and MMC protocols.

35.2.4 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 frequency division of the action clock (EXCLK) 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 value should be 50MHz in the data transfer mode, so the user needs to set it according to the frequency of EXCLK Reasonable frequency division coefficient.

35.2.5 Interrupts and DMA Start Requests

35.2.5.1 SD Interrupt

SDIOC provides two types of interrupts, normal interrupt and error interrupt. Common interrupts are interrupts that occur when various events occur during the communication between SDIOC and SD/MMC card. Error interrupts are interrupts that are generated when various errors occur during SDIOC operation communication. Ordinary interrupts and error interrupts have their own interrupt status registers, interrupt status enable registers, and interrupt signal enable registers, respectively. 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.

35.2.5.2 SDIO Interrupt

The SDIO device can send a card interrupt (card interrupt) request to the host when the transmission is idle. When using SDIO interrupt, you need to enable the card interrupt (CINTEN) of the interrupt status enable register (NORINTSTEN). If you need to apply for an interrupt to the CPU, you also need to enable the card interrupt (CINTEN) of the interrupt signal enable register (NORINTSGEN). The SDIO device requests an interrupt from the host by pulling the SDIOx_D1 ($x=1\sim 2$) data line low. SDIOC detects that SDIOx_D1 ($x=1\sim 2$) is pulled low, sets the card interrupt (CINT) of the interrupt status register (NORINTST), and applies for an interrupt to the CPU according to the setting. The SDIO device will release SDIOx_D1 ($x=1\sim 2$) after judging that the interrupt processing is completed.

35.2.5.3 DMA Request

SDIOC can read and write data in communication through DMA. When using DMA to transfer and write data to the device, set the start source of a 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 it is a fixed address. When the SDIOC sends the write data command, if the data FIFO is empty at this time, a write request signal is sent to the DMA, and the DMA is started to write data to the data buffer register (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, SDIOC will send the data through SDIOx_Dy($x=1\sim 2$) ($y=0\sim 7$). If it is a multi-block write command, SDIOC will continue to send a start request to the DMA to write data at the same time. When using DMA to transmit read data from the device, set the start source of another channel of DMA to the read request of SDIOC, and then set the transmission source address of DMA to the data buffer register (BUF0, BUF1) and it is a fixed address. After SDIOC sends the read data command, the device will send data through SDIOx_Dy ($x=1\sim 2$) ($y=0\sim 7$), and the data FIFO of SDIOC starts to receive data. When

the data in FIFO reaches the data length register (BLKSIZE) When the set 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). If it is a multi-block read command, SDIOC will continue to read the next data block from the device at the same time.

35.2.5.4 Card Insertion (insert) and Removal (removal)

The insertion (insert) and removal (removal) of SD/MMC/SDIO devices are identified by the SDIO_{x_CD} ($x=1\sim 2$) signal line. 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 a resistor. When a device is inserted, the SDIO_{x_CD} ($x=1\sim 2$) signal will be pulled low, and will become high again after the device is removed. SDIOC judges whether there is a device by the level of SDIO_{x_CD} ($x=1\sim 2$). Users can judge whether there is a device plugged in by reading the CDPL bit of 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 (NORINTSEN) and the interrupt signal enable register (NORINTSGEN).

35.2.6 Host and Device Initialization

35.2.6.1 Host Initialization

SDIOC needs to be initialized first when using it. SDIOC initialization steps are as follows:

1. Read the host status register (PSTAT), query clock status and device insertion status
2. Configure the power control register (PWRCON), enable SDIOC
3. Configure the clock control register (CLKCON) to make SDIOC output the SD clock, and configure the frequency division of the SD clock according to the frequency of EXCLK to ensure that the frequency of the SD clock in the card identification mode does not exceed 400KHz
4. Configure the host control register (HOSTCON), select one-line (1bit) mode and disable 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 normal and error interrupt status enable registers, interrupt signal enable registers to enable SDIOC to interrupt when needed and set the flag position to

35.2.6.2 SD Card Initialization

After completing the SDIOC initialization configuration, if the SD card is connected. It needs to be initialized according to the SD protocol, and the initialization sequence is as follows:

1. Card reset, send reset command CMD0 to the device, CMD0 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 information (response)
3. Send the initialization command ACMD41 (send CMD55 first, then send CMD41), judge whether the device is initialized 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.

35.2.6.3 MMC Card Initialization

After completing the SDIOC initialization configuration, if the MMC card is connected. It needs to be initialized according to the MMC protocol, and the initialization sequence is as follows:

1. Card reset, send reset command CMD0 to the device, CMD0 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 CMD1 to the device and wait for the response message (response), judge whether the device is initialized according to the response message, 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.

35.2.6.4 SDIO Initialization

After completing the SDIOC initialization configuration, if an SDIO device is connected. It needs to be initialized according to the SDIO protocol, and 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 information (response)
3. Send CMD3 to get the RCA information of the device, and the initialization is complete

35.2.7 SD/MMC Single Block (Single Block) Read and Write

After the SD/MMC card enters the data transmission mode, 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 Make SDIOC OUTPUT The SD CLOCK, and Configure the SD CLOCK FREQUENCY DIVISION THE FREQUENCY OF EXCLK S. O 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 the data transmission state.
 3. If necessary, for the SD card, configure the host control register (HOSTCON) to set the host bus width, and send ACMD6 to set the SD bus width (1bit or 4bit), and for the MMC card, 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 the SD card, you can configure the data length register (BLKSIZE) to set the data block length, and set the size (bytes) of the data block through the CMD16 command, and the configuration range is 1~512 bytes. For SDHC/SDXC and MMC cards, the data block size is fixed at 512 bytes.
 5. If necessary, for the SD card, 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 (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), and set the transfer mode to write, single block transfer. Send a single block write command CMD24. If you use the CPU 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). If you use 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 single block read command CMD17, SDIOC will receive data through 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 register (BUF0, BUF1), if using DMA to read data, wait for the DMA transfer to complete, after the reading 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.

35.2.8 SD/MMC Multi-block (Multi-block) Read and Write

The order of reading and writing multiple data blocks (block) and single data block (block) is as follows:

1. Configure The CLOCK Control Register (CLKCON) to Make SDIOC OUTPUT The SD CLOCK, and Configure the SD CLOCK FREQUENCY DIVISION THE FREQUENCY OF EXCLK S. O 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 the data transmission state.
3. If necessary, for the SD card, configure the host control register (HOSTCON) to set the host bus width, and send ACMD6 to set the SD bus width (1bit or 4bit), and for the MMC card, 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 the SD card, you can configure the data length register (BLKSIZE) to set the data block length, and set the size (bytes) of the data block through the CMD16 command, and the configuration range is 1~512 bytes. For SDHC/SDXC and MMC cards, the data block size is fixed at 512 bytes.
5. Configure the transfer mode register (TRANSMODE), set the transfer mode (read/write), and multi-block transfer. If the data block count is enabled, you need to set the number of data blocks to be transmitted in the data block count register (BLKCNT). If you do not set the data block count register (BLKCNT), you cannot start multi-block transmission. If the data block counting enable is disabled, there is no need to set the data block count register (BLKCNT). At this time, an unlimited number of multi-block transmissions can be performed. When the host decides to end the transmission, it needs to be completed after the last data block transmission. Send a CMD12 to the device to inform the device that the data transmission is over.
6. If necessary, for the SD card, 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 (fpp<=50MHz), switch After success, SDIOx_CK (x=1~2) can be set up to 50MHz
7. When writing data, send the multi-block write command CMD25. If you use the CPU to write data, after confirming that the BWR bit of the interrupt status register (NORINTST) is 1, write data to the data buffer register (BUFO, BUF1). During the sending process, BWR will Keep 0, reset to 1 after sending, 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 in the transmission process, the corresponding error flag will be set and an interrupt request will be generated. After all the data is sent, if it is set to send CMD12 automatically, SDIOC will automatically send a CMD12 to end the transmission, otherwise it is necessary to manually send CMD12 to the device to inform the device that the data transmission is over.

8. When reading data, send the multi-block read command CMD18, and 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 register (BUF0, BUF1). The data of the second block can be read at this time. If you use DMA 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 transmission process, the corresponding error flag will be set and an interrupt will be generated. ask. After all the data is received, if the automatic sending of CMD12 is set, the SDIOC will automatically send a CMD12 to end the transmission, otherwise it is necessary to manually send a CMD12 to the device to inform the device that the data transmission is over.

35.2.9 Transfer Abort, Suspend and Resume

When performing multi-block transfer, it can be terminated (abort) or suspended (suspend) by software control. The abort operation can be performed regardless of whether or not the number of transfer data blocks is set. Termination operations are divided into asynchronous termination and synchronous termination. The asynchronous termination operation needs to send CMD12 to terminate the transmission by writing the command register (CMD) while the transmission is in progress, and the transmission will be terminated immediately at this time. 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 transmission stops at the data block interval by setting the data block gap register (BLKGAP) during the transmission. After the setting is completed, the transmission will stop after the current data block transmission ends. At this time, CMD12 needs to be sent to end the transmission.

When executing the suspend operation, first set the SABGR bit of the data block interval register (BLKGAP) to stop the transmission at the data block interval, and enable the read wait (read wait) function. After the setting is complete, the transmission will stop transmission after the current data block transmission ends. At this time, send CMD52 by writing the command register (CMD) to suspend the transmission. After sending the suspend command, you need 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 is suspended, and the SD bus turns to an idle state. After the transfer is suspended, the host can perform operations on other functions of SDIO. But if you want to resume (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 data block interval register (BLKGAP) to stop the transmission at the data block interval, and then send CMD52 to resume transmission by writing the 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 be transmitted after the recovery is executed. If DF=0, it means that there is no data to be transmitted. At this time The data line should be reset by writing to the software reset register (SFTRST).

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, and only the read wait function can be used.

35.2.10 Read Wait

Read wait (read wait) allows the host to insert CMD52 during continuous data transfers to access other functions of the SDIO device. When performing read wait (read wait), firstly, the transmission is stopped at the data block interval by setting the SABGR bit of the data block interval register (BLKGAP), and the read wait (read wait) function is enabled. After the setting is completed, the host will pull the SDIOx_D2 (x=1~2) data line low after the current data block transmission is completed. The transmission will be suspended at this time, and the host can insert the CMD52 at this time to access other functions that do not require data transmission. When it is necessary to end the read wait (read wait) function, set the CR of the data block gap register (BLKGAP) and clear SABGR to resume transmission.

Note:

- Read wait (read wait) requires support from SDIO devices and combo card devices, and needs to be performed under four-wire bus transmission.

35.2.11 Wake Up

When not working for a long time, the system can be transferred to a low power consumption state to reduce power consumption. In the low power consumption state, the system can be woken up by SD/MMC/SDIO device insertion (insert)/removal (removal) and card interrupt (card interrupt) to continue working. When using the wake-up function, you need to enable the corresponding enable bit 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 the SDIOx_CD(x=1~2)/ SDIOx_D1(x=1~2) port. After the configuration is complete, the system can be transferred to a low power consumption mode. When insert (insert)/removal (removal) and card interrupt (card interrupt), the wake-up function of the SDIOx_CD(x=1~2)/ SDIOx_D1(x=1~2) port will wake up the system and insert (insert)/removal (removal) and card interrupt (card interrupt) interrupts.

Use an insert to wake up a stop mode process:

1. Configure the SDIOC port, select SDIO1_CD as PA10
2. Configure the PCR register of PA10, and select IRQ10 to be valid
3. Configure the PWRC3 register of the power control module to make the CPU enter the stop mode after executing the WFI command
4. Configure the interrupt control module WUPEN register to enable the wake-up function of IRQ10
5. Configure the EIRQCR10 register of the interrupt control module and select the falling edge trigger
6. Configure PWRCON 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 status in the interrupt status register (NORINTST), exit the interrupt subroutine, and perform subsequent operations

Use the insert (insert) to wake up the power down mode (power down mode) process:

1. Configure the SDIOC port, select SDIO1_CD as PA10
2. Configure the PCR register of PA10, and select IRQ10 to be valid
3. Configure the PWRC3 and PWRC0 registers of the power control module to enable the CPU to 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 interrupt control module WUPEN register 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 PWRCON 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

35.3 Register Description

The SDIOC module is designed according to the SD Host Controller Standard Specification, so the registers are the same as the standard description, and the unused addresses and bits are changed to reserved bits (Reserved).

The following table is the register list of the SDIOC module.

BASE ADDR: 0x4006FC00 (SDIOC1) , 0x40070000 (SDIOC2)

Register name	Symbol	Offset address	Bit width	Reset value
Data block length	BLKSIZE	0x04	16	0x0000h
Block count	BLKCNT	0x06	16	0x0000h
Parameter 0	ARG0	0x08	16	0x0000h
Parameter 1	ARG1	0x0A	16	0x0000h
Transfer mode	TRANSMODE	0x0C	16	0x0000h
Order	CMD	0x0E	16	0x0000h
Answer 0	RESP0	0x10	16	0x0000h
Answer 1	RESP1	0x12	16	0x0000h
Answer 2	RESP2	0x14	16	0x0000h
Answer 3	RESP3	0x16	16	0x0000h
Answer 4	RESP4	0x18	16	0x0000h
Answer 5	RESP5	0x1A	16	0x0000h
Answer 6	RESP6	0x1C	16	0x0000h
Answer 7	RESP7	0x1E	16	0x0000h
Data buffer 0	BUF0	0x20	16	0x0000h
Data buffer 1	BUF1	0x22	16	0x0000h
Host Status	PSTAT	0x24	32	0x00000000h
Host control	HOSTCON	0x28	8	0x00h
Power control	PWRCON	0x29	8	0x00h
Data block interval control	BLKGPCON	0x2A	8	0x00h
Clock control	CLKCON	0x2C	16	0x0002h
Timeout control	TOUTCON	0x2E	8	0x00h
Software reset	SFTRST	0x2F	8	0x00h
Normal interrupt status	NORINTST	0x30	16	0x0000h
Error interrupt status	ERRINTST	0x32	16	0x0000h
Normal interrupt status enable	NORINTSTEN	0x34	16	0x0000h
Error interrupt status enable	ERRINTSTEN	0x36	16	0x0000h
Normal interrupt signal enable	NORINTSGEN	0x38	16	0x0000h
Error interrupt signal enable	ERRINTSGEN	0x3A	16	0x0000h
Autocommand error status	ATCERRST	0x3C	16	0x0000h

Register name	Symbol	Offset address	Bit width	Reset value
Force autocommand error state control	FEA	0x50	16	0x0000h
Force Error State Control	FEE	0x52	16	0x0000h

In addition, SDIOC1 and SDIOC2 share an MMC mode enable register for the controller to switch between SD and MMC modes

Register name	Symbol	Address	Bit width	Reset value
MMC Mode Enable Register	MMCER	0x40055404	32	0x00000000h

35.3.1 Data Block Size 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]													

Bit	Marking	Place name	Function	Read and write
b15~b12	Reserved	-	Read 0 when reading, write 0 when writing	R
b11~0	TBS	Data block length	Set the length of the transmitted data block (Transfer Block Size), the length of the transmitted data block is in bytes, and the setting range is 1~512	R/W

35.3.2 Data 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]															

Bit	Marking	Place name	Function	Read and write
b15~b0	BC	Block count	Set the number of transmission data blocks (Block count), set this register 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

35.3.3 Parameter Register 0 (ARG0)

Offset address: 0x08

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ARG0[15:0]															

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 parameters	R/W

35.3.4 Parameter Register 1 (ARG1)

Offset address: 0x0A

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ARG1[15:0]															

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 parameters	R/W

35.3.5 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
b5	MULB	Multi-block	0: The current transfer is a single block transfer (single block) 1: The current transmission is multi-block transmission (multi block)	R/W
b4	DDIR	Data transmission direction	0: Write operation (host sends data) 1: Read operation (the host receives data)	R/W
b3~b2	ATCEN	Autocommand enable	00: Do not send autocommands 01: Automatically send CMD12 after multi-block transmission 10: Setting prohibited 11: Setting prohibited	R/W
b1	BCE	Block count enable	0: Disable data block count enable 1: Allow data block count enable	R/W
b0	Reserved	-	Read 0 when reading, write 0 when writing	R

35.3.6 Command Register (CMD)

Offset address: 0x0E

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved						IDX[5:0]		TYP[1:0]	DAT	ICE	CCE	Rsvd			RESTYP[1:0]

Bit	Marking	Place name	Function	Read and write
b15~b14	Reserved	-	Read 0 when reading, write 0 when writing	R
b13~b8	IDX	Command number	The number of the command sent	R/W
b7~b6	TYP	Command type	00: common command 01: suspend command 10: resume (resume) command 11: Terminate (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 (response) 1: Check the command number in the response (response)	R/W
b3	CCE	CRC check	0: Do not check the CRC check code in the response (response) 1: Check the CRC check code in the response (response)	R/W
b2	Reserved	-	Read 0 when reading, write 0 when writing	R
b1~b0	RESTYP	Response type	00: No response to this command 01: The command has a response length of 136bit 10: The command has a response with a length of 48 bits 11: The command has a length of 48 bits and a response with a busy status	R/W

35.3.7 Response Register 0 (RESP0)

Offset address: 0x10

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP0[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP0[15:0]	Response message	15~0 digits of response information	R

35.3.8 Response Register 1 (RESP1)

Offset address: 0x12

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP1[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP1[15:0]	Response message	31~16 digits of response information	R

35.3.9 Response Register 2 (RESP2)

Offset address: 0x14

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP2[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP2[15:0]	Response message	47~32 digits of response information	R

35.3.10 Response Register 3 (RESP3)

Offset address: 0x16

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP3[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP3[15:0]	Response message	63~48 digits of response information	R

35.3.11 Response Register 4 (RESP4)

Offset address: 0x18

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP4[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP4[15:0]	Response message	79~64 digits of response information	R

35.3.12 Response Register 5 (RESP5)

Offset address: 0x1A

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP5[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP5[15:0]	Response message	95~80 digits of response information	R

35.3.13 Response Register 6 (RESP6)

Offset address: 0x1C

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP6[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP6[15:0]	Response message	111~96 digits of response information	R

35.3.14 Response Register 7 (RESP7)

Offset address: 0x1E

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP7[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP7[15:0]	Response message	127~112 digits of response information	R

35.3.15 Data Buffer Register 0 (BUF0)

Offset address: 0x20

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
BUF0[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	BUF0	data buffer	Write send data and read receive data, this register is the lower 16 bits of data	R/W

35.3.16 Data Buffer Register 1 (BUF1)

Offset address: 0x22

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
BUF1[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	BUF1	data buffer	Write send data and read receive data, this register is the upper 16 bits of data	R/W

35.3.17 Host Status Register (PSTAT)

Offset address: 0x24

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				CMDL	DATL[3:0]				WPL	CDL	CSS	CIN			

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		BRE	BWE	RTA	WTA	Reserved				DA	CID	CIC			

Bit	Marking	Place name	Function	Read and write
b31~b25	Reserved	-	Read 0 when reading, write 0 when writing	R
B24	CMDL	Command line status	Status of the command line (SDIOx_CMD(x=1~2))	R
B23~b20	DATL	Data line status	The state of the data line (SDIOx_Dy(x=1~2) (y=0~3))	R
B19	WPL	Write Protect Line Status	State of write protection line (SDIOx_WP(x=1~2))	R
B18	CDL	Card Identification Line Status	The state of the card identification line (SDIOx_CD(x=1~2))	R
B17	CSS	Device steady state	0: The state of the card identification line is unstable 1: The status of the card identification line is stable, the device is inserted or not	R
B16	CIN	Device plugged in	0: No device inserted 1: There is a device plugged in	R
B15~b12	Reserved	-	Read 0 when reading, write 0 when writing	R
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: The data buffer cannot write data	R
B9	RTA	Read operation status	0: No read operation in progress 1: There is a read operation in progress	R
B8	WTA	Write status	0: no write operation in progress 1: There is a write operation in progress	R
B7~b3	Reserved	-	Read 0 when reading, write 0 when writing	R
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 sending commands with data 1: Prohibit sending commands with data	R
B0	CIC	command suppression	0: Allow sending commands 1: Disable sending commands	R

35.3.18 Host Control Register (HOSTCON)

Offset address: 0x28

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
CDSS	CDTL	EXDW		Reserved	HSEN	DW	Rsvd

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: The card identification test signal is low level (there is a device inserted) 1: The card identification test signal is high level (no equipment inserted)	R/W
b5	EXDW	Extended data 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
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

35.3.19 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
b0	PWON	SDIOC enable	0: Disable SDIOC 1: Enable SDIOC	R/W

35.3.20 Data Block Gap Control Register (BLKGPCON)

Offset address: 0x2A

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
		Reserved		IABG	RWC	CR	SABGR

Bit	Marking	Place name	Function	Read and write
b7~b4	Reserved	-	Read 0 when reading, write 0 when writing	R
b3	IABG	Block Gap Interrupt Control	0: Receive SDIO device interrupt (card interrupt) during data block gap closing 1: Receive SDIO device interrupt (card interrupt) during data block gap opening	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: Release a transfer that was stopped by setting the SABGR bit	R/W
b0	SABGR	Data block gap stop transmission	0: Stop transmission when there is no data block gap 1: Stop transmission when there is a data block gap	R/W

35.3.21 Clock Control Register (CLKCON)

Offset address: 0x2C

Reset value: 0x0002

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FS[7:0]				Reserved				CE	Rsvd		ICE				

Bit	Marking	Place name	Function	Read and write
B15~b8	FS	Frequency selection	SDIO _x _CK (x=1~2) clock frequency division selection, the reference clock is EXCLK 0x80: 256 frequency division of EXCLK 0x40: 128 frequency division of EXCLK 0x20: 64 frequency division of EXCLK 0x10: 32 frequency division of EXCLK 0x08: 16 frequency division of EXCLK 0x04: 8 frequency division of EXCLK 0x02: 4 frequency division of EXCLK 0x01: 2 frequency division of EXCLK 0x00: EXCLK Other: Prohibitions	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 is on 1: SDIOC action clock off	R/W

35.3.22 Timeout Control Register (TOUTCON)

Offset address: 0x2E

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
Reserved						DTO[3:0]	

Bit	Marking	Place name	Function	Read and write
b7~b4	Reserved	-	Read 0 when reading, write 0 when writing	R
b3~b0	DTO	data timeout	Set the data line SDIO _x _Dy (x=1~2) (y=0~7) overtime judgment time, the unit is the clock cycle of EXCLK 0000: EXCLK×2 ¹³ 0001: EXCLK×2 ¹⁴ 1110: EXCLK×2 ²⁷ 1111: Setting prohibited	R/W

35.3.23 Software Reset Register (SFTRST)

Offset address: 0x2F

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
Reserved						RSTD	RSTC

Bit	Marking	Place name	Function	Read and write
b7~b3	Reserved	-	Read 0 when reading, write 0 when writing	R
b2	RSTD	Data reset	Resets all data-related registers, including the following register bits: BUF0, 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: Execute 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: Execute reset	R/W
b0	RSTA	Reset all	Reset all SDIOC registers except card identification function 0: Normal work 1: Execute reset	R/W

35.3.24 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
b8	CINT	Card interrupt	Set when the SDIO device issues a card interrupt application, and reset after the SDIO device cancels the application	R
b7	CRM	Card removal	Set when the device is removed (removal), write 1 to reset	R/W
b6	CIST	Card insertion	Set when the device is inserted (insert), write 1 to reset	R/W
b5	BRR	Buffer readable	Set when the data in the buffer can be read (PSTAT.BRE=1), write 1 to reset	R/W
b4	BWR	Buffer writable	Set when the buffer can write data (PSTAT.BWE=1), write 1 to reset	R/W
b3	Reserved	-	Read 0 when reading, write 0 when writing	R
b2	BGE	Data block gap stop transmission	Set when transfer stops between data blocks, write 1 to reset	R/W
b1	TC	Transmission completion	Set when read and write transfer is complete, write 1 to reset	R/W
b0	CC	command completed	No response command is set after the command is sent and the response of the response command is received, write 1 to reset	R/W

35.3.25 Error Interrupt Status Register (ERRINTST)

Offset address: 0x32

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved							ACE	Rsvd	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R
b8	ACE	Auto send command error	Set when an error occurs in the automatic sending command (auto CMD), the type of error can be queried in the automatic command error register (ATCERRST), write 1 to reset	R/W
b7	Reserved	-	Read 0 when reading, write 0 when writing	R
b6	DEBE	Data stop bit error	Set when the data line detects a low level in the stop bit, write 1 to reset	R/W
b5	DCE	Data CRC check error	Set when a CRC check error occurs on the data line, write 1 to reset	R/W
b4	DTOE	data timeout error	Set when data timeout occurs, write 1 to reset, data timeout time is set by timeout control register (TOUTCON)	R/W
b3	CIE	Wrong command number	Set when the command number contained in the received response is wrong, write 1 to reset	R/W
b2	CEBE	Command stop bit error	Set when the command line detects a low level in the stop bit, write 1 to reset	R/W
b1	CCE	Command CRC check error	Set when a CRC check error occurs on the command line, write 1 to reset	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

35.3.26 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

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R
b8	CINTEN	Card Interrupt Status Enable	0: Disable NORINTST.CINT setting 1: Allow NORINTST.CINT to be set	R
b7	CRMEN	Card Removed Status Enable	0: Disable NORINTST.CRM setting 1: Allow NORINTST.CRM to be set	R/W
b6	CISTEN	Card Insertion Status Enable	0: Disable NORINTST.CIST setting 1: Allow NORINTST.CIST to be set	R/W
b5	BRREN	Buffer Read Status Enable	0: Disable NORINTST.BRR setting 1: Allow NORINTST.BRR to be set	R/W
b4	BWREN	Buffer Writable Status Enable	0: Disable NORINTST.BWR setting 1: Allow NORINTST.BWR to be set	R/W
b3	Reserved	-	Read 0 when reading, write 0 when writing	R
b2	BGEEN	Data block gap stop transmission status enable	0: Disable NORINTST.BGE setting 1: Allow NORINTST.BGE to be set	R/W
b1	TCEN	Transfer Complete Status Enable	0: Disable NORINTST.TC setting 1: Allow NORINTST.TC to be set	R/W
b0	CCEN	command completion status enable	0: Disable NORINTST.CC setting 1: Allow NORINTST.CC to be set	R/W

35.3.27 Error Interrupt Status Enable Register (ERRINTSTEN)

Offset address: 0x36

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
							Reserved		ACEEN	Rsvd	DEBEEN	DCEEN	DTOEEN	CIEEN	CEBEE	CCEEN	CTOEEN
<hr/>																	
Bit	Marking	Place name	Function										Read and write				
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing										R				
b8	ACEEN	Automatically send command error status enable	0: disable ERRINTST.ACE setting 1: Allow ERRINTST.ACE to be set										R/W				
b7	Reserved	-	Read 0 when reading, write 0 when writing										R				
b6	DEBEEN	Data Stop Bit Error Status Enable	0: Disable ERRINTST.DEBE setting 1: Allow ERRINTST.DEBE to be set										R/W				
b5	DCEEN	Data CRC check error status enable	0: Disable ERRINTST.DCE setting 1: Allow ERRINTST.DCE to be set										R/W				
b4	DTOEEN	Data Timeout Error Status Enable	0: Disable ERRINTST.DTOE setting 1: Allow ERRINTST.DTOE to be set										R/W				
b3	CIEEN	command number error status enable	0: Disable ERRINTST.CIE setting 1: Allow ERRINTST.CIE to be set										R/W				
b2	CEBEE	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 setting 1: Allow ERRINTST.CCE to be set										R/W				
b0	CTOEEN	Command Timeout Error Status Enable	0: Disable ERRINTST.CTOE setting 1: Allow ERRINTST.CTOE to be set										R/W				

35.3.28 Normal Interrupt Signal Enable Register (NORINTSGEN)

Offset address: 0x38

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
							Reserved	CINTSEN	CRMSEN	CISTSEN	BRRSEN	BWRSEN	Rsvd	BGESEN	TCESN	CCSEN

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R
b8	CINTSEN	Card interrupt signal enable	0: disable NORINTST.CINT application interrupt 1: Allow NORINTST.CINT to apply for interrupt	R
b7	CRMSEN	Card removal signal enable	0: disable NORINTST.CRM application interruption 1: Allow NORINTST.CRM to apply for interruption	R/W
b6	CISTSEN	Card insertion signal enable	0: disable NORINTST.CIST application interrupt 1: Allow NORINTST.CIST to apply for interruption	R/W
b5	BRRSEN	Buffer readable signal enable	0: disable NORINTST.BRR application interrupt 1: Allow NORINTST.BRR to apply for interrupt	R/W
b4	BWRSEN	Buffer Writable Signal Enable	0: disable NORINTST.BWR application interrupt 1: Allow NORINTST.BWR to apply for interrupt	R/W
b3	Reserved	-	Read 0 when reading, write 0 when writing	R
b2	BGESEN	Data block gap stop transmission signal enable	0: disable NORINTST.BGE application interrupt 1: Allow NORINTST.BGE to apply for interruption	R/W
b1	TCSEN	Transfer complete signal enable	0: disable NORINTST.TC application interrupt 1: Allow NORINTST.TC to apply for interruption	R/W
b0	CCSEN	command complete signal enable	0: Disable NORINTST.CC application interrupt 1: Allow NORINTST.CC to apply for interruption	R/W

35.3.29 Error Interrupt Signal Enable Register (ERRINTSGEN)

Offset address: 0x3A

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
							ACESEN	Rsvd	DEBESEN	DCESEN	DTOESEN	CIESEN	CEBESEN	CCESEN	CTOESEN

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R
b8	ACESEN	Automatically send command error signal enable	0: disable ERRINTST.ACE application interrupt 1: Allow ERRINTST.ACE to apply for interrupt	R/W
b7	Reserved	-	Read 0 when reading, write 0 when writing	R
b6	DEBESEN	Data stop bit error signal enable	0: disable ERRINTST.DEBE application interruption 1: Allow ERRINTST.DEBE to apply for interruption	R/W
b5	DCESEN	Data CRC check error signal enable	0: disable ERRINTST.DCE application interrupt 1: Allow ERRINTST.DCE to apply for interrupt	R/W
b4	DTOESEN	Data timeout error signal enable	0: Disable ERRINTST.DTOE application interrupt 1: Allow ERRINTST.DTOE to apply for interrupt	R/W
b3	CIESEN	Command number error signal enable	0: disable ERRINTST.CIE application interrupt 1: Allow ERRINTST.CIE to apply for interrupt	R/W
b2	CEBESEN	Command stop bit error signal enable	0: Shield ERRINTST.CEBE application interrupt 1: Allow ERRINTST.CEBE to apply for interrupt	R/W
b1	CCESEN	Command CRC check error signal enable	0: disable ERRINTST.CCE application interrupt 1: Allow ERRINTST.CCE to apply for interrupt	R/W
b0	CTOESEN	Command timeout error signal enable	0: disable ERRINTST.CTOE application interrupt 1: Allow ERRINTST.CTOE to apply for interrupt	R/W

35.3.30 Auto Command Error Status Register (ATCERRST)

Offset address: 0x3C

Reset value: 0x0000

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

Bit	Marking	Place name	Function	Read and write
b15~b8	Reserved	-	Read 0 when reading, write 0 when writing	R
b7	CMDE	Error not sent	Set when the error corresponding to bits b4~b0 of this register occurs and other commands are not sent	R
b6~b5	Reserved	-	Read 0 when reading, write 0 when writing	R
b4	IE	Wrong command number	Set when the received reply contains the wrong autocommand number	R
b3	EBE	Stop bit error	Asserted 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 SDIOx_CK ($x=1\sim 2$) cycles after the automatic command is sent	R
b0	NE	Not executed error	Set when an autocommand was not sent for other reasons	R

35.3.31 Force 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

Bit	Marking	Place name	Function	Read and write
b15~b8	Reserved	-	Read 0 when reading, write 0 when writing	R
b7	FCMDE	Force not send error	0: Nothing 1: Force an ATCERRST.CMDE error	W
b6~b5	Reserved	-	Read 0 when reading, write 0 when writing	W
b4	FIE	Mandatory command number error	0: Nothing 1: Force the ATCERRST.IE error to occur	W
b3	FEBE	Forced stop bit error	0: Nothing 1: Force an ATCERRST.EBE error	W
b2	FCE	Force data timeout error	0: Nothing 1: Force an ATCERRST.CE error	W
b1	FTOE	Force command timeout error	0: Nothing 1: Force an ATCERRST.TOE error	W
b0	FNE	force not executed error	0: Nothing 1: Force an ATCERRST.NE error	W

35.3.32 Force 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

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R
b8	FACE	Force auto send command error	0: Nothing 1: Force an ERRINTST.ACE error	W
b7	Reserved	-	Read 0 when reading, write 0 when writing	R
b6	FDEBE	Force data stop bit error	0: Nothing 1: Force an ERRINTST.DBE error	W
b5	FDCE	Mandatory data CRC check error	0: Nothing 1: Force an ERRINTST.DCE error	W
b4	FDTOE	Force data timeout error	0: Nothing 1: Force an ERRINTST.DTOE error	W
b3	FCIE	Mandatory command number error	0: Nothing 1: Force an ERRINTST.CIE error	W
b2	FCEBE	Force command stop bit error	0: Nothing 1: Force an ERRINTST.CEBE error	W
b1	FCCE	Mandatory command CRC check error	0: Nothing 1: Force an ERRINTST.CCE error	W
b0	FCTOE	Force command timeout error	0: Nothing 1: Force an ERRINTST.CTOE error	W

35.3.33 MMC Mode Enable Register (MMCER)

Address: 0x40055404

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
												SELMMC2	Reserved	SELMMC1	Reserved

Bit	Marking	Place name	Function	Read and write
b31~b4	Reserved	-	Read as "0", write as "0"	R
b3	SELMMC2	SDIOC2 MMC mode enable	0: SDIOC2 selects SD mode 1: SDIOC2 selects MMC mode	R/W
b2	Reserved	-	Read as "0", write as "0"	R
b1	SELMMC1	SDIOC1 MMC mode enable	0: SDIOC1 selects SD mode 1: SDIOC1 selects MMC mode	R/W
b0	Reserved	-	Read as "0", write as "0"	R

36 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

36.1 Introduction

The core of this MCU is Cortex™-M4F, which contains hardware for advanced debugging functions. 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. This MCU is not equipped with ETM debugging device.

Two debugging interfaces are provided:

- Serial Debugging Tracking Interface SWD
- Parallel debug trace interface JTAG

36.2DBG System Block Diagram

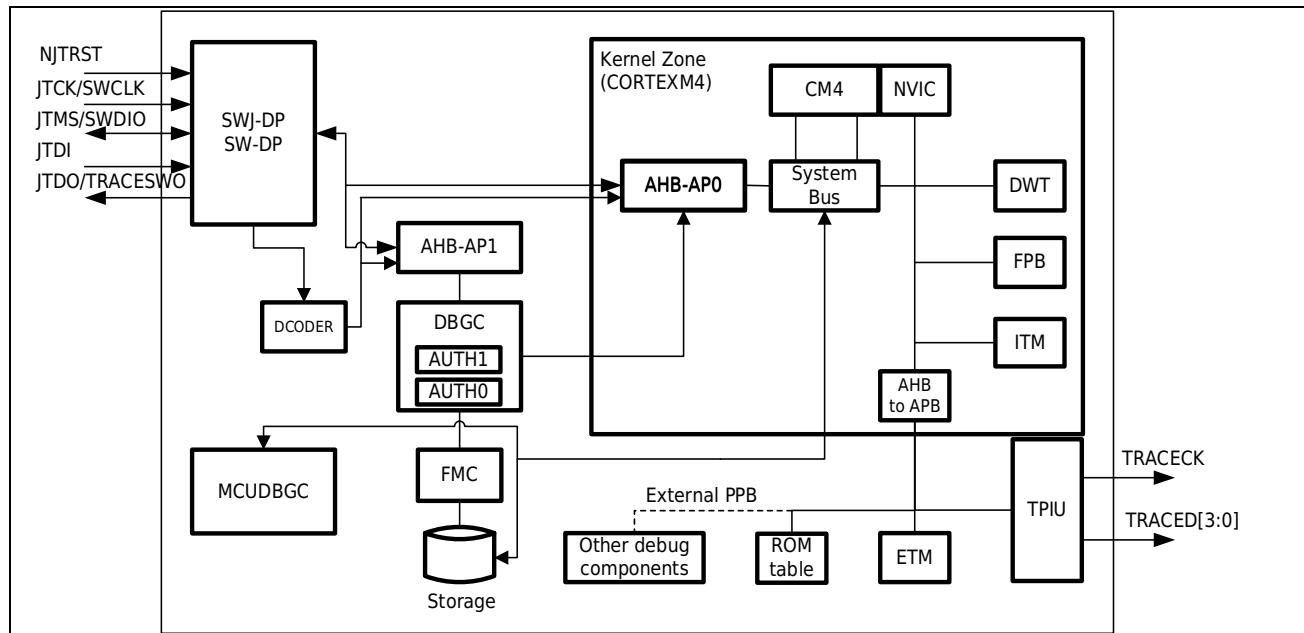


Figure 36-1 Commissioning Control System

ARM Cortex™ -M4F + cores provide integrated on-chip debugging support. It includes:

- SWJ-DP: SWD/JTAG debug port
- AHB-AP: AHB Access Port
- ITM: Instruction Tracking Unit
- FPB: Flash instruction breakpoint
- DWT: Data breakpoint trigger
- TPIU: Trace Port Unit Interface (available on larger packages 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.

36.3 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 JTAG-DP (5 pins) interface and SW-DP (2 pins) interface.

- The JTAG Debug Port (JTAG-DP) provides a 5-pin standard JTAG interface for connection to the AHP-AP port.
- The Serial Wire Debug Port (SW-DP) provides a 2-pin (clock+data) interface for 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 reuses TRACESWO and TDO. This means that asynchronous trace can only be implemented on SW-DP, not on JTAG-DP.

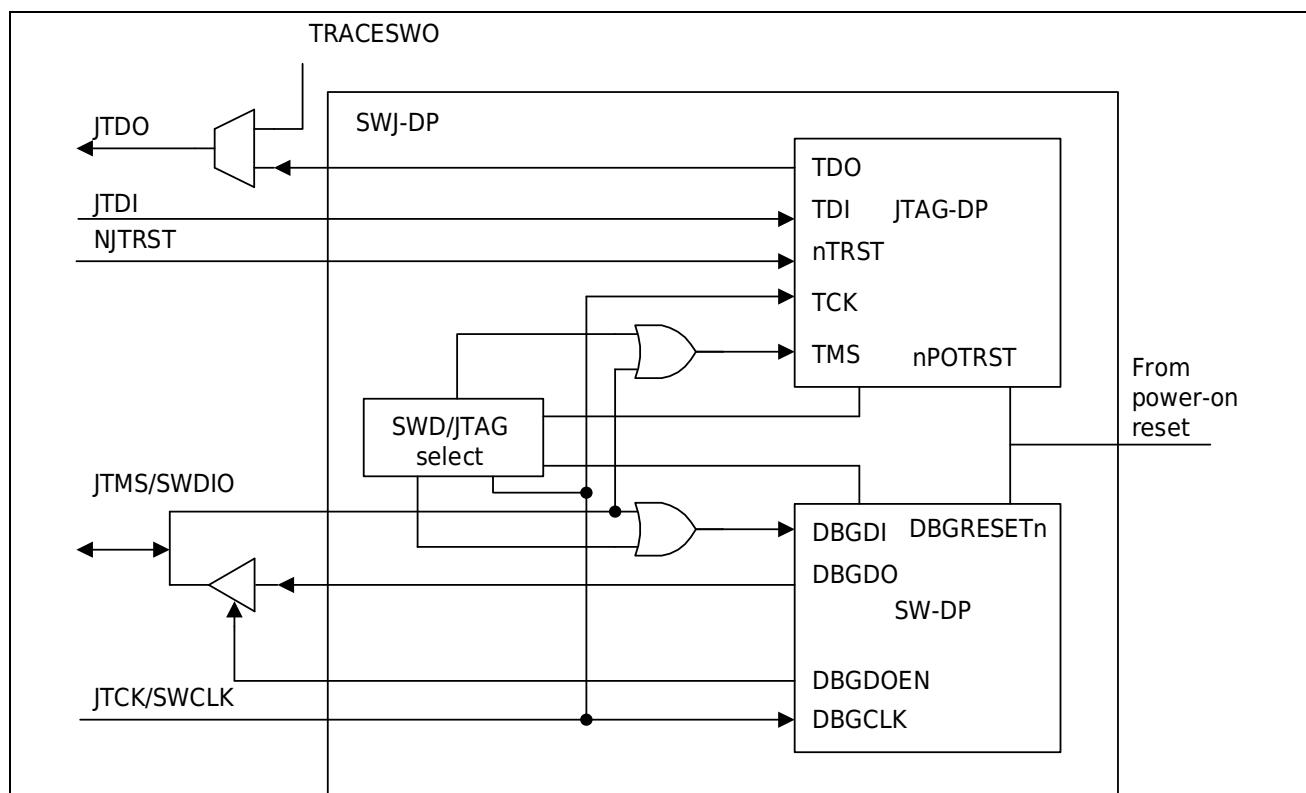


Figure 36-2 Commissioning control system

36.3.1 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) for disabling JTAG-DP and enabling SW-DP. This allows access to the SW-DP 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

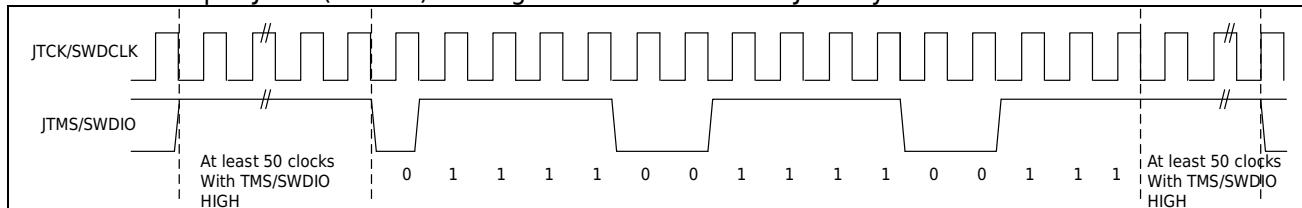


Figure 36-3 JTAG-DP to SW-DP Switching Timing

36.4 Pinout and Debug Port Pinout

Depending on the MCU package, there are different effective pin counts. Therefore, some pin-related functions may vary from package to package.

36.4.1 SWJ Debug Port Pin

5 common I/O ports of MCU can be used as SWJ-DP interface pins.

Table 36-1 SWJ Debug Port Pins

SWJ-DP Pin name	JTAG debug port		SW debug port	
	Types	Note	Types	Debug allocation
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	-	-

36.4.2 Flexible SWJ-DP pin assignment

After reset (power-up or pin reset), all 5 pins for SWJ-DP are designated as dedicated pins, available 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 SWJ-DP port pins, see: General IO Special Control Register PSPCR.

Table 36-2 Flexible SWJ-DP pin assignment

Available debug ports	Assigned SWJ IO pins				
	JTMS/ SWDIO	JTCK/ SWCLK	JTDI	JTDO	NJTRST
All SWJ (JTAG-DP+SW-DP) - reset state	✓	✓	✓	✓	✓
Disable JTAG-DP and enable SW-DP	✓	✓	Releasable	Releasable	Releasable
Disable JTGA-DP and Disable SW-DP	Releasable	Releasable	Releasable	Releasable	Releasable

36.4.3 Internal Pull-ups on JTAG Pins

According to the JTAG IEEE standard, it must be ensured that the JTAG input pins are not floating because these pins are directly connected to the MCU internals for controlling the debug functions. Special attention must also be paid to the JTCK/SWCLK pin, which is used directly for the debug control clock function. In order to avoid IO level floating, the MCU has built-internal pull-up resistors on the JTAG pins:

- NJTRST: Internal pull-up
- JTDI: Internal pull-up
- JTMS/SWDIO: internal pull-up
- JTCK/SWCLK: Internal pull-up
- JTDO: high-impedance state

When the debugger is not connected, the user software can release the JTAG IO as a normal I/O port by setting the GPIO special control register. Since the internal pull-up of the chip is a weak pull-up of <100K ohms, it is recommended to use an external pull-up of 10K ohms.

36.4.4 Use the Serial Interface and Release Unused Debug Pins for GPIO

Some GPIOs can be released when using SWD, user software must change the GPIO configuration in the GPIO control register, so that the corresponding pins can be released as GPIO.

When debugging, the host does the following:

- In system reset state, assign all SWJ pins (JTAG-DP+SW-DP).
- In system reset state, the debug host sends a JTAG sequence to switch from JTAG-DP to SW-DP.
- Still in the system reset state, the debug host sets a breakpoint at the reset address.
- The reset signal is released and the core stops at the reset address.
- From this debug port switch to SW-DP. The other JTAG pins are then reassigned as GPIOs by user software.

Note:

- For user software design, when it is necessary to release the debug pins, these pins are still input pull-ups after reset until the user software releases the pins (NJTRST, JTMS, JTDI, JTCK, and JTDO)

36.5 Register

The register is described as follows:

Base address: 0xE004_2000

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	0x00000003	CPU/Debug IDE*
TRACE port control register	MCUTRACECTL	0x0024	32	0x00000000	CPU/Debug IDE*

Note:

- Registers are located in the PPB area and can only be accessed by the CPU in privileged mode.

36.5.1 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

Bit	Marking	Place name	Function	Read and write
b31~2	Reserved	-	Read as "0", write as "0"	R/W
b1	CDBGPWRUPACK	Power-on feedback of the debugger	0: No feedback 1: Commissioning Power-on Feedback	R/W
b0	CDBGPWRUPREQ	Debugger power-on request	0: No power-on request 1: Power-on request	R/W

36.5.2 Peripheral Debugging Pause Register (MCUSTPCTL)

The peripheral module pauses control when the CPU is in the debug state.

Reset value: 0x0000_0003

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TMR A6 STP	TMR A5 STP	TMR A4 STP	TMR A3 STP	TMR A2 STP	TMR A1 STP	TMR 63 STP	TMR 62 STP	TMR 61 STP	TMR 43 STP	TMR 42 STP	TMR 41 STP	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TMR 02 STP	TMR 01 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	TMRA6STP	TimerA-6 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b30	TMRA5STP	TimerA-5 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b29	TMRA4STP	TimerA-4 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b28	TMRA3STP	TimerA-3 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b27	TMRA2STP	TimerA-2 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b26	TMRA1STP	TimerA-1 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b25	TMR63STP	Timer6-3 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b24	TMR62STP	Timer6-2 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b23	TMR61STP	Timer6-1 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b22	TMR43STP	Timer4-3 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b21	TMR42STP	Timer4-2 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b20	TMR41STP	Timer4-1 counting pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	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	TMR02STP	Timer0-2 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b14	TMR01STP	Timer0-1 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b13~6	Reserved	-	Read as "0", write as "0"	R/W
b5	PVD2STP	PVD2 interrupt/reset mask	0: PVD2 interrupt request or reset is still generated even if the core is stopped 1: Block PVD2interrupt application or reset when kernel stops	R/W
b4	PVD1STP	PVD1 interrupt/reset mask	0: PVD1 interrupt request or reset is still generated even if the core is stopped 1: Block PVD1interrupt application or reset when kernel stops	R/W
b3	PVD0STP	PVD0 interrupt/reset mask	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: RTC counter still counts even if the kernel is stopped 1: RTC counter pause count when kernel stops	R/W
b1	WDTSTP	WDT count pause signal	0: WDT counter still counts even if the kernel is stopped	R/W

			1: WDT counter pause count when kernel stops	
b0	SWDTSTP	SWDT count pause signal	0: SWDTcounter still counts even if the kernel stops 1: SWDTcounter pause count when kernel stops	R/W

36.5.3 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
-	-	-	-	-	-	-	-	-	-	-	-	-	TRAC EIOE N	TRACEMODE	

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	TRACEIOEN	TRACE pin output control	0: Synchronous tracking pin output disabled 1: Synchronous trace pin output permission	R/W
b1~b0	TRACEMODE	TRACEDATA output pin control	00: asynchronous tracking 01: synchronous tracking 1 bit TRACEDATA[0] 10: synchronous tracking 2-bit TRACEDATA[1:0] 11: synchronous tracking 4-bit TRACEDATA[3:0]	R/W

36.6SW Debug Port

36.6.1 Introduction to SW Agreement

This synchronous serial protocol uses two pins:

- SWCLK: clock from master to slave
- SWDIO: Bidirectional

LSB is ahead when transferring data.

For SWCLK and SWDIO, the line needs to be pulled up on the circuit board (recommended to use 10K ohms).

36.7TPIU (Trace Port Interface Unit)

36.7.1 Introduction

The TPIU is the bridge between the ITM and the on-chip trace data.

The output data stream is encapsulated into a trace source ID, which is then captured by the trace port analyzer (TPA).

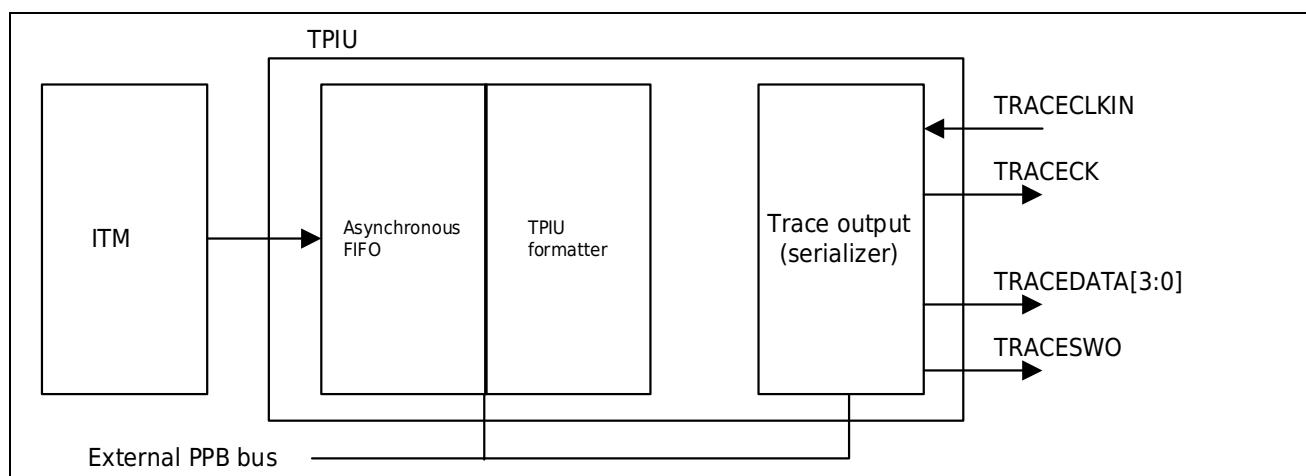


Figure 36-4 TPIU Block Diagram

36.7.2 TRACE Pin Allocation

- Asynchronous mode

Asynchronous mode requires 1 extra pin and is available in all packages. Asynchronous mode is only available when using serial line mode (not available in JTAG mode).

TPIU pin name	Track asynchronous mode	
	Types of	Note
TRACESWO	O	TRACE asynchronous data output

- Synchronous mode

Synchronous mode requires 2 to 5 extra pins, depending on the length of the data being traced, and is only available on 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 of	Note
TRACECK	O	TRACE clock
TRACED[3:0]	O	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 CPU.

Also, the number of pins to assign depends on the trace configuration (asynchronous trace or synchronous trace).

- Asynchronous mode: requires 1 extra pin
- Synchronous mode: 5 additional pins required
 - TRACECK
 - TRACED[0] (if the port data length is configured as 1, 2 or 4)
 - TRACED[1] (if the port data length is configured as 2 or 4)
 - TRACED[2] (if the port data length is configured as 4)
 - TRACED[3] (if the 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). By default the TRACE pin is not assigned.

This register is mapped on the external PPB bus and is reset by power-on (not pin reset). This register can be written 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 bit TRACE_IOEN =1 TRACE_MODE=10	Freed*	TRACECK	TRACED[0]	TRACED[1]	Freed	Freed
Sync tracking 4 bit TRACE_IOEN =1 TRACE_MODE=11	Freed*	TRACECK	TRACED[0]	TRACED[1]	TRACED[2]	TRACED[3]

Note:

- When using serial mode, release this pin. But when using JTAG, this pin is assigned to TDO.

36.7.3 MCU Internal TRACECLKIN Connection

In this MCU, the clock TRACECLKIN of TPIU is connected to the internal clock. The default clock of 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 adopted in the system reset state, the MRC calibration value will be updated every time the system reset is released. Therefore, the trace port analyzer (TPA) should not enable trace (using the TRACE_IOEN bit) in the system reset state, because the bit width of the sync frame packet in the reset state is different from the packet after reset.

36.7.4 TPIU Register

The TPIU APB register can only be read from or written to when bit TRCENA in the Debug Exception and Monitor Control Register (DEMCR) is set. Otherwise, these registers will read zero (the output of this bit enables the clock to the TPIU).

36.7.5 TPIU Configuration Example

- Set bit TRCENA in the Debug Exception and Monitor Control Register (DEMCR)
- Write desired value to TPIU current port size register (default 0x1 for 1-bit port size)
- Write 0x102 to TPIU formatter and refresh control register (default value)
- Write TPIU select pin protocol to select synchronous mode or asynchronous mode. Example: 0x2 means asynchronous NRZ mode (similar to USART)
- Write 0x20 to the MCUTRACECTL control register (bit IO_TRACEN) to assign TRACE I/O for asynchronous mode.
- Send TPIU synchronization packet at this time (FF_FF_FF_7F)
- Configure the ITM and write to the ITM stimulus register to output a value

Version Revision History

Version Number	Revision Date	Content Modification
Rev1.0	2019/11/12	First edition release.
Rev1.1	2020/01/10	<ul style="list-style-type: none">1) Add 256KB product description to the full text;2) Added VFBGA package description in the full text;3) Initialization Configuration (ICG)Describe the modification of clerical errors;4) Control Register (WDT_CR)Correction of typos;5) Table 34-1 CRC_RESTLT is changed to CRC_RESLT;6) Modified the current max value at 105°C in power-down mode in Electrical Characteristics.
Rev1.2	2020/08/26	<ul style="list-style-type: none">1) Add description of ultra-high-speed operation mode, update CoreMark/DMIPS, add switching process between ultra-high-speed simulation and high-speed mode, and ultra-low-speed mode, add BOR/PVD characteristics and current characteristics in ultra-high-speed mode. Update the waiting period of SRAM and Flash at 200Mhz, and the waiting period when reading the port. Update the frequency value in the bus architecture and update the functional block diagram;2) 256KB models are added to the pin configuration diagram;3) Add the AOS chapter, and add the common trigger source enable bit of each AOS target register;4) Figure 6.1 TCK_SWCLK is changed to JTCK_SWCLK, and TCK is changed to JTCK;<ul style="list-style-type: none">6.10.1 Correct typos;6.11.7 The bit1-0 of CMU_XTAL32NFR was misrepresented and modified;6.11.13 "Frequency calibration must be within the guaranteed range of LRC frequency" typo correction;6.11.15 The bit27-24 of CMU_PLLCFG was incorrectly modified;5) 7.4.3 In order to reduce power consumption, add the description of the corresponding AD function enable bit of PWR_FCG3 before entering STOP mode;6) 7.7 Update the reset value of PWR_PWRC3/ PWR_XTAL32CS/ PWR_STPMCR; delete the underline of the WKE_n_m bit in PWR_PDWKE0/ PWR_PDWKE1; delete the XTAL32 stop wake-up related bits in PWR_PDWKF1/ PWR_PDWKE2; PWR_FCG0/ PWR_FCG1/ PWR_FCG The function description of 2 is more detailed;<ul style="list-style-type: none">7.7.13 Unify the labels of Ret-SRAM/SRAMHS/SRAMECC in the PWR_FCG register and 10.2 SRAM register;The description method of the reserved bits in the unified power control (PWC);Update the clear description of the PVD2DETFLG and PVD1DETFLG bits in the PWR_PVDDSR register;7) 9.6.3 Modify step 10;<ul style="list-style-type: none">9.9.2 b1 FSLP is changed to FSTP;9.9.3 b24 CRST0 is changed to CRST;9 Overview erasing is changed to sector erasing;9.9.7 RDCOLERRITE is changed to COLERRITE, and read conflicts are changed to read-write conflicts;8) 12.3 In the interrupt vector table, USART_x_EI / USART_x_RTO are changed to USART_x_REI / USART_x_RTI respectively;

Version Number	Revision Date	Content Modification
		<p>12.4.9 Added "The use of internal trigger events needs to clear the PWR_FCG0.AOS bit to enable the peripheral circuit trigger function";</p> <p>12.5.2 INT_NMIENR.PRENR is changed to INT_NMIENR.REPENR, and INT_NMIENR.RDEDENR is changed to INT_NMIENR.RECCENR;</p> <p>12.5.3 INT_NMIFR.RPEFR is changed to INT_NMIFR.REPFR, and INT_NMIFR.RDEDFR is changed to INT_NMIFR.RECCFR;</p> <p>12.5.4 INT_NMICFR.RPECFR changed to INT_NMICFR.REPCFR, INT_NMICFR.RDEDCFR changed to INT_NMICFR.RECCCFR;</p> <p>12.5.6 Delete the INT_ prefix in the bit description of the INT_EIFR register;</p> <p>12.5.7 Delete the INT_ prefix in the bit description of the INT_EICFR register;</p> <p>12.5.10 Add the prefix V in the bit description of the INT_VSEL register;</p> <p>12.5.14 Delete the INT_ prefix in the bit description of the INT_IER register;</p> <p>9) 15.3 Update application examples;</p> <p>15.4.7 Update register description;</p> <p>15.3.1/15.3.2 Delete illegal access action description;</p> <p>15.4.1 Register flag error modification;</p> <p>15.4.3 b17/b16/b9/b8/b1/b0 mark and function error modification;</p> <p>10) 16.4 Update application examples;</p> <p>11) 18.5.3 In order to reduce power consumption, add the description of the corresponding AD function enable bit of PWR_FCG3 before entering STOP mode;</p> <p>12) In Chapter 20, optimize the legend of Timer6; add [Pulse Width Measurement], [Period Measurement], [Typical Application Example] chapters; add cycle by cycle description of EMB control;</p> <p>13) 21.5.3 CCSR.DCLK is changed to CCSR.CKDIV;</p> <p>14) In chapters 23 and 24, optimize the legends of Timer0 and TimerA; Modify the description of bit9~8 of TimerA CCONR register and bit1~0 of PCONR register in Chapter 23;</p> <p>15) 25.5.3 ALMF is omitted in the RTC_CR2 register bit description;</p> <p>16) 27.5.3 The initial value modification bit of USART_BRR is 0x0000 FFFF; 27.5.6 BCN bit width matching in the USART_CR3 register table;</p> <p>17) 32.4.3 RAIE is changed to RAFIE; 32.5.7 The bit5 of CAN_RTIE was misrecorded and modified; 32.5.8 The bit5 of CAN_RTIF was misrecorded and modified; 32.5.9 CAN_ERRINT Form b5 WPIE is changed to EPIE; 32.5.12 CAN_AFWL changed to CAN_LIMIT; 32 Change TRG_TRIG to TT_TRIG in all chapters, and TRG_WTRIG to TT_WTRIG; 32.5.12 CAN_AFWL changed to CAN_LIMITt;</p> <p>18) Chapter 37 adds the description of the MMC mode enable register;</p> <p>19) 33.6.2 HFI is changed to HFIR; Add 33.7.2.1 USBFS VBUS control register; 33.7.2.5 b31 WKUPINT changed to WKUINT, b7 GOUTNAKEFF changed to GONAKEFF;</p> <p>20) 33.7.2.6 b31 WKUPINTM is changed to WKUIM, b30 VBUSVM is changed to VBUSWIM, b29 DISCM is changed to DISCIM, b28 CIDSCHG is changed to CIDSCHGM, b7 GOUTNAKEFFM is changed</p>

Version Number	Revision Date	Content Modification
		to GONAKEFFM; 33.7.3.4 PTXQTOP is changed to PTXQSAV; 33.5.4.4 Change after 15ms to within 15ms; 33.7.4.8 INEPxkTXFEM changed to INEPTXFEM; 33.7.4.2 Delete the TCTL bit in the table; 33.7.4.11 TO is changed to TOC and ITTXFE is changed to TTXFE; 21) 38.4.1 Updated JTAG/SWJ debug port pins.
Rev1.3	2021/12/10	1) Delete product features, pin configuration, packaging information, etc. (please refer to the latest Data Manual for related information), and modify the statement; 2) Update some names and optimize descriptions, and update clerical errors; 3) Supplement the precision description of UART communication; 4) Timer 6 adds a description of the initialization method of the effective function of the interval period; 5) Added instructions on CAN use precautions and sampling points; 6) 18.3.4 The digital filter function of this group of ports is realized between unit 1 -> the digital filter function of this group of ports is set by the FCONR of unit 1, and the FCONR of other units is invalid for the digital filter function of this group of ports, so any When the unit uses this function, it needs to set the TIMER6_1 bit in the function controller (PWR_FCG2); 7) 19.5.7 TMR4 OCERn b13, b12 description modification;
Rev1.4	2021/12/30	Added HC32F45x series, modifiedFigure 5-6,Figure 5-8,Figure 18-13 and Table 18-3.
Rev1.41	2022/03/09	The company logo is updated.
Rev1.42	2022/03/29	1) In Table 1-1, the description of the data security protection address is added, and "ROM" is changed to "FLASH"; 2) 4.11.1 Correction of clerical errors in digit b7; CMU XTAL configuration register b5~b4 function description modification; 3) 6.1 "Addresses 0x0000_0408~0x0000_041F are reserved functions" is changed to "Addresses 0x0000_0408~0x0000_040F and 0x0000_0418~0x0000_041F are reserved functions"; 6.2.3 Initialize configuration register split description, add 6.2.4/6.2.5/6.2.6; 4) 7.2 Modify the description about supporting data security protection in the main features; 7.3 Add data security protection addresses and related descriptions to FLASH map (Figure 7-1, Figure 7-2); 7.6.5 Added a new description about data security protection; 5) 18.3.11.3 Add the description "When using this function, it can only be realized by the combination of unit 1 and 2, unit 1 is used as a position counting unit, and unit 2 is used as a revolution counting unit"; 6) 20.3.1 EMB_CTL0 register bits PWMSEN[2:0], CMPEN[2:0] description modification; 20.3.2 EMB_CTL1~3 register bit CMPEN[2:0] description modification; 20.3.3 EMB_PWMLV0 register bit PWMLV description modification; 7) 21.3.1.3 Figure 21-5 Correction of clerical errors; 8) 25.5.1 b1 "CLR.CFE" is changed to "CR1.CFE"; 25.5.4 b20 "CRTOF bit writes 1 to clear CRTOF" is changed to "CRTOF bit writes 1 to clear RTOF";

Version Number	Revision Date	Content Modification
		<p>9) 26.1 Correction of clerical errors; 26.2.1 The arrows in Figure 26-1 indicate incorrect revisions;</p> <p>10) 30.4.1 Add the description of CAN communication clock restriction; 30.4.5 Add the description of data sending restrictions; 30.5.1 Change the offset address "0x50" to "0x00", and "mailbox" to "message"; 30.5.2 The offset address "0x00" is changed to "0x50"; 30.5.8 Modify the function description of bit b6; 30.5.9 b7 bit R/W modification; b6, b0 bit function description modification; 30.5.11 b4~b0 bit R/W modification; 30.5.13 R/W modification of bits b7~b0; 30.5.14 R/W modification of bits b7~b0; 30.5.15 Modification of b3~b0 function description; 30.5.19 Modify the function description of bit b5.</p>
Rev1.5	2022/09/14	<p>1) Add HC32A460 series product model;</p> <p>2) The introduction chapter "HC32F460" is changed to "HC32F460_F45x_A460"; Add the description of "suitable for automotive electronics";</p> <p>3) 18.3.4 Modification of digital filter function description;</p> <p>4) 19.3.1.3 Figure 19-6 The flat peak of the original sawtooth wave is changed to a sharp peak; 19.3.3.1 Figure 19-16 The flat peak of the original sawtooth wave is changed to a sharp peak; 19.3.3.2 Figure 19-18 The flat peak of the original sawtooth wave is changed to a sharp peak; 19.5.3 In the function description of bit b15 of register TMR4_CCSR, the description of "Note 2" has been modified;</p> <p>5) 23.3.1 RTC register reset description modification; 23.5.14 Correction of clerical errors, "hourly alarm clock register" is changed to "weekly alarm clock register";</p> <p>6) 25.5.1 Correction of clerical error in the function description of bit b1 of register USART_SR;</p> <p>7) 27.12.3 Correction of clerical error in the function description of bits b30~b28 of register SPI_CFG1; 28.3.1 Correction of clerical errors in Figure 28-4 and Figure 28-7; 28.4.5 Correction of clerical errors; 28.5.6 Correction of clerical errors in Figure 28-20;</p> <p>8) 30.4.17 Receive BUF full interrupt flag typo modification; 30.5.18 Register CAN_TBSLOT bit b6 mark typo modification; 30.5.23 Register CAN_TT_WTRIG bits b15~b0 marked by clerical error modification;</p> <p>9) 33.4.4 Register DCU_FLAGCLR read and write changed to "R/W".</p>