



Technology Innovator

Puya

PY32T090 Reference Manual

32-bit ARM[®] Cortex[®]-M0+ Microcontroller



Puya Semiconductor (Shanghai) Co., Ltd.

Contents

1. Documentation conventions	26
1.1. List of abbreviations for registers	26
1.2. Availability of peripherals	26
2. System block diagram	27
3. Memory and bus architecture	28
3.1. System architecture	28
3.2. Memory organization	29
3.2.1. Introduction	29
3.2.2. Memory map	30
3.2.3. Register map	31
3.3. Embedded SRAM	33
3.3.1. SRAM parity	33
3.3.2. SRAM Write Protection	33
3.4. Flash memory	34
3.5. Start Mode	34
3.5.1. Physical mapping	35
3.5.2. Embedded boot loader	35
4. Embedded Flash	36
4.1. Introduction to Flash	36
4.2. Flash main features	36
4.3. Flash Feature Description	36
4.3.1. Flash memory organization	36
4.3.2. Flash ECC	37
4.3.3. Flash read access latency	37
4.3.4. Flash acceleration	38
4.3.5. Flash programming and erase operations	38
4.4. Unique device ID (UID)	41
4.5. Flash option bytes	42
4.5.1. Description of Flash option bytes	42
4.5.2. User option byte 0	43
4.5.3. User option byte 1	43
4.5.4. Flash protection configuration 2 (BANK0_WRP)	44
4.5.5. Flash protection configuration 3 (BANK1_WRP)	45
4.5.6. Flash Protection Configuration 4 (PCROP0SR)	45
4.5.7. Flash Protection Configuration 5 (PCROP0ER)	45
4.5.8. Flash Protection Configuration 6 (PCROP1SR)	46
4.5.9. Flash Protection Configuration 7 (PCROP1ER)	46

4.5.10.	Flash Option Byte Programming	47
4.6.	Flash factory configuration bytes	48
4.6.1.	HSI_TRIMMING_FOR_USER.....	49
4.6.2.	NORMAL_TS_CALIBRATION_PARA.....	49
4.6.3.	HIGH_TS_CALIBRATION_PARA.....	49
4.6.4.	HSI2M_EPPARA0	49
4.6.5.	HSI2M_EPPARA1	49
4.6.6.	HSI2M_EPPARA2	50
4.6.7.	HSI2M_EPPARA3	50
4.6.8.	HSI2M_EPPARA4	50
4.6.9.	HSI2M_EPPARA5	50
4.6.10.	HSI2M_EPPARA6	50
4.7.	Flash user data bytes.....	51
4.7.1.	Flash user data description	51
4.7.2.	Flash user data programming	51
4.8.	Flash storage protection	52
4.8.1.	Proprietary code readout protection (PCROP).....	52
4.8.2.	Flash read protection.....	53
4.8.3.	Flash write protection (WRP)	54
4.8.4.	Option byte write protection.....	54
4.8.5.	Summary of Flash memory protection permissions	55
4.9.	Flash interrupt	56
4.10.	Flash Register	56
4.10.1.	Flash access control register (FLASH_ACR).....	56
4.10.2.	Flash key register (Flash_KEYR)	56
4.10.3.	Flash option byte key register (FLASH_OPTKEYR)	57
4.10.4.	Flash status register (FLASH_SR)	57
4.10.5.	Flash control register (FLASH_CR).....	58
4.10.6.	Flash ECC register (FLASH_ECCR).....	60
4.10.7.	Flash option byte register (FLASH_OPTR).....	61
4.10.8.	Flash write protection register (FLASH_WRP)	62
4.10.9.	Flash Bank0 proprietary code protection address register (FLASH_PCROPR0)	63
4.10.10.	Flash Bank1 proprietary code protection address register (FLASH_PCROPR1)	63
4.10.11.	Flash low-power configuration register (FLASH_LPCR)	63
4.10.12.	Flash TS0 register (FLASH_TS0).....	64
4.10.13.	Flash TS1 Register (Flash_TS1).....	64
4.10.14.	Flash TS2P Register (FLASH_TS2P)	64
4.10.15.	Flash TPS3 Register (FLASH_TPS3)	65
4.10.16.	Flash TS3 Register (FLASH_TS3)	65
4.10.17.	Flash Page Erase TPE Register (FLASH_PERTPE)	65

4.10.18.	Flash sector erase time TPE register (FLASH_SMERTPE)	66
4.10.19.	Flash programming time TPE register (FLASH_PRGTPE).....	66
4.10.20.	Flash pre-programming time TPE register (FLASH_PRETPE).....	66
4.10.21.	Flash page write wait time TACLK2PW register (FLASH_TACLK2PW).....	66
5.	Power Management Unit (PMU)	68
5.1.	Power supply.....	68
5.1.1.	Power Structure	68
5.1.2.	System Main Power Supply	69
5.1.3.	Dynamic voltage regulation	69
5.2.	Power monitoring	70
5.2.1.	Power-on reset (POR)/Power-down reset (PDR)/Brown-out reset (BOR).....	70
5.2.2.	Programmable Voltage Detector (PVD)	71
5.3.	Low power control	72
5.3.1.	Power mode	72
5.3.2.	Reduce the system clock frequency.....	79
5.3.3.	Software Flow.....	79
5.4.	PWR registers	81
5.4.1.	Power Control Register 1 (PWR_CR1)	81
5.4.2.	Power control register 2 (PWR_CR2)	82
5.4.3.	Power control register 3 (PWR_CR3)	83
5.4.4.	Power control register 4 (PWR_CR4)	84
5.4.5.	Power status register (PWR_SR).....	85
5.4.6.	Power status clear register (PWR_SCR)	87
5.4.7.	Power PortA Pull-Up Control Register (PWR_PUCRA).....	88
5.4.8.	Power PortA Pull-Down Control Register (PWR_PDCRA)	88
5.4.9.	Power PortB pull-up control register (PWR_PUCRB)	88
5.4.10.	Power PortB pull-down control register (PWR_PDCRB)	89
5.4.11.	Power PortC pull-up control register (PWR_PUCRC).....	89
5.4.12.	Power PortC Pull-Down Control Register (PWR_PDCRC).....	89
5.4.13.	Power PortD Pull-Up Control Register (PWR_PUCRD)	90
5.4.14.	Power PortD Pull-Down Control Register (PWR_PDCRD).....	90
6.	Reset and Clock Control (RCC).....	91
6.1.	Reset.....	91
6.1.1.	Power reset	91
6.1.2.	System reset.....	91
6.1.3.	RTC domain reset	93
6.2.	Clocks	93
6.2.1.	Clock sources.....	93
6.2.2.	Clock structure.....	96

6.2.3.	System Clock (SYSCLK) Selection	96
6.2.4.	Clock Security System (CSS).....	97
6.2.5.	TIM16/TIM17 measures internal and external clocks	98
6.2.6.	Clock Output.....	100
6.2.7.	Module clock types	100
6.3.	RCC registers.....	101
6.3.1.	RCC Clock Control Register (RCC_CR)	101
6.3.2.	RCC Internal Clock Source Calibration Register (RCC_ICSCR)	103
6.3.3.	RCC clock configuration register (RCC_CFGR)	104
6.3.4.	RCC PLL Configuration Register (RCC_PLLCFGR)	105
6.3.5.	RCC external clock source configuration register (RCC_ECSCR)	106
6.3.6.	RCC clock interrupt enable register (RCC_CIER)	106
6.3.7.	RCC clock interrupt flag register (RCC_CIFR).....	107
6.3.8.	RCC clock interrupt clear register (RCC_CICR)	108
6.3.9.	RCC I/O port reset register (RCC_IOPRSTR)	108
6.3.10.	RCC AHB peripheral reset register (RCC_AHBSTR).....	109
6.3.11.	RCC APB peripheral reset register 1 (RCC_APBSTR1).....	109
6.3.12.	RCC APB peripheral reset register 2 (RCC_APBSTR2).....	111
6.3.13.	RCC I/O port clock enable register (RCC_IOPENR)	112
6.3.14.	RCC AHB peripheral clock enable register (RCC_AHBENR).....	112
6.3.15.	RCC APB Peripheral Clock Enable Register 1 (RCC_APBENR1)	113
6.3.16.	RCC APB peripheral clock enable register 2 (RCC_APBENR2)	114
6.3.17.	RCC peripheral clock configuration register (RCC_CCIPR)	115
6.3.18.	RCC RTC Field Control Register (RCC_BDCR).....	117
6.3.19.	RCC Control/Status Register (RCC_CSR)	118
7.	General-purpose I/Os (GPIO).....	120
7.1.	Introduction to GPIO	120
7.2.	GPIO main features	120
7.3.	GPIO functional description	120
7.3.1.	General-purpose I/Os (GPIO)	122
7.3.2.	I/O pin alternate function	122
7.3.3.	I/O port control registers.....	123
7.3.4.	I/O data registers	123
7.3.5.	I/O data bit manipulation	123
7.3.6.	GPIO locking mechanism	124
7.3.7.	I/O alternate function input/output	124
7.3.8.	External interrupt line/wake-up line	124
7.3.9.	Input configuration	124
7.3.10.	Output configuration	125
7.3.11.	Alternate function configuration.....	125

7.3.12.	Analog configuration.....	126
7.3.13.	Using HSE/LSE pins as GPIO.....	127
7.3.14.	Backup power domain GPIO pins	127
7.4.	GPIO registers	127
7.4.1.	GPIO port mode register (GPIOx_MODER) (x=A,B,C,D).....	127
7.4.2.	GPIO port output type register (GPIOx_OTYPER) (x = A,B,C,D).....	128
7.4.3.	GPIO port output drive register (GPIOx_OSPEEDR) (x = A,B,C, D).....	128
7.4.4.	GPIO port pull-up/pull-down register (GPIOx_PUPDR) (x = A,B,C, D)	129
7.4.5.	GPIO port input data register (GPIOx_IDR) (x = A,B,C, D)	129
7.4.6.	GPIO port output data register (GPIOx_ODR) (x = A,B,C, D)	130
7.4.7.	GPIO port bit set/reset register (GPIOx_BSRR) (x = A,B,C, D).....	130
7.4.8.	GPIO port configuration lock register (GPIOx_LCKR) (x = A,B,C, D).....	130
7.4.9.	GPIO alternate function low register (GPIOx_AFR1) (x = A, B, C, D)	131
7.4.10.	GPIO alternate function high register (GPIOx_AFRH) (x = A, B, C, D)	131
7.4.11.	GPIO Port Bit Reset Register (GPIOx_BRR) (x = A,B,C,D)	132
8.	Peripheral Interconnection	133
8.1.	Introduction	133
8.2.	Interconnection Details	133
8.2.1.	Timer Input Trigger (ITR).....	133
8.2.2.	Timer External Trigger (ETR).....	133
8.2.3.	Clear timer OCxREF signal	134
8.2.4.	Timer input capture	134
8.2.5.	Timer brake	135
8.2.6.	Timer System Brake.....	135
8.2.7.	Low-Power Timer LPTIM Trigger	136
8.2.8.	Infrared output waveform control (IRTIM)	136
8.2.9.	Comparator blanking.....	136
8.2.10.	ADC hardware trigger input.....	137
8.2.11.	ADC internal channel source.....	137
8.2.12.	DAC hardware trigger input.....	138
9.	System configuration controller (SYSCFG).....	139
9.1.	SYSCFG main features	139
9.2.	SYSCFG registers.....	139
9.2.1.	SYSCFG configuration register 1 (SYSCFG_CFGR1)	139
9.2.2.	SYSCFG configuration register 2 (SYSCFG_CFGR2)	140
9.2.3.	SYSCFG configuration register 3 (SYSCFG_CFGR3)	140
9.2.4.	SYSCFG configuration register 4 (SYSCFG_CFGR4)	142
9.2.5.	GPIOA noise filter enable register (PA_ENS)	142
9.2.6.	GPIOB noise filter enable register (PB_ENS)	142

9.2.7.	GPIOC noise filter enable register (PC_ENS).....	143
9.2.8.	GPIOD noise rate enable register (PD_ENS)	143
9.2.9.	I2C type IO configuration register (SYSCFG_IOCFG).....	143
9.2.10.	LED IO configuration register (SYSCFG_LEDCFG).....	144
9.2.11.	GPIOA analog PAD2 enable register (PA_ANA2EN)	145
9.2.12.	GPIOB Analog PAD2 Enable Register (PB_ANA2EN)	145
9.2.13.	GPIOC Analog PAD2 Enable Register (PC_ANA2EN)	145
9.2.14.	GPIOD analog PAD2 enable register (PD_ANA2EN).....	146
9.2.15.	SRAM control status register (SYSCFG_SCSR)	147
9.2.16.	SRAM write protection register (SYSCFG_SWPR)	147
10.	Direct Memory Access (DMA).....	148
10.1.	Introduction to DMA	148
10.2.	DMA main features	148
10.3.	DMA functional description	149
10.3.1.	DMA block diagram	149
10.3.2.	CPU and DMA bus sharing	149
10.3.3.	DMA transmission	149
10.3.4.	DMA Arbiter	150
10.3.5.	DMA channel	150
10.3.6.	Programmable data width, data alignment, and endianness	152
10.3.7.	DMA error management.....	154
10.4.	DMA interrupts	154
10.5.	DMA peripheral request mapping	155
10.6.	DMA registers.....	156
10.6.1.	DMA interrupt status register (DMA_ISR)	156
10.6.2.	DMA Interrupt Flag Clear Register (DMA_IFCR)	158
10.6.3.	DMA channel x configuration register (DMA_CCRx) (x=1~7).....	160
10.6.4.	DMA Channel x Data Transfer Number Register (DMA_CNDTRx) (x=1~7)	161
10.6.5.	DMA Channel x Peripheral Address Register (DMA_CPARx) (x=1~7)	162
10.6.6.	DMA channel x memory address register (DMA_CMARx) (x=1~7).....	162
10.6.7.	DMA channel x circular transfer configuration register (DMA_CCCFGRx) (x=1~7)	163
11.	Interrupts and events	164
11.1.	Nested Vectored Interrupt Controller (NVIC)	164
11.1.1.	NVIC main features	164
11.1.2.	SysTick calibration value register.....	164
11.1.3.	Interrupt and exception vectors	164
11.2.	Extended interrupt and event controller (EXTI).....	165
11.2.1.	EXTI main features.....	165
11.2.2.	EXTI block diagram	166

11.2.3.	EXTI function description	166
11.2.4.	EXTI configurable event input wakeup.....	167
11.2.5.	EXTI direct event input wake-up	167
11.2.6.	EXTI multiplexing.....	167
11.2.7.	EXTI Line.....	169
11.3.	EXTI register	169
11.3.1.	EXTI Rising Edge Trigger Selection Register (EXTI_RTISR).....	169
11.3.2.	EXTI Falling Trigger Selection Register (EXTI_FTSR)	170
11.3.3.	EXTI Software Interrupt Event Register (EXTI_SWIER).....	172
11.3.4.	EXTI pending register (EXTI_PR)	174
11.3.5.	EXTI external interrupt selection register 1 (EXTI_IOSELR1)	176
11.3.6.	EXTI external interrupt selection register 2 (EXTI_IOSELR2)	177
11.3.7.	EXTI Interrupt Mask Register (EXTI_IMR).....	178
11.3.8.	EXTI event mask register (EXTI_EMR)	180
12.	Cyclic redundancy check calculation unit (CRC).....	182
12.1.	Introduction to CRC.....	182
12.2.	CRC main features.....	182
12.3.	CRC functional description.....	183
12.3.1.	CRC block diagram	183
12.3.2.	CRC functional description	183
12.4.	CRC registers.....	184
12.4.1.	CRC Data Register (CRC_DR)	184
12.4.2.	CRC Independent Data Register (CRC_IDR)	185
12.4.3.	CRC Control Register (CRC_CR)	185
12.4.4.	CRC Initial Value Register (CRC_INIT)	185
12.4.5.	CRC Polynomial Register (CRC_POL)	186
13.	Analog-to-Digital Converter (ADC).....	187
13.1.	Introduction to ADC	187
13.2.	ADC main features	187
13.3.	ADC functional description.....	189
13.3.1.	ADC block diagram	189
13.3.2.	ADC single-ended and differential channels	189
13.3.3.	ADC Calibration (ADCAL)	191
13.3.4.	Software read/write calibration factor	192
13.3.5.	ADC switch control (ADEN, ADDIS, ADRDY).....	192
13.3.6.	ADC control bit register operations	193
13.3.7.	ADC clock.....	194
13.3.8.	ADC Channel Selection (SQRx, JSQR).....	194
13.3.9.	ADC programmable sampling time (SMP)	195

13.3.10.	ADC single conversion mode (CONT=0, DISCEN=0)	195
13.3.11.	ADC continuous conversion mode (CONT=1, DISCEN=0)	196
13.3.12.	ADC discontinuous conversion mode (DISCEN=1 or JDISCEN=1)	197
13.3.13.	ADC injected channel management	198
13.3.14.	ADC conversion start (ADSTART, JADSTART).....	199
13.3.15.	ADC timing.....	200
13.3.16.	ADC stops ongoing conversion (ADSTP, JADSTP).....	200
13.3.17.	ADC hardware trigger and trigger polarity (EXTSEL, EXTEN, JEXTSEL, JEXTEN)	202
13.3.18.	ADC Programmable Resolution (RES) – Fast Conversion	202
13.3.19.	ADC Conversion/Sampling End (EOC, JEOC, EOSMP)	203
13.3.20.	ADC Conversion Sequence End (EOS, JEOS).....	203
13.3.21.	ADC Timing Examples (Single/Continuous Mode, Hardware/Software Trigger)	204
13.3.22.	ADC data management	205
13.3.23.	ADC analog watchdog (AWDEN, JAWDEN, AWDSGL, AWDCH, AWD_TR).....	211
13.3.24.	ADC Dynamic Low-Power Feature.....	213
13.3.25.	ADC oversampling.....	217
13.3.26.	Temperature sensor	221
13.3.27.	Monitoring the internal reference voltage	223
13.3.28.	Monitoring the power supply.....	224
13.4.	ADC interrupts.....	224
13.5.	ADC registers	224
13.5.1.	ADC Interrupt Status Register (ADC_ISR).....	224
13.5.2.	ADC Interrupt Enable Register (ADC_IER).....	225
13.5.3.	ADC control register (ADC_CR).....	226
13.5.4.	ADC configuration register (ADC_CFGR).....	228
13.5.5.	ADC configuration register 2 (ADC_CFGR2).....	231
13.5.6.	ADC Sample Time Register 1 (ADC_SMPR1).....	233
13.5.7.	ADC Sample Time Register 2 (ADC_SMPR2).....	233
13.5.8.	ADC Sample Time Register 3 (ADC_SMPR3).....	233
13.5.9.	ADC Sample Time Register 4 (ADC_SMPR4).....	234
13.5.10.	ADC watchdog threshold register (ADC_TR).....	234
13.5.11.	ADC channel selection register 1 (ADC_SQR1)	234
13.5.12.	ADC channel selection register 2 (ADC_SQR2)	236
13.5.13.	ADC channel selection register 3 (ADC_SQR3)	236
13.5.14.	ADC channel selection register 4 (ADC_SQR4)	237
13.5.15.	ADC data register (ADC_DR).....	237
13.5.16.	ADC injected channel selection register (ADC_JSQR).....	238
13.5.17.	ADC offset register (ADC_OFRy).....	239
13.5.18.	ADC injected channel data register (ADC_JDRy)	240
13.5.19.	ADC calibration factor register (ADC_CALFACT)	240

13.5.20.	ADC gain compensation register (ADC_GCOMP)	241
13.5.21.	ADC common configuration register (ADC_CCR)	242
14.	Digital-to-analog converter (DAC)	244
14.1.	Introduction	244
14.2.	DAC main features	244
14.3.	DAC functional description	245
14.3.1.	DAC block diagram	245
14.3.2.	DAC channel enable.	245
14.3.3.	DAC output buffer enable	245
14.3.4.	DAC calibration	245
14.3.5.	DAC data format	246
14.3.6.	DAC conversion	246
14.3.7.	DAC output voltage	247
14.3.8.	DAC Trigger Selection	247
14.3.9.	DMA functionality	248
14.3.10.	DAC Noise Generation	248
14.3.11.	DAC Triangle Wave Generation	249
14.3.12.	DAC output clock	251
14.3.13.	DAC output	251
14.4.	DAC registers	251
14.4.1.	DAC control register 1 (DAC_CR1)	251
14.4.2.	DAC control register 2 (DAC_CR2)	253
14.4.3.	DAC 12-bit right-aligned data holding register (DAC_DHR12R)	253
14.4.4.	DAC 12-bit left-aligned data holding register (DAC_DHR12L)	253
14.4.5.	DAC 8-bit right-aligned data holding register (DAC_DHR8R)	254
14.4.6.	DAC data output register (DAC_DOR)	254
14.4.7.	DAC status register (DAC_SR)	254
14.4.8.	DAC calibration register (DAC_CALR)	254
15.	Comparator (COMP)	256
15.1.	Introduction to COMP	256
15.2.	COMP main features	256
15.3.	COMP functional description	257
15.3.1.	COMP block diagram	257
15.3.2.	COMP interface and internal signals	257
15.3.3.	COMP reset and clocks	257
15.3.4.	Filtering function of COMP	258
15.3.5.	COMP output blanking	258
15.3.6.	COMP locking mechanism	259
15.3.7.	Window comparator	259

15.3.8.	Hysteresis	260
15.3.9.	Low-power modes	260
15.4.	COMP interrupts	261
15.5.	COMP registers	261
15.5.1.	Comparator 1 control and status register (COMP1_CSR)	261
15.5.2.	COMP1 filter register (COMP1_FR)	262
15.5.3.	COMP Common Control and Status Register (COMP_CCSR)	263
15.5.4.	COMP2 Control and Status Register (COMP2_CSR)	263
15.5.5.	COMP2 filter register (COMP2_FR)	264
16.	Operational amplifier (OPA)	266
16.1.	Introduction	266
16.2.	OPA main features	266
16.2.1.	OPA functional description	266
16.3.	OPA registers	266
16.3.1.	OPA Output Control Register (OPA_OCR)	266
16.3.2.	OPA Control Register (OPA_CR)	267
17.	Liquid Crystal Display Controller (LCD)	268
17.1.	Introduction	268
17.2.	LCD main features	268
17.3.	LCD functional description	269
17.3.1.	LCD block diagram	269
17.3.2.	LCD clock (refresh frequency)	269
17.3.3.	LCD scanning waveform	269
17.3.4.	LCD contrast control	271
17.3.5.	LCD Display Data Storage Mode	273
17.3.6.	LCD Bias Circuit	279
17.3.7.	Double Buffer Memory	280
17.4.	LCD interrupts	280
17.5.	Software configuration process	280
17.5.1.	LCD initialization process	280
17.5.2.	LCD data update	281
17.5.3.	Procedure for disabling the LCD	281
17.6.	LCD registers	281
17.6.1.	LCD Control Register (LCD_CR)	281
17.6.2.	LCD Control Status Register (LCD_CSR)	283
17.6.3.	LCD interrupt clear register (LCD_INTCLR)	284
17.6.4.	LCD output configuration register 0 (LCD_POEN0)	285
17.6.5.	LCD output configuration register 1 (LCD_POEN1)	285
17.6.6.	LCD_RAM 0 to 7	285

17.6.7.	LCD_RAM8~15	286
18.	Advanced Timer (TIM1)	287
18.1.	TIM1 introduction	287
18.2.	TIM1 main features	287
18.3.	TIM1 functional description	288
18.3.1.	Functional block diagram	288
18.3.2.	Time-base unit.....	288
18.3.3.	Counting modes	289
18.3.4.	Repetition counter	295
18.3.5.	External trigger input	296
18.3.6.	Clock sources.....	297
18.3.7.	Capture/Compare channels	299
18.3.8.	Input capture mode:	300
18.3.9.	PWM input mode	301
18.3.10.	Forced Output Mode.....	301
18.3.11.	Output compare mode.....	302
18.3.12.	PWM mode.....	302
18.3.13.	Complementary outputs and dead-time insertion.....	304
18.3.14.	Using the break function.....	305
18.3.15.	Clearing the OCxREF signal upon an external event	307
18.3.16.	6-step PWM.....	308
18.3.17.	One-pulse mode	308
18.3.18.	Retriggerable one pulse mode (OPM).....	310
18.3.19.	Encoder interface mode	310
18.3.20.	Timer input XOR mode.....	312
18.3.21.	Connecting Hall sensors.....	312
18.3.22.	Timer and External Trigger Synchronization	313
18.3.23.	Timer synchronization	315
18.3.24.	DMA Continuous Transfer Mode	316
18.4.	TIM1 Interrupt.....	316
18.5.	Debug mode.....	317
18.6.	TIM1 registers	317
18.6.1.	TIM1 control register 1 (TIM1_CR1)	317
18.6.2.	TIM1 control register 2 (TIM1_CR2)	318
18.6.3.	TIM1 slave mode control register (TIM1_SMCR).....	320
18.6.4.	TIM1 DMA/interrupt enable register (TIM1_DIER).....	322
18.6.5.	TIM1 status register (TIM1_SR)	323
18.6.6.	TIM1 event generation register (TIM1_EGR).....	326
18.6.7.	TIM1 Capture/Compare Mode Register 1 (TIM1_CCMR1)	327
18.6.8.	TIM1 Capture/Compare Mode Register 2 (TIM1_CCMR2)	330

18.6.9.	TIM1 capture/compare enable register (TIM1_CCER)	332
18.6.10.	TIM1 counter register (TIM1_CNT)	334
18.6.11.	TIM1 prescaler register (TIM1_PSC)	334
18.6.12.	TIM1 Auto-reload Register (TIM1_ARR)	334
18.6.13.	TIM1 Repetition Counter Register (TIM1_RCR)	335
18.6.14.	TIM1 Capture/Compare Register 1 (TIM1_CCR1)	335
18.6.15.	TIM1 capture/compare register 2 (TIM1_CCR2)	336
18.6.16.	TIM1 capture/compare register 3 (TIM1_CCR3)	336
18.6.17.	TIM1 capture/compare register 4 (TIM1_CCR4)	337
18.6.18.	TIM1 break/dead-time register (TIM1_BDTR)	337
18.6.19.	TIM1 input selection register (TIM1_TISEL)	339
18.6.20.	TIM1 alternate function option register 1 (TIM1_AF1)	339
18.6.21.	TIM1 alternate function option register 2 (TIM1_AF2)	341
18.6.22.	TIM1 DMA control register (TIM1_DCR)	341
18.6.23.	TIM1 DMA continuous transfer register (TIM1_DMAR)	341
19.	General-purpose timers (TIM2/TIM3)	343
19.1.	TIM2/TIM3 introduction	343
19.2.	TIM2/TIM3 main features	343
19.3.	TIM2/TIM3 functional description	344
19.3.1.	Functional block diagram	344
19.3.2.	Time-base unit	345
19.3.3.	Counting modes	346
19.3.4.	Clock sources	352
19.3.5.	Capture/Compare channels	355
19.3.6.	Input capture mode:	356
19.3.7.	PWM input mode	356
19.3.8.	Forced Output Mode	357
19.3.9.	Output compare mode	357
19.3.10.	PWM mode	358
19.3.11.	Clearing the OCxREF signal on an external event	360
19.3.12.	One-pulse mode	361
19.3.13.	Encoder interface mode	362
19.3.14.	Timer input XOR mode	363
19.3.15.	Timer and External Trigger Synchronization	363
19.3.16.	Timer synchronization	366
19.4.	TIM2/TIM3 interrupts	369
19.5.	TIM2/TIM3 debug mode	369
19.6.	TIM2/TIM3 registers	370
19.6.1.	TIMx Control Register 1 (TIMx_CR1) (x=2,3)	370
19.6.2.	TIMx control register 2 (TIMx_CR2) (x=2,3)	371

19.6.3.	TIMx slave mode control register (TIMx_SMCR) (x=2,3).....	372
19.6.4.	TIMx DMA/interrupt enable register (TIMx_DIER) (x=2,3).....	374
19.6.5.	TIMx status register (TIMx_SR) (x=2,3).....	375
19.6.6.	TIMx Event Generation Register (TIMx_EGR) (x=2,3).....	377
19.6.7.	TIMx capture/compare mode register 1 (TIMx_CCMR1) (x=2,3).....	378
19.6.8.	TIMx capture/compare mode register 2 (TIMx_CCMR2) (x=2,3).....	381
19.6.9.	TIMx capture/compare enable register (TIMx_CCER) (x=2,3).....	382
19.6.10.	TIMx counter register (TIMx_CNT) (x=2,3).....	383
19.6.11.	TIMx prescaler register (TIMx_PSC) (x=2,3).....	384
19.6.12.	TIMx auto-reload register (TIMx_ARR) (x=2,3).....	384
19.6.13.	TIMx capture/compare register 1 (TIMx_CCR1) (x=2,3).....	384
19.6.14.	TIMx capture/compare register 2 (TIMx_CCR2) (x=2,3).....	385
19.6.15.	TIMx capture/compare register 3 (TIMx_CCR3) (x=2,3).....	385
19.6.16.	TIMx capture/compare register 4 (TIMx_CCR4) (x=2,3).....	386
19.6.17.	TIMx input selection register (TIMx_TISEL) (x=2,3).....	386
19.6.18.	TIMx alternate function register 1 (TIMx_AF1) (x=2,3).....	387
19.6.19.	TIMx alternate function register 2 (TIMx_AF2) (x=2,3).....	387
19.6.20.	TIMx DMA control register (TIMx_DCR) (x=2,3).....	387
19.6.21.	TIMx DMA burst transfer address register (TIMx_DMAR) (x=2,3).....	388
20.	General-purpose timers (TIM15).....	389
20.1.	TIM15 main features	389
20.2.	TIM15 functional description	390
20.2.1.	Functional block diagram	390
20.2.2.	Time-base unit.....	390
20.2.3.	Counting mode (upcounting).....	391
20.2.4.	Repetition counter	393
20.2.5.	Clock sources.....	394
20.2.6.	Capture/Compare channels	396
20.2.7.	Input capture mode:	397
20.2.8.	PWM input mode	398
20.2.9.	Forced Output Mode	399
20.2.10.	Output compare mode.....	399
20.2.11.	PWM mode.....	400
20.2.12.	Complementary outputs and dead-time insertion.....	401
20.2.13.	Using the break function	402
20.2.14.	One-pulse mode	404
20.2.15.	Timer and External Trigger Synchronization	405
20.2.16.	Timer synchronization	407
20.3.	TIM15 Interrupt.....	407
20.4.	TIM15 Debug Mode	408

20.5.	TIM15 registers	408
20.5.1.	TIM15 control register 1 (TIM15_CR1)	408
20.5.2.	TIM15 control register 2 (TIM15_CR2)	409
20.5.3.	TIM15 slave mode control register (TIM15_SMCR).....	410
20.5.4.	TIM15 DMA/interrupt enable register (TIM15_DIER).....	411
20.5.5.	TIM15 status register (TIM15_SR)	412
20.5.6.	TIM15 event generation register (TIM15_EGR).....	413
20.5.7.	TIM15 Capture/Compare mode register (TIM15_CCMR1).....	414
20.5.8.	TIM15 capture/compare enable register (TIM15_CCER)	416
20.5.9.	TIM15 counter register (TIM15_CNT)	418
20.5.10.	TIM15 prescaler register (TIM15_PSC)	418
20.5.11.	TIM15 auto-reload register (TIM15_ARR).....	418
20.5.12.	TIM15 repetition counter register (TIM15_RCR).....	419
20.5.13.	TIM15 capture/compare register 1 (TIM15_CCR1).....	419
20.5.14.	TIM15 capture/compare register 2 (TIM15_CCR2).....	420
20.5.15.	TIM15 brake and dead-time register (TIM15_BDTR).....	420
20.5.16.	TIM15 input selection register (TIM15_TISEL).....	421
20.5.17.	TIM15 alternate function option register 1 (TIM15_AF1)	422
20.5.18.	TIM15 DMA control register (TIM15_DCR)	423
20.5.19.	TIM15 DMA continuous transfer address register (TIM15_DMAR)	423
21.	General-purpose timer (TIM16/17)	424
21.1.	TIM16/TIM17 main features	424
21.2.	TIM16/TIM17 functional description	424
21.2.1.	Functional block diagram	424
21.2.2.	Time-base unit.....	424
21.2.3.	Counting mode (upcounting)	425
21.2.4.	Repetition counter	427
21.2.5.	Clock sources	428
21.2.6.	Capture/Compare channels	428
21.2.7.	Input capture mode:	429
21.2.8.	Forced Output Mode	430
21.2.9.	Output compare mode.....	430
21.2.10.	PWM mode	431
21.2.11.	Complementary outputs and dead-time insertion.....	432
21.2.12.	Using the break function	433
21.3.	TIM16/TIM17 interrupt.....	435
21.4.	TIM16/TIM17 debug mode	435
21.5.	TIM16/TIM17 registers	436
21.5.1.	TIM16/17 Control Register 1 (TIMx_CR1) (x=16,17)	436
21.5.2.	TIM16/17 Control Register 2 (TIMx_CR2) (x=16,17)	436

21.5.3.	TIM16/17 DMA/Interrupt Enable Register (TIMx_DIER) (x=16,17).....	437
21.5.4.	TIM16/17 Status Register (TIMx_SR) (x=16,17).....	437
21.5.5.	TIM16/17 Event Generation Register (TIMx_EGR) (x=16,17).....	438
21.5.6.	TIM16/17 Capture/Compare Mode Register 1 (TIMx_CCMR1) (x=16,17)	439
21.5.7.	TIM16/17 Capture/Compare Enable Register (TIMx_CCER) (x=16,17).....	441
21.5.8.	TIM16/17 counter register (TIMx_CNT) (x=16,17).....	442
21.5.9.	TIM16/17 prescaler register (TIMx_PSC) (x=16,17)	442
21.5.10.	TIM16/17 auto-reload register (TIMx_ARR) (x=16,17).....	443
21.5.11.	TIM16/17 Repetition Counter Register (TIMx_RCR) (x=16,17)	443
21.5.12.	TIM16/17 Capture/Compare Register 1 (TIMx_CCR1) (x=16,17)	443
21.5.13.	TIM16/17 brake and dead-time register (TIMx_BDTR) (x=16,17)	444
21.5.14.	TIM16/17 input selection register (TIMx_TISEL) (x=16,17)	446
21.5.15.	TIM16/17 alternate function option register 1 (TIMx_AF1) (x=16,17)	446
21.5.16.	TIM16/17 DMA control register (TIMx_DCR) (x=16,17).....	447
21.5.17.	TIM16/17 DMA continuous transfer address register (TIMx_DMAR) (x=16,17)	448
22.	Basic timers (TIM6/TIM7).....	449
22.1.	TIM6/TIM7 introduction	449
22.2.	TIM6/TIM7 Key Features	449
22.3.	TIM6/TIM7 Functional Description	449
22.3.1.	Functional block diagram	449
22.3.2.	Time-base unit.....	449
22.3.3.	Counting mode (upcounting).....	450
22.4.	TIM6/TIM7 Debug Mode	452
22.5.	TIM6/TIM7 Registers.....	453
22.5.1.	TIM6/TIM7 Control Register 1 (TIMx_CR1)	453
22.5.2.	TIM6/TIM7 Control Register 2 (TIMx_CR2)	453
22.5.3.	TIM6/TIM7 DMA/Interrupt Enable Register (TIMx_DIER).....	454
22.5.4.	TIM6/TIM7 Status Register (TIMx_SR).....	454
22.5.5.	TIM6/TIM7 Event Generation Register (TIMx_EGR)	455
22.5.6.	TIM6/TIM7 counter register (TIMx_CNT)	455
22.5.7.	TIM6/TIM7 prescaler register (TIMx_PSC)	455
22.5.8.	TIM6/TIM7 auto-reload register (TIMx_ARR).....	456
23.	Pulse Width Modulation (PWM).....	457
23.1.	PWM main features.....	457
23.2.	PWM functional description.....	457
23.2.1.	Functional block diagram	457
23.2.2.	Counting modes	457
23.2.3.	Clock selection	462
23.2.4.	Output compare mode.....	463

23.2.5.	PWM mode	464
23.2.6.	Complementary outputs and dead-time insertion	465
23.2.7.	Using the break function	466
23.3.	PWM interrupts	466
23.4.	PWM Debug Mode	467
23.5.	PWM registers	467
23.5.1.	PWM control register 1 (PWM_CR1)	467
23.5.2.	PWM control register 2 (PWM_CR2)	468
23.5.3.	PWM slave mode control register (PWM_SMCR)	469
23.5.4.	PWM DMA/Interrupt Enable Register (PWM_DIER)	470
23.5.5.	PWM status register (PWM_SR)	470
23.5.6.	PWM Event Generation Register (PWM_EGR)	472
23.5.7.	PWM output comparison mode register 1 (PWM_CMR)	472
23.5.8.	PWM output comparison enable register (PWM_CER)	473
23.5.9.	PWM counter register (PWM_CNT)	474
23.5.10.	PWM Prescaler (PWM_PSC)	474
23.5.11.	PWM Auto-reload Register (PWM_ARR)	474
23.5.12.	PWM Comparison Register 1 (PWM_CCR1)	474
23.5.13.	PWM comparison register 2 (PWM_CCR2)	475
23.5.14.	PWM comparison register 3 (PWM_CCR3)	475
23.5.15.	PWM comparison register 4 (PWM_CCR4)	475
23.5.16.	PWM Brake and Dead Time Register (PWM_BDTR)	476
23.5.17.	PWM DMA Control Register (PWM_DCR)	476
23.5.18.	DMA address for PWM continuous mode (PWM_DMAR)	477
24.	Infrared Interface (IRTIM)	479
25.	Low-Power Timer (LPTIM)	480
25.1.	Introduction to LPTIM	480
25.2.	LPTIM main features	480
25.3.	LPTIM functional description	481
25.3.1.	LPTIM block diagram	481
25.3.2.	LPTIM pin and interface description	481
25.3.3.	LPTIM reset and clocks	481
25.3.4.	Trigger signal glitch filter	481
25.3.5.	Prescaler	482
25.3.6.	Trigger selection	482
25.3.7.	Mode of operation	483
25.3.8.	Waveform generation	484
25.3.9.	Register update	485
25.3.10.	Counter mode	485

25.3.11.	Timer enable.....	485
25.3.12.	Timer reset.....	485
25.3.13.	Encoder Mode	486
25.3.14.	Debug mode	487
25.3.15.	Programming Guide	487
25.4.	LPTIM low-power modes	487
25.5.	LPTIM interrupts.....	487
25.6.	LPTIM registers	488
25.6.1.	LPTIM interrupt and status register (LPTIM_ISR).....	488
25.6.2.	LPTIM Interrupt Clear Register (LPTIM_ICR)	488
25.6.3.	LPTIM interrupt enable register (LPTIM_IER).....	489
25.6.4.	LPTIM configuration register (LPTIM_CFGR).....	489
25.6.5.	LPTIM control register (LPTIM_CR).....	491
25.6.6.	LPTIM comparison register (LPTIM_CMP)	492
25.6.7.	LPTIM Auto Reload Register (LPTIM_ARR).....	492
25.6.8.	LPTIM Counter Register (LPTIM_CNT)	493
25.6.9.	LPTIM Select Register (LPTIM_OR).....	493
26.	Independent watchdog (IWDG)	495
26.1.	Introduction to IWDG	495
26.2.	IWDG main features.....	495
26.3.	IWDG functional description.....	495
26.3.1.	IWDG block diagram	495
26.3.2.	Register access protection	495
26.3.3.	Debug mode	495
26.3.4.	Stop mode	495
26.4.	IWDG registers.....	496
26.4.1.	IWDG key register (IWDG_KR).....	496
26.4.2.	IWDG prescaler register (IWDG_PR)	496
26.4.3.	IWDG reload register (IWDG_RLR)	496
26.4.4.	IWDG status register (IWDG_SR).....	497
27.	System Window Watchdog (WWDG)	498
27.1.	Introduction to WWDG	498
27.2.	WWDG main features	498
27.3.	WWDG functional description	498
27.3.1.	WWDG Block Diagram	498
27.3.2.	Enable Watchdog	498
27.3.3.	Controls the countdown.....	499
27.3.4.	Watchdog Interrupt Advanced Feature	499
27.3.5.	How to Set the Watchdog Timeout.....	499

27.3.6.	Debug mode	499
27.4.	WWDG registers	500
27.4.1.	WWDG Control Register (WWDG_CR)	500
27.4.2.	WWDG Configuration Register (WWDG_CFR)	500
27.4.3.	WWDG Status Register (WWDG_SR)	500
28.	Real-Time Clock (RTC)	502
28.1.	Introduction to RTC	502
28.2.	RTC main features	502
28.3.	RTC functional description	503
28.3.1.	RTC block diagram	503
28.3.2.	RTC pins and internal signals	504
28.3.3.	RTC and TAMP controlled GPIO	504
28.3.4.	RTC clock and prescaler	504
28.3.5.	Real-time clock and calendar	505
28.3.6.	Programmable alarm	505
28.3.7.	Periodic auto-wakeup	505
28.3.8.	RTC Initialization and Configuration	506
28.3.9.	Reading the Calendar	507
28.3.10.	Resetting the RTC	508
28.3.11.	RTC Synchronization	508
28.3.12.	RTC Reference Clock Detection	508
28.3.13.	RTC Precision Digital Calibration	509
28.3.14.	Timestamp function	510
28.3.15.	Calibration clock output	511
28.3.16.	Tamper and alarm output	511
28.4.	RTC low-power modes	511
28.5.	RTC interrupts	512
28.6.	RTC registers	512
28.6.1.	RTC Time Register (RTC_TR)	512
28.6.2.	RTC Date Register (RTC_DR)	512
28.6.3.	RTC Subsecond Register (RTC_SSR)	513
28.6.4.	RTC Initialization Control and Status Register (RTC_ICSR)	513
28.6.5.	RTC prescaler register (RTC_PRER)	514
28.6.6.	RTC wakeup timer register (RTC_WUTR)	515
28.6.7.	RTC control register (RTC_CR)	515
28.6.8.	RTC Write Protection Register (RTC_WPR)	518
28.6.9.	RTC Calibration Register (RTC_CALR)	518
28.6.10.	RTC shift control register (RTC_SHIFTR)	518
28.6.11.	RTC timestamp time register (RTC_TSTR)	519
28.6.12.	RTC timestamp date register (RTC_TSDR)	519

28.6.13.	RTC timestamp sub-second register (RTC_TSSSR)	520
28.6.14.	RTC alarm A register (RTC_ALRMAR)	520
28.6.15.	RTC Alarm A Subsecond Register (RTC_ALRMASR)	521
28.6.16.	RTC Alarm B Register (RTC_ALRMBR)	521
28.6.17.	RTC Alarm B Sub-Second Register (RTC_ALRMBSSR)	522
28.6.18.	RTC status register (RTC_SR)	523
28.6.19.	RTC Masked Interrupt Status Register (RTC_MISR)	523
28.6.20.	RTC Status Clear Register (RTC_SCR)	524
29.	Tamper and backup registers (TAMP&BKP)	526
29.1.	TAMP introduction	526
29.2.	TAMP main features	526
29.3.	TAMP functional description	526
29.3.1.	TAMP block diagram	526
29.3.2.	TAMP pin and interface description	527
29.3.3.	TAMP register write protection	527
29.3.4.	Tamper detection	527
29.4.	Low-power modes	528
29.5.	TAMP interrupts	528
29.6.	TAMP registers	528
29.6.1.	TAMP control register 1 (TAMP_CR1)	528
29.6.2.	TAMP control register 2 (TAMP_CR2)	529
29.6.3.	TAMP filter control register (TAMP_FLTCR)	529
29.6.4.	Tamper Interrupt Enable Register (TAMP_IER)	530
29.6.5.	Tamper Status Register (TAMP_SR)	530
29.6.6.	Tamper Masked Interrupt Status Register (TAMP_MISR)	531
29.6.7.	TAMP status clear register (TAMP_SCR)	531
29.6.8.	TAMP backup x register (TAMP_BKPxR)	531
30.	Internal integrated circuit interface (I2C)	533
30.1.	I2C introduction	533
30.2.	I2C key features	533
30.3.	I2C functional description	534
30.3.1.	I2C block diagram	534
30.3.2.	Mode selection	534
30.3.3.	I2C initialization	535
30.3.4.	I2C slave mode	535
30.3.5.	I2C master mode	537
30.3.6.	Error condition	542
30.3.7.	SDA/SCL Line Control	542
30.3.8.	DMA request	543

30.3.9.	Packet Error Checking	544
30.3.10.	System Management Bus (SMBus)	544
30.4.	I2C interrupts	547
30.5.	I2C Debug Mode	547
30.6.	I2C Low Power	547
30.7.	I2C IO Configuration	547
30.8.	I2C registers	547
30.8.1.	I2C Control Register 1 (I2C_CR1).....	547
30.8.2.	I2C Control Register 2 (I2C_CR2).....	550
30.8.3.	I2C own address register 1 (I2C_OAR1)	551
30.8.4.	I2C own address register 2 (I2C_OAR2)	551
30.8.5.	I2C Data register (I2C_DR)	552
30.8.6.	I2C status register (I2C_SR1).	552
30.8.7.	I2C status register 2 (I2C_SR2)	555
30.8.8.	I2C clock control register (I2C_CCR)	556
30.8.9.	I2C TRISE register (I2C_TRISE).....	557
30.8.10.	I2C TIMEOUT Register (I2C_TIMEOUTR)	558
31.	Universal Synchronous/Asynchronous Receiver/Transmitter (USART)	559
31.1.	USART Introduction	559
31.2.	USART main features	559
31.3.	USART functional description	561
31.3.1.	USART block diagram	561
31.3.2.	USART signals	561
31.3.3.	USART Character Description	562
31.3.4.	USART transmitter	562
31.3.5.	USART Receiver	564
31.3.6.	USART baud rate generation	567
31.3.7.	USART reception tolerance.....	568
	OVR8=0	568
	OVR8=1	568
31.3.8.	USART Auto baud rate detection	568
31.3.9.	USART multiprocessor communications.....	569
31.3.10.	USART Parity Control.....	570
31.3.11.	USART synchronous mode	571
31.3.12.	USART single-wire half-duplex communication	572
31.3.13.	USART DMA Continuous Communication	573
31.3.14.	RS232 hardware flow control	575
31.3.15.	USART LIN mode	576
31.3.16.	USART Smart Card Mode	578
31.3.17.	USART IrDA SIR ENDEC function block	580

31.4.	USART interrupts	582
31.5.	Software Flow.....	582
31.5.1.	Smart Card Mode Operation Flow Using BLEN and EOBF Registers.....	582
31.6.	USART register	583
31.6.1.	USART Status Register (USART_SR)	583
31.6.2.	USART data register (USART_DR)	586
31.6.3.	USART baud rate register (USART_BRR).....	586
31.6.4.	USART control register 1 (USART_CR1)	587
31.6.5.	USART control register 2 (USART_CR2)	589
31.6.6.	USART control register 3 (USART_CR3)	590
31.6.7.	USART Guard Time and Prescaler Register (USART_GTPR).....	592
31.6.8.	USART receive timeout register (USART_RTOR).....	592
32.	Universal asynchronous receivers/transmitter (UART).....	594
32.1.	Introduction to UART	594
32.2.	UART main features.....	594
32.3.	UART functional description.....	594
32.3.1.	UART block diagram	594
32.3.2.	UART (RS232) serial protocol.....	595
32.3.3.	UART 9-bit data transmission	596
32.3.4.	UART baud rate.....	598
32.3.5.	UART reception tolerance	599
32.3.6.	UART DMA communications.....	599
32.4.	UART interrupts.....	600
32.5.	UART registers.....	601
32.5.1.	UART Data Register (UART_DR)	601
32.5.2.	UART baud rate register (UART_BRR)	601
32.5.3.	UART status register (UART_SR).....	602
32.5.4.	UART control register 1 (UART_CR1)	603
32.5.5.	UART control register 2 (UART_CR2)	605
32.5.6.	UART control register 3 (UART_CR3)	605
32.5.7.	UART receive address register (UART_RAR)	606
32.5.8.	UART transmit address register (UART_TAR)	607
32.5.9.	UART baud rate decimal register (UART_BRRF).....	607
33.	Low-power universal asynchronous receivers/transmitters (LPUART)	608
33.1.	LPUART main features	608
33.2.	LPUART functional description	609
33.2.1.	LPUART block diagram.....	609
33.2.2.	LPUART signals	609
33.2.3.	LPUART Character Description	609

33.2.4.	LPUART transmission	610
33.2.5.	LPUART reception	611
33.2.6.	LPUART baud rate generation	613
33.2.7.	LPUART reception tolerance.....	613
33.2.8.	LPUART multiprocessor communication	614
33.2.9.	LPUART parity.....	615
33.2.10.	LPUART single-line half-duplex communication	615
33.2.11.	DMA continuous communication	615
33.2.12.	RS232 hardware flow control and RS485 driver enablement	617
33.2.13.	LPUART low-power modes	618
33.3.	LPUART interrupts	619
33.4.	LPUART registers	620
33.4.1.	LPUART control register 1 (LPUART_CR1)	620
33.4.2.	LPUART control register 2 (LPUART_CR2)	622
33.4.3.	LPUART control register 3 (LPUART_CR3)	623
33.4.4.	LPUARTbaud rate register (LPUART_BRR).....	624
33.4.5.	LPUARTrequest register (LPUART_RQR)	625
33.4.6.	LPUARTinterrupt and status register (LPUART_ISR).....	625
33.4.7.	LPUARTInterrupt Flag Zero Register (LPUART_ICR).....	627
33.4.8.	LPUARTreceive data register (LPUART_RDR).....	627
33.4.9.	LPUARTSend Data Register (LPUART_TDR).....	627
33.4.10.	LPUARTPrescaler Register (LPUART_PRESC).....	628
34.	Serial peripheral interface / inter-IC sound (SPI/I2S)	629
34.1.	Introduction to SPI/I2S	629
34.2.	SPI/I2S Main Features	629
34.2.1.	SPI main features	629
34.2.2.	I2S Main Features	629
34.3.	SPI functional description.....	630
34.3.1.	SPI block diagram	630
34.3.2.	Communication between a master and a slave	630
34.3.3.	Standard multi-slave communication	632
34.3.4.	Multi-master communication	633
34.3.5.	Slave Selection Interface (NSS).....	634
34.3.6.	Communication format	635
34.3.7.	Configuring the SPI	636
34.3.8.	Enable SPI step.....	636
34.3.9.	Data transmission and reception.....	636
34.3.10.	Status flag.....	639
34.3.11.	Error flags	640
34.3.12.	TI mode.....	641

34.3.13.	SPI CRC	641
34.3.14.	SPI interrupts	642
34.4.	I2S functional description	643
34.4.1.	I2S block diagram	643
34.4.2.	Supports audio protocols	644
34.4.3.	Clock generation	651
34.4.4.	I2S transmission	653
34.4.5.	I2S logo	655
34.4.6.	I2S DMA	656
34.4.7.	I2S interrupts	656
34.5.	SPI/I2S Register	656
34.5.1.	SPI control register 1 (SPI_CR1)	656
34.5.2.	SPI control register 2 (SPI_CR2)	658
34.5.3.	SPI status register (SPI_SR)	659
34.5.4.	SPI Data Register (SPI_DR)	660
34.5.5.	SPI CRC polynomial register (SPI_CRCPR)	660
34.5.6.	SPI Receive CRC Register (SPI_RXCRC)	660
34.5.7.	SPI Send CRC Register (SPI_TXCRC)	661
34.5.8.	SPI_I2S configuration register (SPI_I2SCFGR)	661
34.5.9.	SPI_I2S Prescaler Register (SPI_I2SPR)	662
35.	Voltage reference buffer (V_{REFBUF})	664
35.1.	Introduction to V _{REFBUF}	664
35.2.	V _{REFBUF} register	664
35.2.1.	V _{REFBUF} Control Register (VREFBUF_CR)	664
36.	Debug support (DBG)	665
36.1.	Debug Introduction	665
36.2.	Pinout and Debug Port Pins	665
36.2.1.	SWD Port	665
36.2.2.	SW-DP Pin Division	665
36.2.3.	SWD Pin Pull-up/Pull-down	666
36.3.	Debugger ID	666
36.4.	SWD Port	666
36.4.1.	SWD protocol introduction	666
36.4.2.	SWD protocol sequence	666
36.4.3.	SW-DP State Machine (Reset, Idle State, ID Code)	667
36.4.4.	DP/AP Read-Write Access	667
36.4.5.	SW-DP Registers	667
36.4.6.	SW-AP registers	667
36.5.	Core debug registers	668

36.6.	BPU (Breakpoint Unit)	668
36.6.1.	BPU functionality	668
36.7.	DWT (Data Watchpoint)	668
36.7.1.	DWT functionality	668
36.7.2.	DWT Program Counter Sampling Register	668
36.8.	MCU Debug Component (DBGMCU)	668
36.8.1.	Low-power mode debugging support	669
36.8.2.	Timer/Watchdog/I2C Debugging	669
36.9.	DBGMCU registers	669
36.9.1.	DBG ID Register (DBG_IDCODE)	669
36.9.2.	Debug Configuration Register (DBG_CR)	669
36.9.3.	DBG APB Peripheral Freeze Register 1 (DBG_APB_FZ1)	670
36.9.4.	DBG APB Peripheral Freeze Register 2 (DBG_APB_FZ2)	671
37.	Revision history	672

1. Documentation conventions

1.1. List of abbreviations for registers

Abbreviation	Description
Read/Write (RW)	Software can read and write to this bit.
Read Only	Software can only read this bit.
Write Only	Software can only write to this bit.
Read/Clear by writing 0 (RC_W0)	Software can read this bit or clear it by writing 0; writing 1 has no effect on this bit.
Read/Clear by writing 1 (RC_W1)	Software can read this bit or clear it by writing 1; writing 0 has no effect on this bit.
Read/Write Clear (RC_W)	Software can read as well as clear this bit by writing to the register. The value written to this bit is not important.
Read/Read Clear (RC_R)	Software can read this bit. Reading this bit automatically clears it to 0. Writing this bit has no effect on the bit value.
Read/Read Set (RS_R)	Software can read this bit. Reading this bit automatically sets it to 1, writing this bit has no effect on its value.
Read/Set (RS)	Software can read this bit or set it to 1; writing 0 has no effect on this bit.
Switch (T)	The software can toggle this bit by writing 1. Writing 0 has no effect.
Reserved (Res.)	Reserved bit, must be kept at reset value

1.2. Availability of peripherals

For availability of peripherals and their number across all sales types, refer to the particular device datasheet.

3. Memory and bus architecture

3.1. System architecture

The system consists of:

- Two Masters
 - Cortex®-M0+ Core
 - General-purpose DMA
- Three Slaves
 - Internal SRAM
 - Internal Flash memory
 - AHB with AHB-APB bridge for connecting all APB peripherals

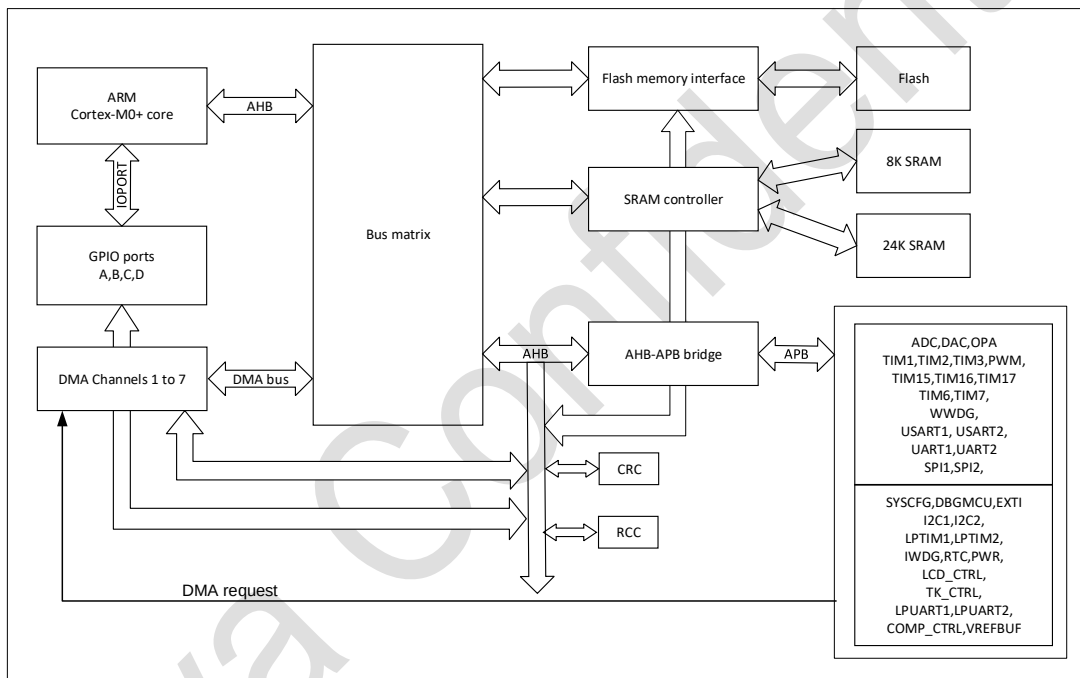


Figure 3-1 System Bus Architecture

■ AHB

This bus connects the system bus of the Cortex®-M0+ core to the bus matrix, which manages arbitration between the CPU core and DMA.

■ DMA bus

This bus connects the AHB master interface of the DMA to the bus matrix that manages the access of CPU and DMA to SRAM, Flash memory and AHB/APB peripherals.

■ Bus matrix

The bus matrix manages the access arbitration between the core system bus and the DMA master bus. The arbitration uses a Round Robin algorithm. The bus matrix is composed of masters (CPU, DMA) and slaves (Flash memory interface, SRAM and AHB-to-APB bridge).

■ AHB-APB Bridge (APB)

The AHB-APB bridge provides a fully synchronous connection between the AHB and APB buses.

After each chip reset, all peripheral clocks are turned off (except for SYSCFG, SRAM, and Flash interface). Before using a peripheral, its clock must be enabled in the RCC_AHBENR or RCC_APBENRx (x=1, 2) registers.

For AHB and APB peripheral address mapping information, refer to the "Register Map" section.

3.2. Memory organization

3.2.1. Introduction

Program memory, data memory, registers and I/O ports are organized within the same linear 4 GB address space. The bytes are coded in memory in Little Endian format (the lowest numbered byte in a word is considered the word's least significant byte).

The addressable memory space is divided into eight main blocks, of 512 Mbytes each.

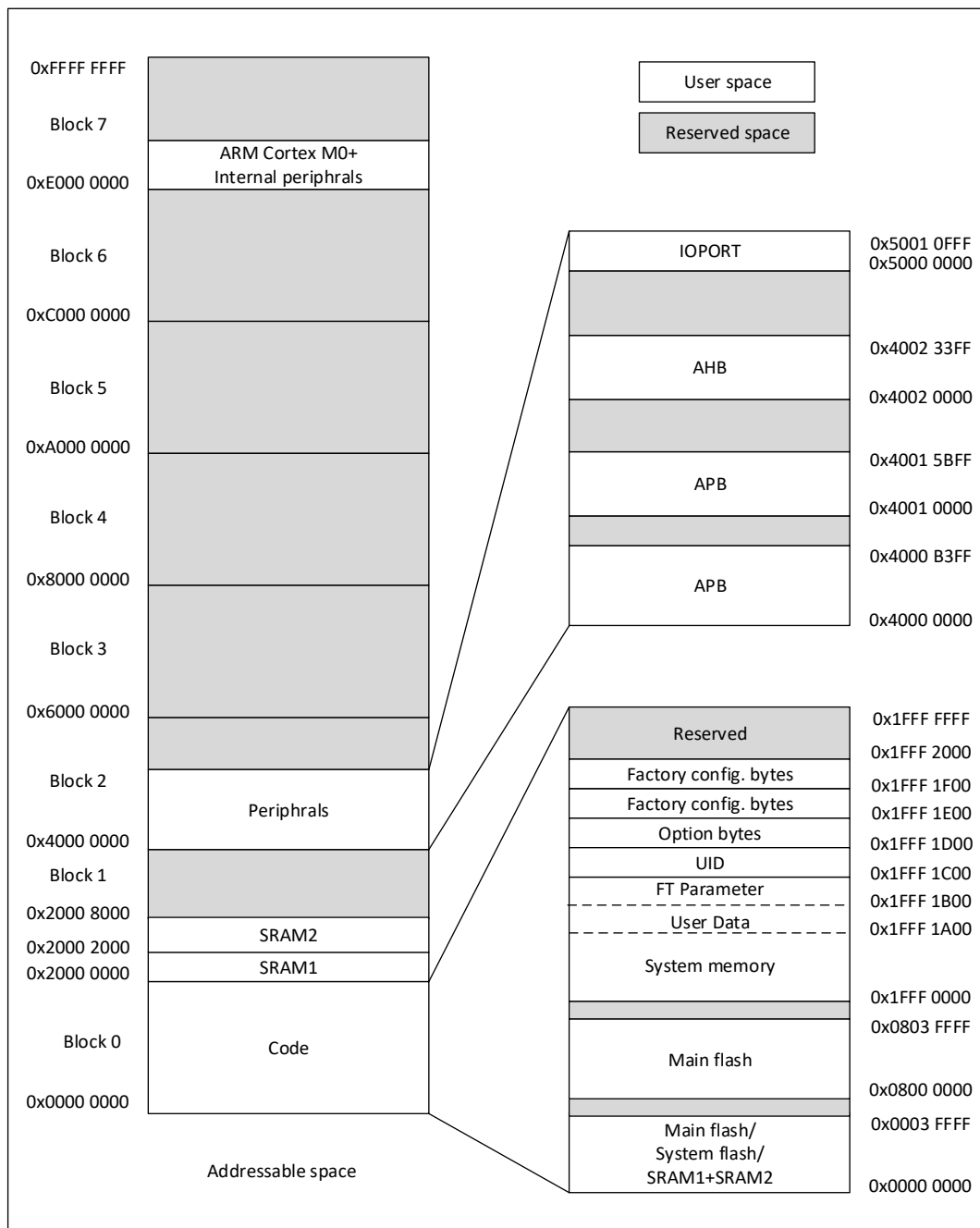


Figure 3-2 Memory map

3.2.2. Memory map

Table 3-1 Memory boundary address

Type	Boundary address	Size	Storage area	Description
SRAM	0x2000 8000-0x3FFF FFFF	~511 MB	Reserved	-
	0x2000 2000-0x2000 7FFF	24 KB	SRAM2	Refer to section 3.3 "Embedded SRAM"
	0x2000 0000-0x2000 1FFF	8 KB	SRAM1	
Code	0x1FFF 2000-0x1FFF FFFF	56 KB	Reserved	-
	0x1FFF 1E00-0x1FFF 1FFF	512 bytes	Factory configuration bytes	-
	0x1FFF 1D00-0x1FFF 1DFF	256 bytes	Option bytes	See 4.4 Flash option bytes
	0x1FFF 1C00-0x1FFF 1CFF	256 bytes	UID	Refer to section 4.3 Unique device ID (UID)

	0x1FFF 1B00-0x1FFF 1BFF	256 bytes	FT parameters	-
	0x1FFF 1A00-0x1FFF 1AFF	256 bytes	User data bytes	Refer to section 4.7 "Flash user data bytes"
	0x1FFF 0000-0x1FFF 19FF	6.5 KB	System memory	-
	0x0804 0000-0x1FFE FFFF	~393 MB	Reserved	-
	0x0800 0000-0x0803 FFFF	256 KB	Main flash	-
	0x0004 0000-0x07FF FFFF	~8 MB	Reserved	-
	0x0000 0000-0x0003 FFFF	256 KB	Depending on the Boot configuration selection, 1) Main flash 2) System memory 3) SRAM	-

3.2.3. Register map

Table 3-2 Peripheral register address

Bus	Boundary address	Size	Peripheral
	0xE000 0000-0xE00F FFFF	1 MB	Cortex®-M0+ core internal peripherals
IOPORT	0x5000 1800-0x5FFF FFFF	~256 MB	Reserved
	0x5000 1400-0x5000 17FF	1 KB	Reserved
	0x5000 1000-0x5000 13FF	1 KB	Reserved
	0x5000 0C00-0x5000 0FFF	1 KB	GPIOB
	0x5000 0800-0x5000 0BFF	1 KB	GPIOC
	0x5000 0400-0x5000 07FF	1 KB	GPIOA
	0x5000 0000-0x5000 03FF	1 KB	Reserved
AHB	0x4002 3400-0x4FFF FFFF	~	Reserved
	0x4002 3000-0x4002 33FF	1 KB	CRC
	0x4002 2400-0x4002 2FFF	~	Reserved
	0x4002 2000-0x4002 23FF	1 KB	Flash
	0x4002 1800-0x4002 1FFF	2 KB	Reserved
	0x4002 1400-0x4002 17FF	1 KB	Reserved
	0x4002 1000-0x4002 13FF	1 KB	RCC
	0x4002 0400-0x4002 0FFF	1 KB	Reserved
	0x4002 0000-0x4002 03FF	1 KB	DMA
APB	0x4001 5C00-0x4001 FFFF	32 KB	Reserved
	0x4001 5800-0x4001 5BFF	1 KB	MCUDBG
	0x4001 5000-0x4001 57FF	2 KB	Reserved
	0x4001 4C00-0x4001 4FFF	1 KB	PWM
	0x4001 4800-0x4001 4BFF	1 KB	TIM17
	0x4001 4400-0x4001 47FF	1 KB	TIM16
	0x4001 4000-0x4001 43FF	1 KB	TIM15
	0x4001 3C00-0x4001 3FFF	1 KB	Reserved
	0x4001 3800-0x4001 3BFF	1 KB	USART1
	0x4001 3400-0x4001 37FF	1 KB	Reserved
	0x4001 3000-0x4001 33FF	1 KB	SPI1

Bus	Boundary address	Size	Peripheral
	0x4001 2C00-0x4001 2FFF	1 KB	TIM1
	0x4001 2800-0x4001 2BFF	1 KB	TK
	0x4001 2400-0x4001 27FF	1 KB	ADC
	0x4001 0C00-0x4001 23FF	6 KB	Reserved
	0x4001 0800-0x4001 0BFF	1KB	VREFBUF
	0x4001 0400-0x4001 07FF	1 KB	EXTI
	0x4001 0300-0x4001 03FF	1 KB	OPA
	0x4001 0200-0x4001 02FF		COMP1/COMP2
	0x4001 0000-0x4001 01FF		SYSCFG
	0x4000 B400-0x4000 FFFF	19 KB	Reserved
	0x4000 B000-0x4000 B3FF	1 KB	BKP(TAMP)
	0x4000 9C00-0x4000 AFFF	5 KB	Reserved
	0x4000 9800-0x4000 9BFF	1 KB	LPUART2
	0x4000 9400-0x4000 97FF	1 KB	LPTIM2
	0x4000 8400-0x4000 93FF	4 KB	Reserved
	0x4000 8000-0x4000 83FF	1KB	LPUART1
	0x4000 7C00-0x4000 7FFF	1 KB	LPTIM1
	0x4000 7800-0x4000 7BFF	1 KB	Reserved
	0x4000 7400-0x4000 77FF	1 KB	DAC
	0x4000 7000-0x4000 73FF	1 KB	PWR
	0x4000 6400-0x4000 6FFF	1 KB	Reserved
	0x4000 5C00-0x4000 63FF	2 KB	Reserved
	0x4000 5800-0x4000 5BFF	1 KB	I2C2
	0x4000 5400-0x4000 57FF	1 KB	I2C1
	0x4000 5000-0x4000 53FF	1 KB	Reserved
	0x4000 4C00-0x4000 4FFF	1KB	UART2
	0x4000 4800-0x4000 4BFF	1KB	UART1
	0x4000 4400-0x4000 47FF	1 KB	USART2
	0x4000 3C00-0x4000 43FF	2 KB	Reserved
	0x4000 3800-0x4000 3BFF	1 KB	SPI2
	0x4000 3400-0x4000 37FF	1 KB	Reserved
	0x4000 3000-0x4000 33FF	1 KB	IWDG
	0x4000 2C00-0x4000 2FFF	1 KB	WWDG
	0x4000 2800-0x4000 2BFF	1 KB	RTC
	0x4000 2400-0x4000 27FF	1 KB	LCD
	0x4000 1800-0x4000 23FF	3 KB	Reserved
	0x4000 1400 - 0x4000 17FF	1 KB	TIM7
	0x4000 1000 - 0x4000 13FF	1 KB	TIM6
	0x4000 0800-0x4000 0FFF	2 KB	Reserved
	0x4000 0400 - 0x4000 07FF	1 KB	TIM3

Bus	Boundary address	Size	Peripheral
	0x4000 0000 - 0x4000 03FF	1 KB	TIM2

3.3. Embedded SRAM

The device integrates 32 KB of SRAM, divided into 8 KB SRAM1 and 24 KB SRAM2, with each byte containing 1 parity bit. SRAM can be accessed by byte (9 bits), half-word (18 bits), or full-word (36 bits).

Note: After power-on, the hardware does not perform a reset operation on the SRAM.

3.3.1. SRAM parity

The IEC60730 standard requires improved reliability of SRAM memory. Therefore, the SRAM in this product adds 1 parity bit for every 8 bits, resulting in a data width of 36 bits.

Users can enable parity check by using the option bit SRAM_PE in the Flash option byte page.

The SRAM parity error detection function appends a parity bit when writing data, checks the parity bit when reading data, and can generate a reset or NMI interrupt upon a parity error based on the configuration of SRAM_SCSR.PERR_RSTEN in SYSCFG. This error can also be connected to the system brake input of a timer.

3.3.2. SRAM Write Protection

SRAM can be write-protected with a granularity of 1KB pages.

Write protection can be enabled in the SRAM_WPR write protection register in SYSCFG. This is a register with a one-time "1" write mechanism, meaning that writing "1" once sets "write protection" for that page of SRAM, and this setting can only be cleared by a system reset.

Table 3-3 SRAM Write Protection Addresses

Page number	Field	Start address	End address
Page 0	SRAM1	0x0000 0000 / 0x2000 0000	0x0000 03FF / 0x2000 03FF
Page 1		0x0000 0400 / 0x2000 0400	0x0000 07FF / 0x2000 07FF
Page 2		0x0000 0800 / 0x2000 0800	0x0000 0BFF / 0x2000 0BFF
Page 3		0x0000 0C00 / 0x2000 0C00	0x0000 0FFF / 0x2000 0FFF
Page 4		0x0000 1000 / 0x2000 1000	0x0000 13FF / 0x2000 13FF
Page 5		0x0000 1400 / 0x2000 1400	0x0000 17FF / 0x2000 17FF
Page 6		0x0000 1800 / 0x2000 1800	0x0000 1BFF / 0x2000 1BFF
Page 7		0x0000 1C00 / 0x2000 1C00	0x0000 1FFF / 0x2000 1FFF
Page 8	SRAM2	0x0000 2000 / 0x2000 2000	0x0000 23FF / 0x2000 23FF
Page 9		0x0000 2400 / 0x2000 2400	0x0000 27FF / 0x2000 27FF
Page 10		0x0000 2800 / 0x2000 2800	0x0000 2BFF / 0x2000 2BFF
Page 11		0x0000 2C00 / 0x2000 2C00	0x0000 2FFF / 0x2000 2FFF
Page 12		0x0000 3000 / 0x2000 3000	0x0000 33FF / 0x2000 33FF
Page 13		0x0000 3400 / 0x2000 3400	0x0000 37FF / 0x2000 37FF
Page 14		0x0000 3800 / 0x2000 3800	0x0000 3BFF / 0x2000 3BFF

Page number	Field	Start address	End address
Page 15		0x0000 3C00 / 0x2000 3C00	0x0000 3FFF / 0x2000 3FFF
Page 16		0x0000 4000 / 0x2000 4000	0x0000 43FF / 0x2000 43FF
Page 17		0x0000 4400 / 0x2000 4400	0x0000 47FF / 0x2000 47FF
Page 18		0x0000 4800 / 0x2000 4800	0x0000 4BFF / 0x2000 4BFF
Page 19		0x0000 4C00 / 0x2000 4C00	0x0000 4FFF / 0x2000 4FFF
Page 20		0x0000 5000 / 0x2000 5000	0x0000 53FF / 0x2000 53FF
Page 21		0x0000 5400 / 0x2000 5400	0x0000 57FF / 0x2000 57FF
Page 22		0x0000 5800 / 0x2000 5800	0x0000 5BFF / 0x2000 5BFF
Page 23		0x0000 5C00 / 0x2000 5C00	0x0000 5FFF / 0x2000 5FFF
Page 24		0x0000 6000 / 0x2000 6000	0x0000 63FF / 0x2000 63FF
Page 25		0x0000 6400 / 0x2000 6400	0x0000 67FF / 0x2000 67FF
Page 26		0x0000 6800 / 0x2000 6800	0x0000 6BFF / 0x2000 6BFF
Page 27		0x0000 6C00 / 0x2000 6C00	0x0000 6FFF / 0x2000 6FFF
Page 28		0x0000 7000 / 0x2000 7000	0x0000 73FF / 0x2000 73FF
Page 29		0x0000 7400 / 0x2000 7400	0x0000 77FF / 0x2000 77FF
Page 30		0x0000 7800 / 0x2000 7800	0x0000 7BFF / 0x2000 7BFF
Page 31		0x0000 7C00 / 0x2000 7C00	0x0000 7FFF / 0x2000 7FFF

3.4. Flash memory

The Flash memory is composed of two distinct physical areas:

- Main memory area, 256 KB, divided into 2 Banks. Can be configured as 256 KB/192 KB/128 KB/64 KB for different products, used to store user programs and user data. When set to a capacity less than 256 KB, software access to the space outside the set range will cause a Hard-Fault.
- Information area, 8 KB, which includes the following parts:
 - Factory config: 512 bytes, used to store chip configuration information.
 - UID: 256 bytes, used to store the chip's UID.
 - Option bytes: 256 bytes, used to store configuration values for chip hardware and memory protection.
 - FT information area: 256 bytes, used to store FT parameters.
 - User data bytes: 256 bytes, user data area.
 - System memory: 6.5 KB, used to store the system bootloader.
 - The Flash interface implements instruction fetching and data access based on the AHB protocol, and it also provides basic Flash erase/program operations through registers.

3.5. Start Mode

Through the BOOT0 pin, the boot configuration bit nBOOT1 in the user option bytes, and the forced boot from Flash configuration bit BOOT_LOCK, the following different boot modes can be selected, as shown in the table below:

Table 3-4 Boot configuration

BOOT_LOCK	Boot mode configuration		Boot modes
	nBOOT1	bit BOOT0	
1	X	X	Forced boot from Main flash
0	X	0	Select the Flash main memory area as the boot area.
0	1	1	Select the Flash system memory area as the boot area.
0	0	1	Boot from SRAM

Users decide which boot mode to select according to the table above.

After reset, the CPU fetches the top-of-stack value from address 0x0000 0000, then starts executing instructions from address 0x0000 0004 of the boot memory. Depending on the selected boot mode, the main Flash memory, system memory, or SRAM is accessed as follows:

- Boot from main Flash memory: The main Flash memory is aligned with the boot memory space at 0x0000 0000, but can still be accessed at its original memory space (0x0800 0000). In other words, the Flash memory contents can be accessed starting from address 0x0000 0000 or 0x0800 0000.
- Boot from system Flash memory: The system Flash memory is aligned with the boot memory space at 0x0000 0000, but can still be accessed at its original address space 0x1FFF 0000.
- Boot from SRAM: The SRAM is aligned with the boot memory space at 0x0000 0000, but can still be accessed at address 0x2000 0000.
- Forced boot from user Flash: Regardless of other boot mode configuration bits, the BOOT_LOCK bit can be used to force booting from a single entry point in the main Flash.

3.5.1. Physical mapping

Once the boot mode is selected, the application software can modify the memory accessible in the code area. This modification is achieved by programming the MEM_MODE bit in the SYSCFG_CFGR1 register.

3.5.2. Embedded boot loader

The embedded bootloader is written during chip production and stored in the system Flash memory. It is used to reprogram the Flash memory using the following serial interfaces:

- USART1, corresponding to PB11/PB12 or PC8/PC9 or PD5/PD6

4. Embedded Flash

4.1. Introduction to Flash

The Flash interface manages CPU access to the Flash memory. This interface performs erase and program operations, as well as read/write protection and security mechanisms for the Flash.

The Flash interface accelerates code execution through instruction prefetch and cache mechanisms.

4.2. Flash main features

- Main memory: 256 KB (dual bank, each 16K x 64bit), used for storing user programs and user data.
- Information area: 8 KB (1K x 64bits)
- Page size: 256 bytes (32 x 64bits)
- Sector size: 8 KB (1K x 64bits)
- Flash only supports writing 0s, not 1s; erased values are 1.
- The minimum unit for Flash programming/erase operations is a page.

The main features of the Flash memory control interface circuit are as follows:

- Flash read operation with a read width of 72 bits (64 bits + 8 ECC bits)
- Flash programming/erase operations with a write width of 72 bits (64 bits + 8 ECC bits)
- Read protection activated by option bytes (RDP)
- Write protection area selected by option bytes (WRP)
- Proprietary code read protection area selected by option bytes (PCROP)
- Error correction code (ECC): 8 ECC bits per 64 bits
- Support: RWW (read while write)
 - Support reading Bank0 while writing Bank1, and reading Bank1 while writing Bank0
 - Simultaneous reading of two Banks is not supported
 - Simultaneous writing of two Banks is not supported
- Prefetch buffer
- Option byte loader

4.3. Flash Feature Description

4.3.1. Flash memory organization

The Flash memory consists of 64+8-bit wide cells with a dual-bank architecture supporting read-while-write (RWW) functionality, usable for program and data storage, with a page size of 256 bytes and a sector size of 8 KB.

Functionally, the Flash memory is divided into a main memory area and an information flash, with the former having a capacity of 256 KB and the latter 8 KB.

Page erase operations can be applied to both the Main flash and the information flash, while sector erase cannot be applied to the information area.

When write protection is not enabled, the main memory area of Flash can perform chip erase on Bank0 and Bank1 separately; the Flash information area does not support chip erase regardless of whether write protection is enabled.

Table 4-1 Flash memory area composition

region	Space			Address	Size
Main flash	Bank0	Sector 0	Page 0-31	0x0800 0000-0x0800 1FFF	8 KB
		Sector 1	Page 32-63	0x0800 2000-0x0800 3FFF	8 KB
		Sector 2	Page 64-95	0x0800 4000-0x0800 5FFF	8 KB
		...	...	...	...
		Sector 14	Pages 448-479	0x0801 C000-0x0801 DFFF	8 KB
		Sector 15	Pages 480-511	0x0801 E000-0x0801 FFFF	8 KB
	Bank1	Sector 0	Page 0-31	0x0802 0000-0x0802 1FFF	8 KB
		Sector 1	Page 32-63	0x0802 2000-0x0802 3FFF	8 KB
		Sector 2	Page 64-95	0x0802 4000-0x0802 5FFF	8 KB
		...	...	...	...
		Sector 14	Pages 448-479	0x0803 C000-0x0803 DFFF	8 KB
		Sector 15	Pages 480-511	0x0803 E000-0x0803 FFFF	8 KB
System Storage Area	Bank0	Sector 0	Pages 0-25	0x1FFF 0000-0x1FFF 19FF	6.5 KB
User Data Area (OTP)			Page 26	0x1FFF 1A00-0x1FFF 1AFF	256 bytes
FT			Page 27	0x1FFF 1B00-0x1FFF 1BFF	256 bytes
UID			Page 28	0x1FFF 1C00-0x1FFF 1CFF	256 bytes
Option bytes			Page 29	0x1FFF 1D00-0x1FFF 1DFF	256 bytes
Factory configuration 0			Page 30	0x1FFF 1E00-0x1FFF 1EFF	256 bytes
Factory configuration 1			Page 31	0x1FFF 1F00-0x1FFF 1FFF	256 bytes

4.3.2. Flash ECC

The data width in Flash is 72 bits: 8 ECC bits are added for every 64 bits.

The ECC mechanism supports:

- One-bit error detection and correction
- Two-bit error detection

When detecting and correcting an error, the ECCC (ECC Corrected) flag is set in the FLASH_ECCR register. If ECCCIE is set, an interrupt will be generated.

When two errors are detected, the ECCD (ECC Detected) flag is set in the FLASH_ECCR register. In this case, an NMI interrupt is generated.

When an ECC error is detected, the fault address and its related status are saved in ADDR_ECC, SYSF_ECC, and BK_ECC within the FLASH_ECCR register.

When ECCC or ECCD is set, if a new ECC error occurs, ADDR_ECC, SYSF_ECC, and BK_ECC are not updated. The FLASH_ECCR register is updated only after the ECC flags are cleared.

4.3.3. Flash read access latency

Flash can be directly addressed and accessed as a general-purpose memory space. Through the special read control timing, the contents of the Flash memory can be read.

The instruction fetch and data access are both done through the AHB bus. Read operations can be controlled by the LATENCY bit in the FLASH_ACR register, which adds either one or no wait states to Flash reads.

The FLASH_ACR.LATENCY bit: when 0, no wait states are added to Flash read operations; when 1, one wait state is added; when 2, two wait states are added. This mechanism is specially designed to match the high-speed system clock and the relatively low Flash read speed.

4.3.4. Flash acceleration

4.3.4.1. Instruction prefetch

Each Flash memory read operation provides 64 bits from either two instructions of 32 bits or four instructions of 16 bits according to the program launched. This 64-bit current instruction line is stored in the current buffer. So, in case of sequential code, at least two CPU cycles are needed to execute the previous read instruction line. When the CPU requests the current instruction line, the next contiguous instruction line in Flash can be read using the CPU system bus prefetch operation.

The prefetch feature can be enabled by setting the PRFTEN bit in the Flash Access Control Register (FLASH_ACR). This feature is useful if at least one wait state is needed to access the Flash memory. When the code is not sequential (branch), the instruction may not be present in the currently used instruction line or in the prefetched instruction line. In this case (miss), the penalty in terms of number of cycles is at least equal to the number of wait states.

If a loop is present in the current buffer, no new access is performed.

4.3.4.2. Cache memory

To limit the time lost due to jumps, it is possible to retain two cache lines of 64 bits (16 bytes) in the instruction cache memory. This feature can be enabled by setting the Instruction Cache Enable (ICEN) bit in the FLASH Access Control Register (FLASH_ACR). Each time a miss occurs (requested data not present in the currently used instruction line, in the prefetched instruction line or in the instruction cache memory), the line read is copied into the instruction cache memory. If some data contained in the instruction cache memory are requested by the CPU, they are provided without inserting any delay. Once all the instruction cache memory lines are filled, the LRU (least recently used) policy is used to determine the line to replace in the instruction memory cache. This feature is particularly useful in case of code containing loops.

The Instruction cache memory is enabled after system reset. During the CPU pipeline execution phase, the CPU data buffer pool in Flash will be accessed via the system bus. Each read access to the system bus will access the 64-bit data stored in the current buffer. The CPU pipeline is consequently stalled until the requested literal pool is provided.

4.3.5. Flash programming and erase operations

Flash can be programmed via In-Circuit Programming (ICP) or In-Application Programming (IAP).

ICP: Used to update the entire Flash memory content. The user application can be loaded into the MCU using the SWD protocol or interfaces supported by the system bootloader. ICP enables fast and efficient design iterations while eliminating unnecessary device packaging handling or socketing.

IAP: Allows downloading programming data to Flash using communication interfaces supported by the chip. IAP enables users to reprogram Flash memory while running the application. Nevertheless, part of the application has to have been previously programmed in the Flash memory using ICP. If a reset occurs during Flash programming or erase operations, the contents of the Flash memory are not guaranteed.

During programming or erase operations on one BANK, DMA or CPU can perform read accesses to another BANK. Code and data cannot be read from the BANK while programming or erase operations are in progress.

Note: For Flash programming and erase operations, HSI must be enabled (hardware automatically adjusts erase/write parameters based on HSI frequency).

4.3.5.1. Unlocking Flash operations

After reset, the Flash memory is protected against unwanted programming/erase operations (e.g., caused by electrical interference). After reset, writing to the FLASH_CR register is prohibited (except for the OBL_LAUNCH bit used to load option bytes). Each Flash erase or programming operation requires unlocking the FLASH_CR register by writing to FLASH_KEYR to initiate the unlock sequence.

This is done in the following steps:

Step 1: write KEY1=0x4567 0123 to FLASH_KEYR register

Step 2: write KEY2=0xCDEF 89AB to FLASH_KEYR register

Any incorrect timing sequence will lock the FLASH_CR register until the next system reset. If the key operation sequence is incorrect, a bus error occurs and triggers a HardFault interrupt. This is done after the first write cycle if KEY1 does not match, or during the second write cycle if KEY1 has been correctly written but KEY2 does not match.

The FLASH_CR register can be locked again by user software by writing the LOCK bit in the FLASH_CR register to 1.

Additionally, the FLASH_CR register cannot be written when the BSY bit in FLASH_SR register is set to 1. Any attempt to write to this register (FLASH_CR) during this time will cause AHB bus stall until the BSY bit is cleared.

4.3.5.2. Flash erase operations

Flash memory can be erased page-by-page, or through sector/bank mass erase (sector and BANK0 mass erase do not affect the Flash information area).

■ Page erase

When a page is protected by WRP, it cannot be erased, and the WRPERR bit is set. To perform a page erase operation, follow these steps:

1. Check that no main Flash memory operation is ongoing by checking the BSY bit in the FLASH_SR register.
2. Write KEY1 and KEY2 to the FLASH_KEYR register to unlock the protection of the FLASH_CR register
3. Set the PER bit and EOPIE bit (if EOP interrupt is required) in the FLASH_CR register

4. Write any data (must be 32-bit data) to the page
5. Wait for the BSY bit to be cleared.
6. Check that EOP flag bit is set
7. Clear EOP flag

■ Sector erase

Sector erase is used to erase an 8 KB area of the Flash main memory but does not affect the information area. In addition, when a sector is protected by WRP, it will not be erased, and the WRPERR bit is set at this time.

The steps for performing a sector erase are as follows:

1. Check the BSY bit to confirm if there are ongoing Flash operations
2. Write KEY1 and KEY2 to the FLASH_KEYR register to unlock the protection of the FLASH_CR register
3. Set the SER bit and EOPIE bit (if EOP interrupt is required) in the FLASH_CR register
4. Write any data (must be 32-bit data) to the sector
5. Wait for the BSY bit to be cleared.
6. Check that EOP flag bit is set
7. Clear EOP flag

■ Bank erase

Bank erase is used to erase a 128 KB bank area of the Flash main memory but does not affect the information area. Additionally, when WRP is enabled, the bank erase function is invalid, no bank erase operation occurs, and the WEPERR bit is set.

The steps for performing a bank erase are as follows:

1. Check the BSY bit to confirm if there are ongoing Flash operations
2. Write KEY1 and KEY2 to the FLASH_KEYR register to unlock the protection of the FLASH_CR register
3. Set the MER0 or MER1 bit and EOPIE bit (if EOP interrupt is required) in the FLASH_CR register
4. Write any data to any space in the corresponding Bank of the Flash main memory
5. Wait for the BSY bit to be cleared.
6. Check that EOP flag bit is set
7. Clear EOP flag

4.3.5.3. Flash main memory programming operation

An erase operation must be performed before a Flash programming operation. Refer to the "Flash erase operation" section for the erase operation steps.

The Flash memory performs programming operations on an entire page in 64-bit double-word units.

When the PG bit in the FLASH_CR register is set, and the CPU writes two 32-bit data to the Flash memory address space, which are combined into 64-bit by hardware, the programming operation starts. When the programming operation ends, the EOP bit in the FLASH_CR register is set.

Note: The CPU must operate in word units; performing half-word or byte operations will cause a HardFault!

If the Flash address space to be programmed is a protected area set by the FLASH_WRPFR register, the programming operation is ignored, and the WRPERR bit in the FLASH_SR register is set.

The detailed steps for Flash programming are as follows:

1. Check that no main Flash memory operation is ongoing by checking the BSY bit in the FLASH_SR register.
2. If there is no ongoing Flash erase or programming operation, the software reads 64 words of the Page (if the page already contains data and partial data needs to be updated, perform this step; otherwise, skip it)
3. Write KEY1 and KEY2 to the FLASH_KEYR register to unlock the protection of the FLASH_CR register
4. Set the PG bit and EOPIE bit (if EOP interrupt generation is required) in the FLASH_CR register; (if necessary, first clear the PGSTRT bit in the FLASH_CR register)
5. Perform programming operations for the 1st to 63rd words at the target address (only 32-bit programming is accepted)
6. Set the PGSTRT bit in the FLASH_CR register
7. Write the 64th word
8. Wait until the BSY bit is reset in the FLASH_SR register
9. Check the EOP flag in the FLASH_SR register (this bit is set when the programming operation is successful), then clear it by software
10. If no further programming operations are required, clear the PG bit by software

When the above step 7 is successfully executed, the programming operation automatically starts, and the BSY bit is set by hardware.

4.3.5.4. Flash erase/write time configuration

The programming and erase times of Flash must be strictly controlled; otherwise, the operation will fail. By default upon power-up, the hardware design sets the time parameters for programming and erase operations to those for HSI at 2 MHz. When the HSI frequency changes, the hardware will keep the clock used by the erase/write time logic at 2 MHz through frequency division.

4.4. Unique device ID (UID)

Typical application scenarios of unique identification code:

- Used as a serial number
- When programming internal flash memory, use it as a key or encryption primitive to improve code security
- To activate secure boot processes, etc.

The UID provides a reference number that is unique to any device.

The user can never change these bits. Unique identifiers can also be read in different ways, such as byte/half word/word, and then concatenated using custom algorithms.

The UID of this product is shown in the following table:

Base address: 0x1FFF 1C00

Table 4-2 UID identification code

Offset address	Description	UID Bits							
		7	6	5	4	3	2	1	0
0	Lot Numer	Lot Number ASCII code							
1	Lot Numer	Lot Number ASCII code							
2	Lot Numer	Lot Number ASCII code							
3	Lot Numer	Lot Number ASCII code							
4	Wafer Number	Wafer Number							
5	Lot Numer	Lot Number ASCII code							
6	Lot Numer	Lot Number ASCII code							
7	Lot Numer	Lot Number ASCII code							
8	Internal coding	Internal coding							
9	Y coordinate low position	Y coordinate low position							
10	X coordinate low position	X coordinate low position							
11	X, Y coordinate high address	Y coordinate high position				X coordinate high position			
12	CP pass ID	CP pass ID							
13	CRC8	CRC8 checksum of data at byte addresses 0x1FFF 1C00~0x1FFF 1C0C							
14	Internal coding	Internal coding							
15	Internal coding	Internal coding							

4.5. Flash option bytes

4.5.1. Description of Flash option bytes

A portion of the Flash information area within the chip is used as option bytes to store hardware configurations set by the chip or user for application requirements. For example, the watchdog can be configured in either hardware or software mode.

For data security, option bytes are stored separately in positive code and negative code form.

Table 4-3 Option byte format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Option byte 1 complement								Option byte 0 complement							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Option byte 1								Option byte 0							

The content of the option bytes can be read from the memory addresses described in the table "Option Byte Composition" or from the following related option byte registers:

- Flash option byte register (FLASH_OPTR)
- Flash write protection register (FLASH_WRPTR)
- Flash Bank0 proprietary code protection address register (FLASH_PCROP0)
- Flash Bank1 proprietary code protection address register (FLASH_PCROP1)

Table 4-4 Option Byte Composition

Word address	Description
0x1FFF 1D00	User option byte 0 and its complement
0x1FFF 1D04	User option byte 1 and its complement
0x1FFF 1D08	Reserved
0x1FFF 1D0C	Reserved

0x1FFF 1D10	BANK0 write protection (WRP) address option bytes and their complements
0x1FFF 1D14	BANK1 write protection (WRP) address option bytes and their complements
0x1FFF 1D18	BANK0 PCROP start address bytes and their complements
0x1FFF 1D1C	BANK0 PCROP end address bytes and their complements
0x1FFF 1D20	BANK1 PCROP start address bytes and their complements
0x1FFF 1D24	BANK1 PCROP end address bytes and their complements
...	Reserved
0x1FFF 1DFC	Reserved

4.5.2. User option byte 0

Flash address: 0x1FFF 1D00

Factory default: 0x4055 BFAA

After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding values are read from the option byte area in the Flash information sector and written to the corresponding option byte bits in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~nBOOT1	~NRST_MODE	~WWDG_SW	~IWDG_SW	~IWDG_STOP	~IWDG_STDBY	~NRST_STOP	~NRST_STDBY	~RDP[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
nBOOT1	NRST_MODE	WWDG_SW	IWDG_SW	IWDG_STOP	IWDG_STDBY	NRST_STOP	NRST_STDBY	RDP[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31	~nBOOT1	R	Complemented code of nBOOT1
30	~NRST_MODE	R	Complemented code of NRST_MODE
29	~WWDG_SW	R	Complemented code of WWDG_SW
28	~IWDG_SW	R	Complemented code of IWDG_SW
27	~IWDG_STOP	R	Complemented code of IWDG_STOP
26	~IWDG_STDBY	R	Complement of IWDG_STDBY
25	~NRST_STOP	R	Complement of NRST_STOP
24	~NRST_STDBY	R	Inverse code of NRST_STDBY
23:16	~RDP	R	Complemented code of RDP
15	nBOOT1	R	Together with the BOOT PIN, select the boot mode For detailed description, refer to section 3.5 "Boot Mode"
14	NRST_MODE	R	0: Reset input only 1: GPIO function
13	WWDG_SW	R	0: Hardware window watchdog 1: Software window watchdog
12	IWDG_SW	R	0: Hardware independent watchdog 1: Software independent watchdog
11	IWDG_STOP	R	Stop mode IWDG count control 0: IWDG counting stops in stop mode 1: IWDG counting operates normally in Stop mode
10	IWDG_STDBY	R	IWDG counting control in Standby mode 0: IWDG counting stops in Standby mode 1: IWDG counting operates normally in Standby mode
9	NRST_STOP	R	0: Generate reset when entering Stop mode 1: Do not generate reset when entering Stop mode
8	NRST_STDBY	R	0: Generate reset when entering Standby mode 1: Do not generate reset when entering Standby mode
7:0	RDP	R	Read protection level. 0xAA: Level 0, read protection not activated Non-0xAA: Level 1, chip read protection activated

4.5.3. User option byte 1

Flash address: 0x1FFF 1D04

Factory default: 0xFDED 0212

After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding values are read from the option byte area in the Flash information sector and written to the corresponding option byte bits in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	ECC_EN	~ BOOT_Lock	~BFB	~SWD_MODE	~SRAM_PE	Res	Res	Res	Res	Res.
						R	R	R	R	R	R				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	ECC_EN	BOOT_LOCK	BFB	SWD_MODE	SRAM_PE	BOR_LEV[2:0]				BOR_EN
						R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function																									
31:26	Reserved	R	Complemented code of Bit [15:10]																									
25	~ECC_EN	R	Inverse of ECC_EN																									
24	~BOOT_Lock	R	Inverse of BOOT_LOCK																									
23	~BFB	R	Complemented code of BFB																									
22:21	~SWD_MODE	R	Complemented code of SWD_MODE																									
20	~SRAM_PE	R	Inverse of SRAM_PE																									
19:10	Reserved	R																										
9	ECC_EN	R	ECC enabled 0: Disable ECC 1: Enable ECC																									
8	BOOT_LOCK	R	Use to force boot from user Flash zone 0: Boot based on IO pin/option bit configuration 1: Force boot from Flash main memory area																									
7	BFB	R	0: Boot from Bank0 (Flash Bank0 mapped to 0x0800_0000/0x0000_0000, Flash Bank1 mapped to 0x0802_0000/0x0002_0000) 1: Boot from Bank1 (Flash Bank1 mapped to 0x0800_0000/0x0000_0000, Flash Bank0 mapped to 0x0802_0000/0x0002_0000)																									
6:5	SWD_MODE[1:0]	R	SWD PIN mode selection. <table><tr><td>SWD_MODE[1:0]</td><td>PA4</td><td>PA5</td><td>PB6</td><td>PB5</td></tr><tr><td>00(default)</td><td>SWDIO</td><td>SWCLK</td><td>GPIO</td><td>GPIO</td></tr><tr><td>01</td><td>GPIO</td><td>GPIO</td><td>SWDIO</td><td>SWCLK</td></tr><tr><td>10</td><td>GPIO</td><td>SWCLK</td><td>SWDIO</td><td>GPIO</td></tr><tr><td>11</td><td>SWDIO</td><td>GPIO</td><td>GPIO</td><td>SWCLK</td></tr></table> When PA5 and PB5 are configured as SWCLK via SWD_MODE, they are set to pull-down mode; When PA4 and PB6 are configured as SWDIO via SWD_MODE, they are set to pull-up mode;	SWD_MODE[1:0]	PA4	PA5	PB6	PB5	00(default)	SWDIO	SWCLK	GPIO	GPIO	01	GPIO	GPIO	SWDIO	SWCLK	10	GPIO	SWCLK	SWDIO	GPIO	11	SWDIO	GPIO	GPIO	SWCLK
SWD_MODE[1:0]	PA4	PA5	PB6	PB5																								
00(default)	SWDIO	SWCLK	GPIO	GPIO																								
01	GPIO	GPIO	SWDIO	SWCLK																								
10	GPIO	SWCLK	SWDIO	GPIO																								
11	SWDIO	GPIO	GPIO	SWCLK																								
4	SRAM_PE	R	SRAM parity check enable 0: Disable SRAM parity check 1: Enable SRAM parity check																									
3:1	BOR_LEV[2:0]	R	000: BOR rising threshold is 1.8V and falling threshold is 1.7V 001: BOR rising threshold is 2.0V and falling threshold is 1.9V 010: BOR rising threshold is 2.2V and falling threshold is 2.1V 011: BOR rising threshold is 2.4V and falling threshold is 2.3V 100: BOR rising threshold is 2.8V and falling threshold is 2.7V 101: BOR rising threshold is 3.1V and falling threshold is 3.0V 110: BOR rising threshold is 3.7V and falling threshold is 3.6V 111: BOR rising threshold is 4.2V and falling threshold is 4.1V																									
0	BOR_EN	R	BOR enabled 0: BOR disabled 1: BOR enabled, BOR_LEV works																									

4.5.4. Flash protection configuration 2 (BANK0_WRP)

Flash address: 0x1FFF 1D10

Factory value: 0x0000 FFFF

After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area in the Flash information area and written to the corresponding option byte bit in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~ BANK0_WRP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BANK0_WRP [15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	~ BANK0_WRP	R	Complemented code of BANK0_WRP
15:0	BANK0_WRP	R	0: Sector[y] is protected 1: Sector[y] is unprotected y=0~15

4.5.5. Flash protection configuration 3 (BANK1_WRP)

Flash Address: 0x1FFF 1D14

Factory value: 0x0000 FFFF

After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding values are read from the option byte area in the Flash information sector and written to the corresponding option byte bits in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~ BANK1_WRP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BANK1_WRP [15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	~ BANK1_WRP	R	Complemented code of BANK1_WRP
15:0	BANK1_WRP	R	0: Sector[y] is protected 1: Sector[y] is unprotected y=0~15

4.5.6. Flash Protection Configuration 4 (PCROP0SR)

Flash Address: 0x1FFF 1D18

Factory Default: 0xFE00 01FF

After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area in the Flash information area and written to the corresponding option byte bit in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
7'h7F							~ PCROP0SR [8: 0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
7'h0							PCROP0SR [8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:25	Reserved	R	7'h7F
24:16	~ PCROP0SR	R	Complemented code of PCROP0SR
15:9	Reserved	R	7'h0
8:0	PCROP0SR	R	BANK0 Proprietary Code Protection Start Address (in pages)

4.5.7. Flash Protection Configuration 5 (PCROP0ER)

Flash Address: 0x1FFF 1D1C

Factory Default: 0xFFFF 0000

After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding values are read from the option byte area in the Flash information sector and written to the corresponding option byte bits in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
7'h7F							~ PCROP0ER [8: 0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
7'h0							PCROP0ER [8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:25	Reserved	R	7'h7F
24:16	~ PCROP0ER	R	Complemented code of PCROP0ER
15:9	Reserved	R	7'h0
8:0	PCROP0ER	R	BANK0 Proprietary Code Protection End Address (in pages)

4.5.8. Flash Protection Configuration 6 (PCROP1SR)

Flash Address: 0x1FFF 1D20

Factory Default: 0xFE00 01FF

After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding values are read from the option byte area in the Flash information sector and written to the corresponding option byte bits in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
7'h7F							~ PCROP1SR [8: 0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
7'h0							PCROP1SR [8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	Reserved	R	7'h7F
24:16	~ PCROP1SR	R	Complemented code of PCROP1SR
15:8	Reserved	R	7'h0
8:0	PCROP1SR	R	BANK1 Proprietary Code Protection Start Address

4.5.9. Flash Protection Configuration 7 (PCROP1ER)

Flash Address: 0x1FFF 1D24

Factory Default: 0xFFFF 0000

After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding values are read from the option byte area in the Flash information sector and written to the corresponding option byte bits in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
7'h7F							~ PCROP1ER [8: 0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
7'h0							PCROP1ER [8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	Reserved	R	7'h7F
24:16	~ PCROP1ER	R	Complemented code of PCROP1ER
15:8	Reserved	R	7'h0
8:0	PCROP1ER	R	BANK1 Proprietary Code Protection End Address (in Page units)

4.5.10. Flash Option Byte Programming

After reset, the bits associated with the option bytes in the FLASH_CR register are write-protected. Before performing any operation on the option bytes, the OPTLOCK bit in the FLASH_CR register must be cleared (Note: This operation is required for each erase or program operation).

The following steps are used to unlock the register:

1. Unlock the write protection of the FLASH_CR register through the unlock sequence (refer to Section 4.3.5.1 "Unlocking Flash Operations").
2. Write OPTKEY1 = 0x0819 2A3B to the FLASH_OPTKEYR register
3. Write OPTKEY2 = 0x4C5D 6E7F to the FLASH_OPTKEYR register

Any incorrect sequence will lock the FLASH_CR register until the next reset. If the key operation sequence is incorrect, a bus error state will occur, triggering a HardFault interrupt.

The option bytes in the Flash information area can be locked by software setting the OPTLOCK bit in the FLASH_CR register to 1, to prevent unwanted erase/program operations.

If the software sets the FLASH_CR.LOCK bit, the OPTLOCK bit is also automatically set.

4.5.10.1. Steps to Modify User Option Bytes

The programming operation for option bytes differs from that for the main Flash memory area. To modify the option bytes, the following steps are needed:

1. Clear the OPTLOCK bit using the previously described steps
2. Check the BSY bit to confirm that there are no ongoing Flash operations
3. Write the desired value (1~4 words) to the option byte registers FLASH_OPTR/FLASH_WRPR/FLASH_PCROPR0/FLASH_PCROPR1 (if not word-aligned, the operation is ignored).
4. Set OPTSTRT bit
5. Write any 32-bit data to 0x1FFF_1D00 (to trigger the formal write operation) (if not word-aligned, a HardFault will occur; writing to a non-0x1FFF_1D00 address will result in an error and the operation will not execute properly).
6. Wait for the BSY bit to be cleared.
7. Wait for the EOP to rise, the software clears

Any modification to the option bytes will first cause the hardware to erase the entire corresponding page of the option bytes, then write the values from the FLASH_OPTR, FLASH_WRPR, or FLASH_PCROPRx (x=0, 1) registers to the option bytes. The hardware automatically calculates the corresponding complement code and writes the calculated value to the corresponding area of the option byte.

4.5.10.2. Option byte loading

After the BSY bit is cleared, all new option bytes are written to the Flash information storage area but are not yet applied to the chip system. Reading the option byte registers will still return the values from the previously loaded option bytes. The new values take effect on the chip system only after they are loaded.

Option byte loading occurs under the following two conditions:

- When the OBL_LAUNCH bit in the FLASH_CR register is set
- After power-on reset (POR/BOR)
- Exiting standby mode with PWR_CR3.LD_EN configured to 1

The "Load Option Bytes" operation involves reading the option byte page in the Flash information storage area and storing the read data in the internal option byte registers (FLASH_OPTR, FLASH_WRP, and FLASH_PCROPx). These internal registers configure the system and can be read by software. Set the OBL_LAUNCH bit to generate a reset, and perform option byte loading under the system reset.

Each option byte bit has its corresponding complement at the same double-word address (next half-word). During option byte loading, the option bits and their complements are verified to ensure correct loading.

If the positive and complement bits match, the option bytes are copied to the option byte registers.

If the word and its complement are not matching, the OPTVERR status bit of the FLASH_SR register is set. The option byte registers are written with values defined as follows:

- For user options
 - The BOR_EN bit is written as 0 (BOR disabled)
 - The NRST_MODE bit is written as 0 (reset input only)
 - RDP bit is written as 0xFF (i.e., Level 1 read protection)
 - BOOT_LOCK bit is written as 0
 - BFB is written as 0
 - SWD_MODE is written as 0
 - The rest of the mismatched values are written as 1
- For PCROP options, PCROPxSR = 0x000, PCROPxER = 0x1FF, meaning all Flash space is set as protected
- For WRP options, mismatched values default to "no write protection"

Upon system reset, the option bytes are copied into the following option registers that can be read and written by software:

- FLASH_OPTR
- FLASH_PCROPx
- FLASH_WRP

These registers are also used to modify option bytes. If these registers are not modified by the user, they reflect the system's option status.

4.6. Flash factory configuration bytes

A portion of the Flash information area inside the chip (totaling 2 pages) is used as factory configuration bytes.

Factory configuration byte 0 stores information for software reading (only positive code, no complement stored):

- HSI frequency selection value and corresponding Trimming value
- Configuration parameter values for HSI 2MHz erase/write time

4.6.1. HSI_TRIMMING_FOR_USER

Flash address: 0x1FFF 1E04~0x1FFF 1E18

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSI_FS[2:0]			HSI_TRIM[12:0]												
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software needs to read the data from this address and then write it to HSI_FS [2: 0] and HSI_TRIM [12: 0] corresponding to the RCC_ICSCR register to change the HSI frequency.

4.6.2. NORMAL_TS_CALIBRATION_PARA

Flash address: 0x1FFF 1E20

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	NTSCAL[11:0]											
				R	R	R	R	R	R	R	R	R	R	R	R

Software needs to read data from this address to obtain normal temperature calibration parameters.

4.6.3. HIGH_TS_CALIBRATION_PARA

Flash address: 0x1FFF 1E24

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	HTSCAL[11:0]											
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software needs to read data from this address to obtain high-temperature calibration parameters.

4.6.4. HSI2M_EPPARA0

Flash address: 0x1FFF 1E28 (2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	TS1 [10:0]										
					R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.				TS0 [8:0]											
							R	R	R	R	R	R	R	R	R

The software reads the corresponding HSI 2MHz data from this address and writes it into the FLASH_TS0 and FLASH_TS1 registers to configure the erase/write time required for the corresponding HSI frequency.

4.6.5. HSI2M_EPPARA1

Flash address: 0x1FFF 1E2C (2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS3 [8:0]								
							R	R	R	R	R	R	R	R	R

The software reads the corresponding HSI2MHz data from this address and writes it into the FLASH_TS3 register to configure the erase/write time required for the corresponding HSI frequency.

4.6.6. HSI2M_EPPARA2

Flash address: 0x1FFF 1E30 (2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	TPS3 [12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS2P [8:0]								
							R	R	R	R	R	R	R	R	R

The software reads the corresponding HSI2MHz data from this address and writes it into the FLASH_TPS3 and FLASH_TS2P registers to configure the erase/write time required for the corresponding HSI frequency.

4.6.7. HSI2M_EPPARA3

Flash address: 0x1FFF 1E34 (2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERTPE[17:16]	
														R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PERTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software reads the corresponding HSI2MHz data from this address and writes it into the FLASH_PERTPE register to configure the erase/write time required for the corresponding HSI frequency.

4.6.8. HSI2M_EPPARA4

Flash address: 0x1FFF 1E38 (2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMERTPE[17:16]	
														R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMERTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software reads the corresponding HSI2MHz data from this address and writes it into the FLASH_SMERTPE register to configure the erase/write time required for the corresponding HSI frequency.

4.6.9. HSI2M_EPPARA5

Flash address: 0x1FFF 1E3C (2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRGTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software reads the corresponding HSI2MHz data from this address and writes it into the FLASH_PRGTPE register to configure the erase/write time required for the corresponding HSI frequency.

4.6.10. HSI2M_EPPARA6

Flash address: 0x1FFF 1E40 (2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PRETPE[13:0]													
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software reads the corresponding HSI2MHz data from this address and writes it into the FLASH_PRETPE register to configure the erase/write time required for the corresponding HSI frequency.

4.7. Flash user data bytes

4.7.1. Flash user data description

The Flash information area contains one page as the user data area. Programming and erasing of this page are handled using the same methods as the Flash main memory area. Additionally, chip erase operations in the Flash main memory area are ineffective for this region.

Setting the USER OTP MEMORY LOCK content does not update immediately; it takes effect after a power-on reset (POR/BOR) load, thereby providing protection.

Table 4-5 Flash User Data is used to store user data

Page	Address	Contents	
26	0x1FFF 1A00	Bit[31:16]: store user data Bit [15: 0]: USER OTP MEMORY LOCK	
		USER OTP MEMORY LOCK	User Data protection
		0xAA55	Read: Yes Programming and erase operations: Disabled
		Other values	Read, program, and erase operations: allowed
	0x1FFF 1A04	Store user data	
	0x1FFF 1A08	Store user data	
	...	Store user data	
	...	Store user data	
	...	Store user data	
	0x1FFF 1AFC	Store user data	

4.7.2. Flash user data programming

After power-on reset, the programmability of the user data area is determined by the loaded USER OTP MEMORY LOCK value.

If programming is allowed, follow these steps:

Modify user data

Programming operations in the user data area are the same as those for the Flash main memory.

The following steps are required:

- 1) Check the BSY bit to confirm that there are no ongoing Flash operations
- 2) If there is no ongoing Flash erase or program operation, the software reads the 64 words of the Page.
- 3) Write KEY1 and KEY2 to the FLASH_KEYR register to unlock the protection of the FLASH_CR register
- 4) Set the UPG bit and EOPIE bit (if EOP interrupt is needed) in the FLASH_CR register.
- 5) Program the 1st to 63rd words to the User data area address (only 32-bit programming is accepted).
- 6) Set the UPGSTRT bit in the FLASH_CR register.
- 7) Write the 64th word
- 8) Wait for the BSY bit to be cleared.
- 9) Wait for the EOP to rise, the software clears

When step 7) is successfully executed, the programming operation automatically starts, and the BSY bit is set by hardware.

4.8. Flash storage protection

Protection mechanisms for the Flash main memory include:

- Read protection (RDP), preventing external access to the Flash main memory.
- Write protection (WRP) prevents unintended writes (caused by confusion of program). The size of write protection is designed to be 8 KB.
- Option byte write protection is a special design for unlock.
- Proprietary code readout protection (PCROP). The granularity of PCROP protection is designed as 256 bytes.

4.8.1. Proprietary code readout protection (PCROP)

Through configuration, certain pages of the Flash main memory can prevent unauthorized read/write by third parties.

The protected area is an execute-only region: it can only be accessed by the CPU as instruction code, while all other accesses (DMA, debugging, CPU data read, write, and erase) are strictly prohibited. When RDP protection changes from level 1 to level 0, the PCROP area is erased.

Each PCROP area is defined by a start page and an end page address associated with a physical flash address. These addresses are defined in the PCROP address registers FLASH_PCROPSR, FLASH_PCROP1SR, FLASH_PCROP0ER, FLASH_PCROP1ER. FLASH_PCROP0SR and FLASH_PCROP0ER are for Bank0, FLASH_PCROP1SR and FLASH_PCROP1ER are for Bank1.

The PCROP area range is defined by the address:

Flash base address + [PCROPxSR x 0x100] (inclusive)

To address:

Flash base address + [(PCROPxER + 1) x 0x100] (exclusive).

When PCROPxER = PCROPxSR, the page configured by PCROPxSR (or PCROPxER) is protected by PCROP.

For example:

To PCROP-protect the address range 0x0800 0800 to 0x0800 13FF, when BFB=0, configure the registers as follows:

- PCROP0SR = 0x08 (PCROP start address 0x0800 0800)
- PCROP0ER = 0x13 (PCROP end address 0x0800 13FF)

To PCROP-protect the address range 0x0800 0800 to 0x0800 13FF, when BFB=1, configure the registers as follows:

- PCROP1SR = 0x08 (PCROP start address 0x0800 0800)
- PCROP1ER = 0x13 (PCROP end address 0x0800 13FF)

Any read data access to the PCROP-protected area triggers an RDERR flag error.

Any PCROP-protected address is also write-protected, and any write access to one of these addresses triggers WRPERR.

Any PCROP areas are also erase protected. Therefore, any erasure of pages in this area is not possible (including pages containing the start and end addresses of this area). Additionally, if an area is PCROP-protected, software bank erase cannot be executed.

Users can modify the PCROP area through option bytes in the information block. PCROP is disabled when RDP changes from level 1 to level 0; for other RDP changes (remaining unchanged or changing from level 0 to level 1), only expanding the PCROP area is allowed—shrinking the PCROP area results in the PCROP area remaining unchanged.

Table 4-6 PCROP Protection

PCROPx register value (x = 0, 1)	PCROP Protected Area
PCROPxSR > PCROPxER	No PCROP area.
PCROPxSR < PCROPxER	The region between PCROPxSR and PCROPxER is protected.
PCROPxSR = PCROPxER	The page configured by PCROPxSR (or PCROPxER) is protected.

4.8.2. Flash read protection

The read protection function can be activated by setting the RDP option byte and performing a POR/BOR or OBL reset to load the new RDP option byte. RDP protects the Flash main memory area and backup domain registers.

If the debugger is still connected via SWD when setting read protection, a power-on reset is required instead of other system resets.

When the RDP option byte and its complement exist correctly in pairs in the option bytes, the Flash main memory area is protected.

Table 4-7 Flash memory read protection status

RDP byte value	RDP complement byte value	Read protection level
0xAA	0x55	Level 0
Any value except (0xAA and 0x55)		Level 1

Level 0: No protection

Read, program, and erase operations can be performed on the Flash main memory area, and any operations can be performed on the option bytes and backup registers.

Level 1: Read protection

When the RDP and its complement in the option bytes contain any combination other than [0xAA, 0x55], Level 1 read protection takes effect.

- User mode: Programs executed in user mode (booted from the Flash main memory area) can perform all operations on the Flash main memory area, option bytes, and backup registers.
- Debug, boot from SRAM, and boot from system memory modes: In debug mode, or when booting from SRAM or the system memory area (system bootloader), the Flash main memory area and backup registers cannot be accessed for data reading. In these modes, read or write access to the Flash main memory area generates a bus error and triggers a HardFault interrupt.

When already at Level 1 (any value other than 0xAA), if modifying to Level 0 (writing 0xAA), the hardware will perform a mass erase operation on the Flash main memory area (bank0 and bank1 mass erase respectively).

4.8.3. Flash write protection (WRP)

Flash can be set to write-protected in response to unwanted write operations. Each bit in the WRP register defines a write protection (WRP) region with a granularity of 8KB, i.e., one sector size. See the description of the WRP register for details.

When a WRP region is activated, erase or program operations are not allowed. Correspondingly, even if only one region is set as write-protected, the mass erase function will not work.

Additionally, if an attempt is made to erase or program a write-protected region, the write protection error flag (WRPERR) in the FLASH_SR register will be set.

Note: Write protection only applies to the Flash main memory area and does not affect the system memory area.

4.8.4. Option byte write protection

By default, option bytes are readable and write-protected. To obtain erase or program access to option bytes, the correct sequence must be written to the OPTKEYR register.

4.8.5. Summary of Flash memory protection permissions

Table 4-8 Flash access permissions

No.	Access object	Protection type	Protection level	In protected area	Kind of error	Boot from main flash									Non-main area boot								
						CPU				DMA		SWD			CPU				DMA		SWD		
						Read data	Read instruction	Write	Erase	Read data	Write/Erase	Read data	Write	Erase	Read data	Read instruction	Write	Erase	Read data	Write/Erase	Read data	Write	Erase
1	Main storage area/backup register	RDP(2)	0	-	-	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
2			1	-	Bus error (BERR)	Y(1)	Y(1)	Y(1)	Y(1)	Y(1)	N	N	N	N	N	N	N	N	N	N	N	N	N
3	Main memory area	PCORP(2)	-	Not present	-	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
4			-	In	Read/Write Error Flag	N	Y	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N
5		WRPR(2)	-	Not present	-	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
6			-	In	Write Error Flag	Y	Y	N	N	Y	N	Y	N	N	Y	Y	N	N	Y	N	Y	N	N
11	System Storage Area			-	Write Bus Error	Y	Y	N	N	Y	N	Y	N	N	Y	Y	N	N	Y	N	Y	N	N
12	OTP	OTP_LOCK=0	-	-	-	Y	-	Y	Y	Y	N	Y	Y	Y	Y	-	Y	Y	Y	N	Y	Y	Y
13		OTP_LOCK=1	-	-	Write Error Flag	Y	-	N	N	Y	N	Y	N	N	Y	-	N	N	Y	N	Y	N	N
15	FT/UID/Factory Configuration			-	Write Bus Error	Y	-	N	N	Y	N	Y	N	N	Y	-	N	N	Y	N	Y	N	N
17	Option bytes		-		-	Y	-	Y	-	Y	N	Y	Y	-	Y	-	Y	-	Y	N	Y	Y	-

Notes:

- (1) Once SWD access has occurred, CPU/DMA access is not permitted. Access is allowed only when no SWD connection has been established.
- (2) The bus error and read/write error flags will not occur simultaneously. For example, if the read/write address has RDP=1 and is within the PCROP protected area, only a bus error will occur.

4.9. Flash interrupt

Table 4-9 Flash Interrupt Request

Interrupt event	Event flag	Event flag/interrupt clear	the interrupt enable
Operation end	EOP	Write EOP=1	EOPIE
Write protection error	WRPERR	Write WRPERR=1	ERRIE
Bank0 PCROP protection error	RDERR0	Write RDERR0=1	RDERR0IE
Bank1 PCROP protection error	RDERR1	Write RDERR1=1	RDERR1IE
ECC error correction	ECCC	Write ECCC=1	ECCCIE

Note: The following events do not have separate interrupt flags but generate a Hard fault:

- Sequence error for unlocking the FLASH_CR register
- Sequence error for unlocking Flash option byte write operations
- Flash programming operation not aligned to 32-bit data
- Flash erase operation (including page erase, sector erase, and chip erase) not aligned to 32-bit data

4.10. Flash Register

4.10.1. Flash access control register (FLASH_ACR)

Offset address: 0x00

Reset value: 0x0000 0700

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

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10	DCEN	RW	1	Data cache enabled 0: Data cache disabled 1: Data cache enabled
9	ICEN	RW	1	Instruction cache enable 0: Instruction cache disabled 1: Instruction cache enable
8	PRFTEN	RW	1	Prefetch enable bit 0: Prefetch prohibition 1: Prefetch enabled
7:2	Reserved	-	-	Reserved
1:0	LATENCY[1:0]	RW	0	Waiting state corresponding to Flash read operation: 0: No waiting state for Flash read operation (system clock at 24 MHz and below) 1: Flash read operation has 1 wait state, meaning each Flash read requires 2 system clock cycles (when system clock is >24 MHz and ≤48 MHz) 2: Flash read operation has 2 wait states, meaning each Flash read requires 3 system clock cycles (when system clock is >48 MHz and ≤72 MHz)

4.10.2. Flash key register (Flash_KEYR)

Offset address: 0x08

Reset value: 0x0000 0000

All register bits are write-only, readout returns 0.

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

Bit	Name	R/W	Reset Value	Function
31:0	KEY[31:0]	W	0x0000	The following values must be written consecutively to unlock the FLASH_CR register and enable Flash programming/erase operations KEY1: 0x4567 0123 KEY2: 0xCDEF 89AB

4.10.3. Flash option byte key register (FLASH_OPTKEYR)

Offset address: 0x0C

Reset value: 0x0000 0000

All register bits are write-only, readout returns 0.

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

Bit	Name	R/W	Reset Value	Function
31:0	OPTKEY[31:0]	W	0x0000 0000	The following values must be written consecutively to unlock the Flash option registers and enable option byte programming/erase operations KEY1: 0x0819 2A3B KEY2: 0x4C5D 6E7F Note: The FLASH_KEYR must be unlocked first before unlocking this register.

4.10.4. Flash status register (FLASH_SR)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BSY1	BSY0
														R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPTV ERR	Res.	USRLOK	RDERR1	RDERR0	Res.	Res.	Res.	Res.	Res.	Res.	WRP ERR	Res.	Res.	Res.	EOP
RC_W1		R	RC_W1	RC_W1							RC_W1				RC_W1

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	Reserved
17	BSY1	R	0	Bank1 Busy bit This bit indicates that the operation of Flash bank1 is in progress. This bit is set by the hardware at the beginning of the flash operation, and is cleared by the hardware when the operation is completed or an error occurs.
16	BSY0	R	0	Bank0 Busy bit This bit indicates that the operation of Flash bank0 is in progress. This bit is set by the hardware at the beginning of the flash operation, and is cleared by the hardware when the operation is completed or an error occurs.
15	OPTVERR	RC_W1	0	Option byte verification error flag. This bit is set by hardware when the option byte and its complement do not match. Failed to load option bytes, forced to safe values. This bit is cleared by writing 1.
14	Reserved	-	-	Reserved

13	USRLCK	R	0	Indicates whether the user data area is writable based on the lower 16 bits of the first word read from the user data area during power-on. 0: The read value is not 0xAA55, the user data area is writable. 1: The read value is 0xAA55, the user data area is not writable. This bit can only be reset by a power-on reset.
12	RDERR1	RC_W1	0	Bank1 PCROP read error. This bit is set by hardware when the address of the data to be read in Bank1 belongs to the Flash's read protection area (PCROP protection). An interrupt will be generated if RDERR1IE in FLASH_CR is set to 1. This bit is cleared by writing 1.
11	RDERR0	RC_W1	0	Bank0 PCROP read error. This bit is set by hardware when the address of the data to be read in Bank0 belongs to the Flash's read protection area (PCROP protection). An interrupt will be generated if RDERR0IE in FLASH_CR is set to 1. This bit is cleared by writing 1.
10:5	Reserved	-	-	Reserved
4	WRPERR	RC_W1	0	Write protection error This bit is set by hardware when the address to be programmed/erased is in a write-protected Flash area (protected by WRP). This bit is also set by hardware when the user data area is programmed/erased while USRLCK is 1. This bit is set by hardware when the address to be programmed/erased/read is in a Flash area protected by proprietary code (PCROP). This bit is cleared by writing 1.
3:1	Reserved	-	-	Reserved
0	EOP	RC_W1	0	This bit is set by hardware when the Flash programming/erasure operation is successfully completed. An interrupt is generated if the EOPIE bit in the FLASH_CR register is enabled. This bit is cleared by writing 1.

4.10.5. Flash control register (FLASH_CR).

Offset address: 0x14.

Reset value: 0xC000 0000.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	OPTLOCK	Res.	Res.	OBLAUNCH	Res.	ERRIE	EOPIE	RDERR1IE	RDERR0IE	Res.	Res.	PGSTRT	UPGSTRT	OPTSTRT	Res.
RS	RS			RS		RW	RW	RW	RW			RW	RW	RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	SER	Res.	Res.	Res.	Res.	Res.	UPER.	UPG	MER1	MER0	PER	PG
				RW						RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RS	1	FLASH_CR register lock bit. The software can only set this bit. When set, the FLASH_CR register is locked. This bit is cleared by hardware upon detecting the unlock sequence, unlocking the FLASH_CR register. If the unlock operation fails, this bit remains set until the next system reset.
30	OPTLOCK	RS	1	Option byte lock bit. The software can only set this bit. When set, the bits related to option bytes in the FLASH_CR register and the option byte registers FLASH_OPTR, FLASH_WRP, FLASH_PCROPx (x=0,1) are locked. When the unlock sequence is detected, this bit is cleared by hardware, unlocking the FLASH_CR register and option byte-related registers.

				The LOCK bit must be cleared before performing the unlock sequence on the OPTLOCK bit. If the unlock operation fails, this bit remains set until the next system reset.
29:28	Reserved	-	-	Reserved
27	OBL_LAUNCH	RS	0	Forces option byte loading. When set, this bit forces the system to reload the option bytes. This bit is cleared by hardware only after the option byte loading is completed. If the OPTLOCK bit is set, the bit cannot be written. 0: Option byte loading complete 1: Generates an option byte load request, causing a system reset to reload the option bytes.
26	Reserved	-	-	Reserved
25	ERRIE	RW	0	Error interrupt enable bit. When the WRPER bit of the FLASH_SR register is set, if the bit is enabled, an interrupt request is generated. 0: No interrupt occurrence 1: An interrupt occurs
24	EOPIE	RW	0	End-of-operation interrupt enable bit. When the EOP bit of the FLASH_SR register is set, if the bit is enabled, an interrupt request is generated. 0: EOP interrupt disabled 1: EOP interrupt enabled
23	RDERR1IE	RW	0	Bank1 PCROP read error interrupt enable. When the RDERR1 bit in FLASH_SR is set, an interrupt request is generated if this bit is enabled. 0: No interrupt occurrence 1: An interrupt occurs
22	RDERR0IE	RW	0	Bank0 PCROP read error interrupt enable. When the RDERR0 bit in FLASH_SR is set, an interrupt request is generated if this bit is enabled. 0: No interrupt occurrence 1: An interrupt occurs
21:20	Reserved	-	-	Reserved
19	PGSTRT	RW	0	Start bit for Flash main memory programming operation. This bit initiates the Flash main memory programming operation. It is set by software and cleared by hardware after the BSY0 or BSY1 bit in the FLASH_SR register is cleared.
18	UPGSTRT	RW	0	Start bit for Flash information block user data area programming operation. This bit initiates the Flash information block user data area programming operation. It is set by software and cleared by hardware after the BSY0 bit in the FLASH_SR register is cleared.
17	OPTSTRT	RW	0	Flash option byte modification start bit This bit triggers the modification of the option byte. Set by software, cleared by hardware after the BSY0 bit in the FLASH_SR register is cleared. Note: When modifying Flash option bytes, the hardware automatically erases the entire 256-byte page and performs programming, including writing the complement automatically.
16:12	Reserved	-	-	Reserved
11	SER	RW	0	8 KB Sector erase operation This bit is set and cleared by software. 0: Disable Flash sector erase operation. 1: Enable Flash sector erase operation. Notes: 1) Sector erase does not affect the Flash information block. 2) Sector erase has no effect on areas set as WRP.
10:6	Reserved	-	-	Reserved
5	UPER	RW	0	User data page erase operation This bit is set and cleared by software. 0: Disable page erase operation for Flash user data pages 1: Enable page erase operation for Flash user data pages

				Note: If the user data area cannot be erased or written, this configuration is invalid.
4	UPG	RW	0	User data area programming operation This bit is set and cleared by software. 0: Disable programming operation for the user data area 1: Enable programming operation for the user data area Note: If the user data area cannot be erased or written, this configuration is invalid.
3	MER1	RW	0	Bank 1 mass erase operation This bit is set and cleared by software. 0: Disable mass erase operation for Flash Bank1 1: Enable mass erase operation for Flash Bank1
2	MER0	RW	0	Bank 0 mass erase operation This bit is set and cleared by software. 0: Disable mass erase operation for Flash Bank0 1: Enable mass erase operation for Flash Bank0
1	PER	RW	0	Page erase operation This bit is set and cleared by software. 0: Disable page erase operation for Flash 1: Enable page erase operation for Flash
0	PG	RW	0	Programming operation This bit is set and cleared by software. 0: Disable programming operation for Flash 1: Enable programming operation for Flash

4.10.6. Flash ECC register (FLASH_ECCR)

Offset address: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCD	ECCC	Res	Res	Res	Res	Res	ECCIE	Res	SYSF_ECC	BK_ECC	Res	Res	Res	Res	ADDR_ECC[16]
RC_W1	RC_W1						RW		R	R					R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_ECC[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset value	Function
31	ECCD	RC_W1	0	ECC detection bits Set by hardware when more than 1-bit ECC error is detected (only when ECCC/ECCD was previously cleared). When this bit is set, an NMI interrupt is generated. Cleared by writing 1.
30	ECCC	RC_W1	0	ECC correction Set by hardware when a 1-bit ECC error is detected and corrected (only when ECCC/ECCD was previously cleared). Generates an interrupt when ECCCIE is set. Cleared by writing 1.
29:25	Reserved	-	-	Reserved
24	ECCCIE	RW	0	ECC correction interrupt enable 0: Disable ECCC interrupt 1: Enable ECCC interrupt. When the ECCC bit in the FLASH_ECCR register is set, this bit enables interrupt generation.
23	Reserved	-	-	Reserved
22	SYSF_ECC	R	0	Flash information zone ECC failure flag. 0: ECC not enabled or no ECC error 1: ECC error detected or ECC correction occurred
21	BK_ECC	R	0	Flash ECC fault Bank Indicates which Bank is related to ECC correction or ECC error detection. 0: Bank 0 1: Bank 1 This flag is only for the Flash main memory area.
20:17	Reserved	-	-	Reserved
16:0	ADDR_ECC[16:0]	R	17'b0	ECC fault address

				This bit indicates which address in which region is related to ECC error correction or ECC error detection. When ECC or ECCD occurs, the erroneous logical address indicated by this register is as shown in the following table: <table><tr><th>SYSF_ECC</th><th>BK_ECC</th><th>Erroneous Logical Address</th></tr><tr><td>0</td><td>0</td><td>{12'h080,3'b0, ADDR_ECC[16:0]}</td></tr><tr><td>0</td><td>1</td><td>{12'h080,3'b0, ADDR_ECC[16:0]+Bank0 maximum address}</td></tr><tr><td>1</td><td>0</td><td>{16'h1FFF,3'b0,ADDR_ECC[12:0]};</td></tr></table> Note: If the flag (SYSF_ECC or BK_ECC) set after an ECC error or correction is not handled by software, new errors will not set the flag again (whether SYSF_ECC or BK_ECC).	SYSF_ECC	BK_ECC	Erroneous Logical Address	0	0	{12'h080,3'b0, ADDR_ECC[16:0]}	0	1	{12'h080,3'b0, ADDR_ECC[16:0]+Bank0 maximum address}	1	0	{16'h1FFF,3'b0,ADDR_ECC[12:0]};
SYSF_ECC	BK_ECC	Erroneous Logical Address														
0	0	{12'h080,3'b0, ADDR_ECC[16:0]}														
0	1	{12'h080,3'b0, ADDR_ECC[16:0]+Bank0 maximum address}														
1	0	{16'h1FFF,3'b0,ADDR_ECC[12:0]};														

4.10.7. Flash option byte register (FLASH_OPTR)

Offset address: 0x20

Reset value: 0b0000 00xx xxxx xxxx xxxx xxxx xxxx. After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area in the Flash information area and written to the corresponding option byte bit in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	ECC_EN	BOOT_LOCK	BFB	SWD_MODE	SRAM_PE	BOR_LEV[2:0]				
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
nBOOT1	NRS_T_MOD_E	WWDG_SW	IWDG_SW	IWDG_STOP	IWDG_STDBY	NRST_STOP	NRST_STDBY	RDP[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function																									
31:26	Reserved	-	-	Reserved																									
25	ECC_EN	RW	-	ECC enabled 0: Disable ECC 1: Enable ECC																									
24	BOOT_LOCK	RW	-	Use to force boot from user Flash zone 0: Boot based on IO pin/option bit configuration 1: Force boot from Flash main memory area																									
23	BFB	RW	-	0: Boot from Bank0 (Flash Bank0 mapped to 0x0800_0000/0x0000_0000, Flash Bank1 mapped to 0x0802_0000/0x0002_0000) 1: Boot from Bank1 (Flash Bank1 mapped to 0x0800_0000/0x0000_0000, Flash Bank0 mapped to 0x0802_0000/0x0002_0000)																									
22:21	SWD_MODE[1:0]	RW	-	SWD PIN mode selection. <table border="1"> <thead> <tr> <th>SWD_MODE[1:0]</th><th>PA4</th><th>PA5</th><th>PB6</th><th>PB5</th></tr> </thead> <tbody> <tr> <td>00(default)</td><td>SWDIO</td><td>SWCLK</td><td>GPIO</td><td>GPIO</td></tr> <tr> <td>01</td><td>GPIO</td><td>GPIO</td><td>SWDIO</td><td>SWCLK</td></tr> <tr> <td>10</td><td>GPIO</td><td>SWCLK</td><td>SWDIO</td><td>GPIO</td></tr> <tr> <td>11</td><td>SWDIO</td><td>GPIO</td><td>GPIO</td><td>SWCLK</td></tr> </tbody> </table> When PA5 and PB5 are configured as SWCLK via SWD_MODE, they are set to pull-down mode; When PA4 and PB6 are configured as SWDIO via SWD_MODE, they are set to pull-up mode;	SWD_MODE[1:0]	PA4	PA5	PB6	PB5	00(default)	SWDIO	SWCLK	GPIO	GPIO	01	GPIO	GPIO	SWDIO	SWCLK	10	GPIO	SWCLK	SWDIO	GPIO	11	SWDIO	GPIO	GPIO	SWCLK
SWD_MODE[1:0]	PA4	PA5	PB6	PB5																									
00(default)	SWDIO	SWCLK	GPIO	GPIO																									
01	GPIO	GPIO	SWDIO	SWCLK																									
10	GPIO	SWCLK	SWDIO	GPIO																									
11	SWDIO	GPIO	GPIO	SWCLK																									
20	SRAM_PE	RW	-	SRAM parity check enable 0: Disable SRAM parity check 1: Enable SRAM parity check																									
19:17	BOR_LEV[2:0]	RW	-	000: BOR rising threshold is 1.8V and falling threshold is 1.7V 001: BOR rising threshold is 2.0V and falling threshold is 1.9V 010: BOR rising threshold is 2.2V and falling threshold is 2.1V																									

				011: BOR rising threshold is 2.4V and falling threshold is 2.3V 100: BOR rising threshold is 2.8V and falling threshold is 2.7V 101: BOR rising threshold is 3.1V and falling threshold is 3.0V 110: BOR rising threshold is 3.7V and falling threshold is 3.6V 111: BOR rising threshold is 4.2V and falling threshold is 4.1V
16	BOR_EN	RW	-	BOR enabled 0: BOR not enabled 1: BOR enabled
15	nBOOT1	RW	-	Together with the BOOT PIN, select the boot mode
14	NRST_MODE	RW	-	0: Reset input only 1: GPIO function
13	WWDG_SW	RW	-	0: Hardware window watchdog 1: Software window watchdog
12	IWDG_SW	RW	-	0: Hardware independent watchdog 1: Software independent watchdog
11	IWDG_STOP	RW	-	Stop mode IWDG count control 0: IWDG counting stops in stop mode 1: IWDG counting operates normally in stop mode
10	IWDG_STDBY	RW	-	IWDG counting control in Standby mode 0: IWDG counting stops in Standby mode 1: IWDG counting operates normally in Standby mode
9	NRST_STOP	RW	-	0: Generate reset when entering Stop mode 1: No reset generated when entering Stop mode
8	NRST_STDBY	RW	-	0: Generate reset when entering Standby mode 1: Do not generate reset when entering Standby mode
7:0	RDP	RW	-	Read protection level. 0xAA: Level 0, read protection not activated Non-0xAA: Level 1, chip read protection activated

4.10.8. Flash write protection register (FLASH_WRP)

Offset address: 0x28

Reset value: 0xFFFF FFFF

After power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding values are read from the option byte area in the Flash information sector and written to the corresponding option byte bits in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BK1_WRP [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BK0_WRP [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	BK1_WRPx	RW	1	Bank1 sector write protection 0: Sector x (x=0~15) is write-protected; programming and erasing are not allowed 1: Sector x (x=0~15) has no write protection
15:0	BK0_WRPx	RW	1	Bank0 sector write protection 0: Sector x (x=0~15) is write-protected; programming and erasing are not allowed 1: Sector x (x=0~15) has no write protection

4.10.9. Flash Bank0 proprietary code protection address register (FLASH_PCROPR0)

Offset address: 0x30

Reset value: 0b0000 000X XXXX XXXX 0000 000X XXXX XXXX. After power-on reset

(POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area in the Flash information area and written to the corresponding option byte bit in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP0ER [8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP0SR [8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24:16	PCROP0ER [8:0]	RW		Bank0 PCROP area end address (in pages)
15:9	Reserved	-	-	Reserved
8:0	PCROP0SR [8:0]	RW		Bank0 PCROP area start address (in pages)

4.10.10. Flash Bank1 proprietary code protection address register (FLASH_PCROPR1)

Offset address: 0x34

Reset value: 0b0000 000X XXXX XXXX 0000 000X XXXX XXXX. After power-on reset

(POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area in the Flash information area and written to the corresponding option byte bit in this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1ER [8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1SR [8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24:16	PCROP1ER [8:0]	RW		Bank1 PCROP area end address (in pages)
15:9	Reserved	-	-	Reserved
8:0	PCROP1SR [8:0]	RW		Bank1 PCROP area start address (in pages)

4.10.11. Flash low-power configuration register (FLASH_LPCR)

Offset address: 0x90

Reset value: 0x0000 2800

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPRUN_EXITT[2:0]			Res.	LPRUN_ENTERT[2:0]		
									RW	RW	RW		RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLEEP_TIME[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	SLEEP_EN
RW	RW	RW	RW	RW	RW	RW	RW								RW

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	Reserved
22:20	LPRUN_EXITT[2:0]	RW	0x6	Time control between the falling edge of Flash interface signal FM_LPMODE and the rising edge of FM_ACLKBKx. This register defines the count of MSISYS clock cycles.
19	Reserved	-	-	Reserved
18:16	LPRUN_ENTERT[2:0]	RW	0x6	Time control between the rising edge of Flash interface signal FM_LPMODE and the rising edge of FM_ACLKBKx. This register defines the count of MSISYS clock cycles.
15:8	SLEEP_TIME[7:0]	RW	0x28	Flash sleep time configuration (based on HSI_10M count)

				When the system clock selects LSI or LSE, in order to obtain a more optimized operating mode power consumption, the function of using this register can be selected (this function is only recommended when LSI or LSE is the system clock). When this feature is enabled, the time width for which Flash is in Sleep mode for every half of the system clock low cycle is: $t_{HSI\_10M} * SLEEP\_TIME$ Note: t_{HSI_10M} is the period of HSI_10M; To ensure the function of Flash, the maximum value of this register is recommended to be set to 0x28.
7:1	Reserved	-	-	Reserved
0	SLEEP_EN	RW	0	Flash Sleep enable 0: Disabled 1: Enabled

4.10.12. Flash TS0 register (FLASH_TS0)

The default values of FLASH_TS0~FLASH_TACKL2PW registers correspond to the time configuration when HSI frequency is 2 MHz. HSI 2 MHz is generated by frequency division of HSI.

Offset address: 0x100

Reset value: 0x0000 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.					TS0 [8:0]				
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	TS0 [8:0]	RW	9'h0F	This register contains the count value based on HSI 2 MHz. Note: Writing to this register requires unlocking the LOCK bit in the CR register; otherwise, the write operation will be ignored.

4.10.13. Flash TS1 Register (Flash_TS1)

Offset Address: 0x104

Reset Value: 0x0000 0024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.							TS1 [10:0]				
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10:0	TS1 [10:0]	RW	11'h24	This register contains the count value based on HSI 2 MHz. Note: Writing to this register requires unlocking the LOCK bit in the CR register; otherwise, the write operation will be ignored.

4.10.14. Flash TS2P Register (FLASH_TS2P)

Offset Address: 0x108

Reset value: 0x0000 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.					TS2P [8:0]				

4.10.15. Flash TPS3 Register (FLASH_TPS3)

Reset Value: 0x0000 0090

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12:0	TPS3 [12:0]	RW	13'h90	This register contains the count value based on HSI 2 MHz. Note: Writing to this register requires unlocking the LOCK bit in the CR register; otherwise, the write operation will be ignored.

Reset value: 0x0000 000F

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	TS3 [8:0]	RW	9'h0F	This register contains the count value based on HSI 2 MHz. Note: Writing to this register requires unlocking the LOCK bit in the CR register; otherwise, the write operation will be ignored.

Reset Value: 0x0000 1B58

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	Reserved
17:0	PERTPE[17:0]	RW	18'h1B58	This register contains the count value based on HSI 2 MHz. Note: Writing to this register requires unlocking the LOCK bit in the CR register; otherwise, the write operation will be ignored.

4.10.18. Flash sector erase time TPE register (FLASH_SMERTPE)

Offset address: 0x118

Reset Value: 0x0000 1B58

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

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	Reserved
17:0	SMERTPE[17:0]	RW	18'h1B58	This register contains the count value based on HSI 2 MHz. Note: Writing to this register requires unlocking the LOCK bit in the CR register; otherwise, the write operation will be ignored.

4.10.19. Flash programming time TPE register (FLASH_PRGTPPE)

Offset address: 0x11C

Reset value: 0x0000 07D0

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PRGTPPE[15:0]	RW	16'h7D0	This register contains the count value based on HSI 2MHz. Note: Writing to this register requires unlocking the LOCK bit in the CR register; otherwise, the write operation will be ignored.

4.10.20. Flash pre-programming time TPE register (FLASH_PRETPE)

Offset address: 0x120

Reset value: 0x0000 0190

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

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13:0	PRETPE[13:0]	RW	14'h0190	This register contains the count value based on HSI 2MHz. Note: Writing to this register requires unlocking the LOCK bit in the CR register; otherwise, the write operation will be ignored.

4.10.21. Flash page write wait time TACLK2PW register (FLASH_TACLK2PW)

Offset address: 0x124

Reset value: 0x0000 0006

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TACLK2PW [7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	TACLK2PW	RW	8'h6	Wait time from Flash ACLK rising edge to PW (page write). Software must configure this register according to the HSI frequency after configuring the FLASH_CR register. This register contains the count value based on HSI 2 MHz. Note: Writing to this register requires unlocking the LOCK bit in the CR register; otherwise, the write operation will be ignored.

5. Power Management Unit (PMU)

5.1. Power supply

5.1.1. Power Structure

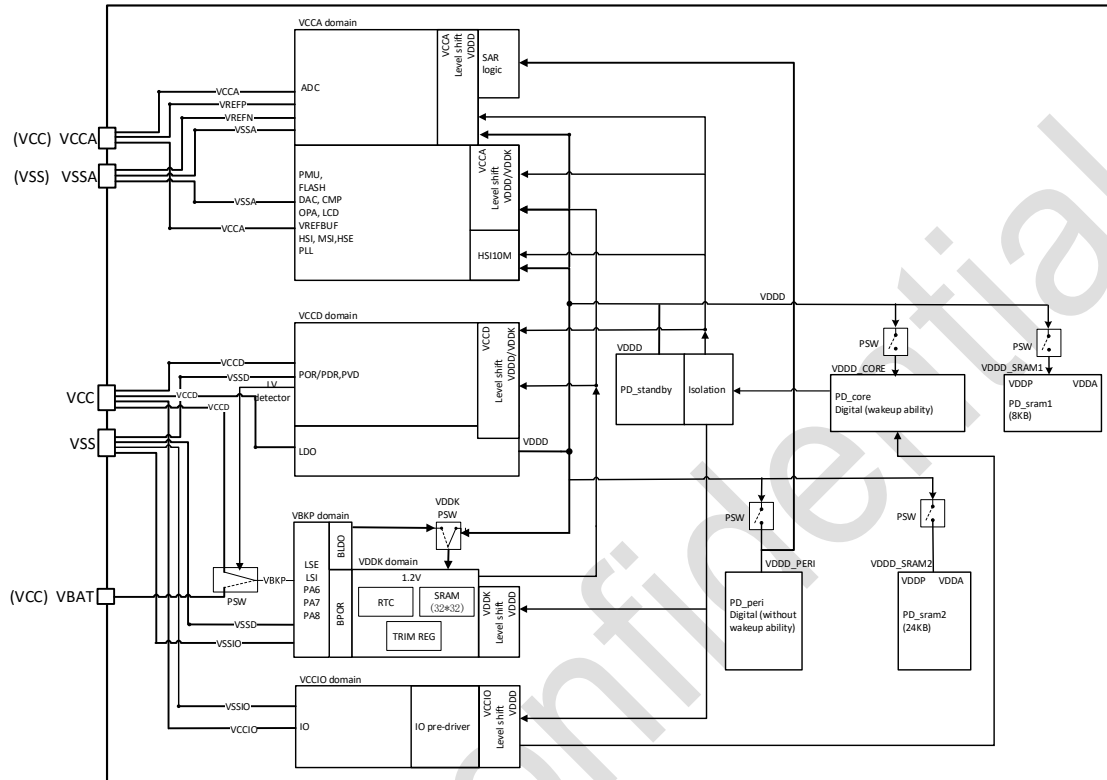


Figure 5-1 Power structure block diagram

Table 5-1 Power supply overview

No.	Power supply	Power value	Description
1	V _{CC}	1.8 to 5.5 V	The power is supplied to the device through the power pins, with the power supply module comprising: Partial analog circuits.
2	V _{CCA}	1.8 to 5.5 V	Powers for most analog modules, sourced from the V _{CC} PAD (a dedicated power PAD can also be designed separately).
3	V _{REFP}	1.8 to 5.5 V	Reference voltage for ADC and DAC
4	V _{DDD}	1.2 V	Output from VR, powers the chip standby mode wake-up circuit and IWDG module. When the MR is powered, it outputs 1.2 V. According to the software configuration, when entering the Stop or Standby mode it powered by MR, LPR and DLPR.
5	V _{DDD_CO} RE	1.2 V	Output controlled by the digital switch from V _{DDD} , powers the internal main logic circuits (CPU, bus, RCC, and digital circuits with stop mode wake-up capability) and SRAM1 (8 KB).
6	V _{DDD_SR} AM1	1.2 V	Output controlled by the digital switch from V _{DDD} , powers SRAM1 (8 KB).
7	V _{DDD_PEI} R	1.2 V	Output controlled by the digital switch from V _{DDD} , powers digital circuits without stop mode wake-up capability.
8	V _{DDD_SR} AM2	1.2 V	Output controlled by the digital switch from V _{DDD} , powers SRAM2.
9	V _{DDK}	1.2/1.0 V	Backup power supply. Provides power to the backup domain modules (RTC, backup registers, Trim registers).

5.1.2. System Main Power Supply

5.1.2.1. ADC and DAC reference voltage

To achieve higher precision for low-voltage input and output, users can connect V_{REFP} to an independent reference voltage lower than V_{CCA} . For analog input (ADC) or output (DAC) signals, V_{REFP} represents the highest voltage, expressed as the full-scale value.

V_{REFP} can be provided by an external reference or the internally buffered voltage reference (V_{REFBUF}).

Enable the internally buffered voltage reference by setting the $VREFBUF\_EN$ bit in the $VREFBUF$ control register ($VREFBUF\_CR$) to 1. The voltage value of V_{REFBUF} is selected via the $VREFBUF\_CR.VREFBUF\_OUT\_SEL$ register.

5.1.3. Dynamic voltage regulation

Dynamic voltage scaling is a power management technique that involves increasing or decreasing the voltage for digital peripherals based on application performance and power consumption requirements.

Increasing the voltage for digital peripherals through dynamic voltage scaling is called overvoltage. It enhances device performance, allowing the chip to operate at higher frequencies.

Decreasing the voltage for digital peripherals through dynamic voltage scaling is called undervoltage. It is used to save power while maintaining defined performance.

When the LDO operates in MR mode, it provides two voltage ranges:

5.1.3.1. MR mode 1: Standard voltage performance range

The output voltage provided by the main regulator supports access at a maximum system clock frequency of 72 MHz. The Flash access time for read operations is 30 ns, enabling Flash programming and erase operations.

5.1.3.2. MR Mode 3: Low-voltage performance range

The output voltage provided by the main regulator reduces power consumption by lowering the voltage at lower system clock frequencies (e.g., 8 MHz). The Flash access time for read operations is 30 ns, enabling programming and erase operations.

When VR_MODE in the PWR_CR1 register is configured as 2'b00, the voltage value in MR mode is selected via the $PWR\_CR1.MR\_VSEL[1:0]$ bits.

5.1.3.3. MR mode switching from low voltage to high voltage

In MR mode, the switching steps of the system clock frequency from low frequency to high frequency and the voltage from low frequency to high frequency are as follows:

1. VR_MODE bit in Power Control Register 1 (PWR_CR1) is configured as 2'b00
2. Based on the supported frequency, configure the appropriate MR voltage mode by referring to the frequency supported in $MR\_VSEL[1:0]$
3. Wait for 3 μ s
4. Configure the value of $FLASH\_ACR.LATENCY$ based on the supported frequency
5. Switch the system clock frequency

5.1.3.4. MR mode switching from high voltage to low voltage

In MR mode, the switching steps of the system clock frequency from high frequency to low frequency and the voltage from high frequency to low frequency are as follows:

1. VR_MODE bit in Power Control Register 1 (PWR_CR1) is configured as 2'b00
2. Switch the system frequency
3. Configure the value of FLASH_ACR.LATENCY based on the supported frequency
4. Based on the supported frequency, configure the appropriate MR voltage mode by referring to the frequency supported in MR_VSEL[1:0]

5.1.3.5. Frequency switching and VR operating mode switching

When switching frequencies, the following must be satisfied if the VR operating mode also needs to be switched:

- For MR->LPR or MR->DLPR, the system clock frequency must be reduced first before switching the VR mode.
- For LPR->MR or DLPR->MR, the VR mode must be switched first, followed by increasing the frequency (if needed).

5.2. Power monitoring

5.2.1. Power-on reset (POR)/Power-down reset (PDR)/Brown-out reset (BOR)

The device has an integrated power-on reset (POR) / power-down reset (PDR). The module keeps working in all modes.

In addition to POR/PDR, a brown-out reset (BOR) is also implemented. BOR can only be enabled and disabled through the option byte.

When the BOR is turned on, the BOR threshold can be selected by the option byte and both the rising and falling detection points can be individually configured.

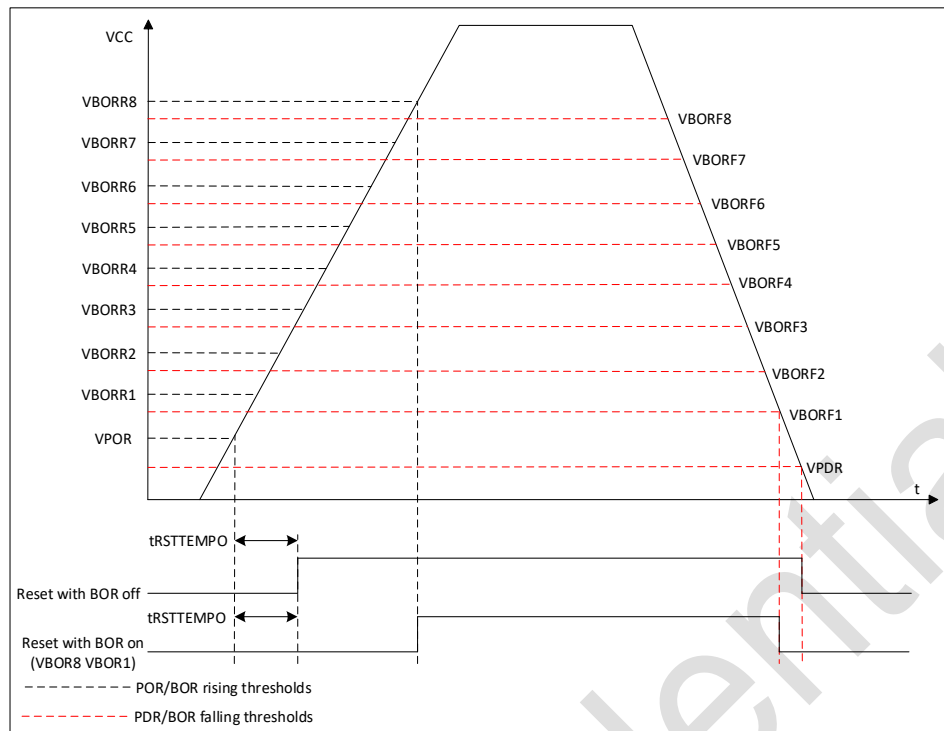


Figure 5-2 POR, PDR, and BOR Thresholds

5.2.2. Programmable Voltage Detector (PVD)

This module can be used to monitor the V_{CC} power supply, and the detection point can be configured via registers. When monitoring the V_{CC} power supply, if V_{CC} rises above or falls below the PVD detection threshold (configured by `PWR_CR2.PLS`), a corresponding flag is generated.

This event is internally connected to EXTI line 16. Depending on the EXTI line 16 rising/falling edge configuration, an interrupt is generated when V_{CC} rises above or falls below the PVD detection point. In the interrupt service routine, the user can perform emergency system shutdown tasks.

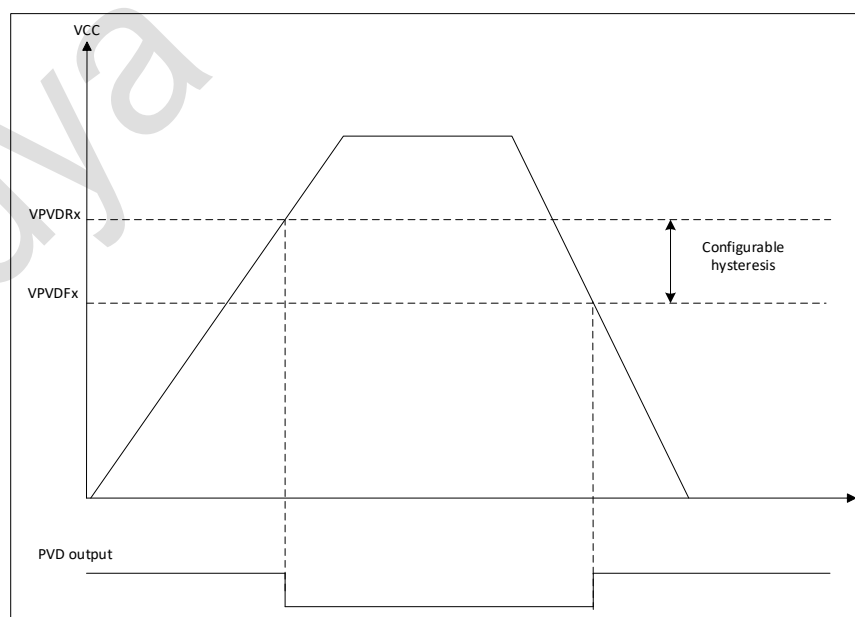


Figure 5-3 PVD threshold

5.3. Low power control

By default, the chip enters run mode after a system or power reset. When the CPU does not require continuous operation, the chip can enter several low-power modes, such as when waiting for an external event. Software can make trade-offs between power consumption, wake-up time, and wake-up sources.

5.3.1. Power mode

5.3.1.1. Timer

The chip has seven power modes:

- Run mode: Normal full-function mode, the default mode after power-on.
- Lower power run mode: The system clock selects MSISYS, with a maximum frequency of 2 MHz. The digital power supply V_{DD} is recommended to operate in LPR mode to save power. Flash erase/write operations are not supported in this mode.
- Sleep mode: The CPU HCLK clock is turned off (NVIC, SysTick, etc., remain operational), and peripherals can be configured to continue working. (It is recommended to enable only necessary modules and turn them off after their tasks are completed.) The digital power supply V_{DD} is recommended to operate in MR mode.
- Lower power sleep mode: Enter sleep mode from low-power mode. The digital power supply V_{DD} is recommended to operate in LPR mode.
- Stop mode (Stop0/1/2/3): High-speed clocks (PLL, HSI, HSE, and MSI) are turned off. LSI and LSE can be configured to remain on based on wake-up source requirements. The digital power supply V_{DD} can be configured to operate in MR, LPR, or DLPR mode. These three modes progressively reduce power consumption while increasing wake-up time.
- Standby mode: V_{DD_CORE} , V_{DD_PERI} , and V_{DD_SRAM2} are powered down. If V_{DD_SRAM1} is configured to power down, SRAM data is lost. V_{DD} and V_{DDK} remain operational. Standby mode wake-up sources and low-power control modules are active.

Note: To enter low-power modes (Sleep/Stopx/Standby) from Run mode, the HSI clock must be configured to 8 MHz.

5.3.1.2. Power state transition

The figure below shows the transition relationships between various states:

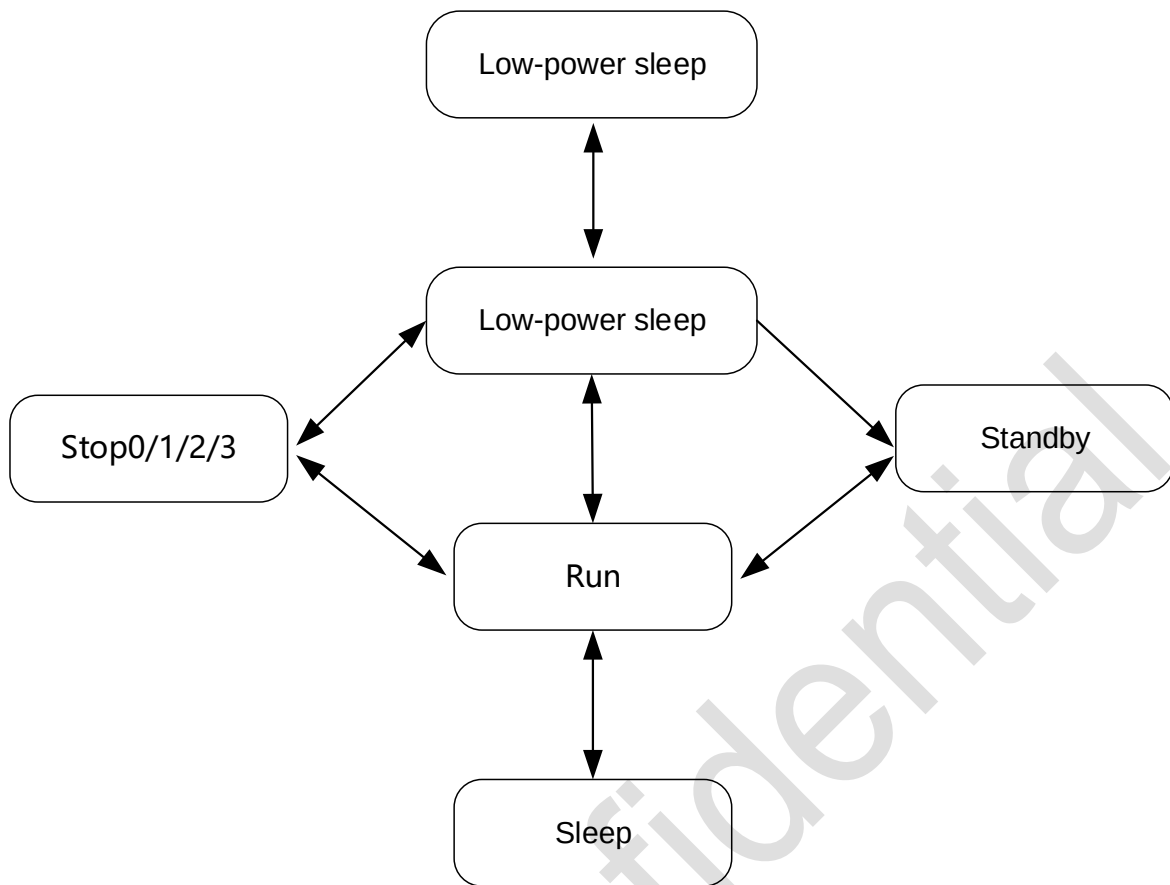


Figure 5-4 Power state transition

The operating conditions of modules in various modes are shown in the table below. The VR operating modes in the table are configured as an example to balance performance and power consumption.

Table 5-2 Module operating conditions in various modes(1)

Peripheral	RUN	LP RUN	SLEEP	LP SLEEP	STOP0		STOP1		STOP2		STOP3		STANDBY	
		VDDD VR@LPR		VDDD VR@LPR	VDDD VR@LPR	Wake up capa- bility	VDDD VR@LPR	Wake up capa- bility	VDDD VR@DLPR	Wake up capa- bility	VDDD VR@DLPR	Wake up capa- bility	VDDD VR@DLPR	Wake up capa- bility
CPU Core	Y	Y	-	-	-	-	-	-	-	-	-	-	-	-
Flash memory	Y	Y (9)	O(3)	O(3)	- (2)	-	- (2)	-	- (2)	-	- (2)	-	-	-
SRAM1	Y	Y	O(3)	O(3)	- (4)	-	- (4)	-	- (4)	-	- (4)	-	-	-
SRAM2	Y	Y	O(3)	O(3)	-(6)	-	-(5)	-	-(6)	-	-(5)	-	-	-
Brown-out re- set (BOR)	O	O	O	O	O	O	O	O	O	O	O	O	O	O
PVD	O	O	O	O	O	O	O	O	O	O	O	O	-	-
DMA	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
HSI	O	O(10)	O	O(10)	-	-	-	-	-	-	-	-	-	-
MSI	O	O	O	O	-	-	-	-	-	-	-	-	-	-
HSE	O	O(10)	O	O(10)	-	-	-	-	-	-	-	-	-	-
LSI	O	O	O	O	O	-	O	-	O	-	O	-	O	-
LSE	O	O	O	O	O	-	O	-	O	-	O	-	O	-
PLL	O	O(10)	O	O(10)	-	-	-	-	-	-	-	-	-	-
HSE CSS	O	O	O	O	-	-	-	-	-	-	-	-	-	-
LSE CSS	O	O	O	O	O(8)	O	O(8)	O	O(8)	O	O(8)	O	-	-
RTC	O	O	O	O	O	O	O	O	O	O	O	O	O	O
BKP	O(13)	O(13)	O(13)	O(13)	O(13)	O(13)	O(13)	O(13)	O(13)	O(13)	O(13)	O(13)	O(13)	O(13)
USART1 USART2	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
UART	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
LPUART1 LPUART2	O	O	O	O	O	O	O	O	O	O	O	O	-	-
I2C1 I2C2	O	O	O	O	O	O	O	O	O	O	O	O	-	-
SPI1 SPI2	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
ADC	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
DAC	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
COMP1/2	O	O	O	O	O	O	O	O	O	O	O	O	-	-
OPA	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-

Peripheral	RUN	LP RUN	SLEEP	LP SLEEP	STOP0		STOP1		STOP2		STOP3		STANDBY	
		VDDD VR@LPR		VDDD VR@LPR	VDDD VR@LPR	Wake up capability	VDDD VR@LPR	Wake up capability	VDDD VR@DLPR	Wake up capability	VDDD VR@DLPR	Wake up capability	VDDD VR@DLPR	Wake up capability
Temperature sensor	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
Timers	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
PWM	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
LPTIM1 LPTIM2	O	O	O	O	O	O	O	O	O	O	O	O	-	-
IWDG	O	O	O	O	O	O	O	O	O	O	O	O	O	O
WWDG	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
SysTick timer	O	O	O	O	-	-	-	-	-	-	-	-	-	-
CRC	O	O	O	O	-	-	-	-	-(7)	-	-(7)	-	-	-
LCD	O	O	O	O	O(14)	-	O(14)	-	O(14)	-	O(14)	-	-	-
GPIOs	O	O	O	O	O	O	O	O	O	O	O	O	O(11)	O(11)
TK	O	O	O	O	O	O	O	O	O	O	O	O	-	-

注1. Legend: Y = Yes (Enable). O = Optional (Disable by default. Can be enabled by software). - = Not available

注2. Flash does not power down, but no clock is provided, and enters the lowest power consumption state. In this case, FLASH requires a 3us wake-up time.

注3. The clocks for Flash and SRAM1/SRAM2 can be configured to be on or off in SLEEPING mode.

注4. SRAM1 remains powered, retains content, but has no clock supply, entering the lowest power state.

注5. SRAM2 is configured to power down.

注6. SRAM2 remains powered, retains content, but has no clock supply, entering the lowest power state.

注7. This module is configured to power down.

注8. Before entering low-power mode, if LSE CSS is enabled, the system will wake up and enter NMI interrupt when an LSE CSS issue occurs.

注9. Flash remains powered, clock is provided, and enters low-power operation state. When switching to RUN state, wait for the transition to complete.

注10. For minimum power consumption, this module is recommended to be configured as off.

注11. The IOs for standby mode wake-up function normally, and the VBKP domain IOs (PA6/PA7/PA8) function normally.

注12. Only PA6, PA7, and PA8 function normally.

注13. BKP PCLK control is consistent with RTC PCLK.

注14. The LCD can maintain display in this low-power mode.

In addition, power consumption can be reduced in Run mode by:

- Reduce the system clock frequency

- For unused peripherals, turn off their clocks (system clock and module clock).

Note: In DLPR mode, selecting a lower voltage via DLPR_VSEL may cause malfunction due to insufficient voltage. Refer to the datasheet for power consumption values under different configurations.

5.3.1.3. Entering/exiting low-power mode

Table 5-3 Summary of Low-Power Mode Entry/Exit

Mode	Entry	Wake-up source	Wake-up clock	Clock requirements for this mode/steps to enter this mode	Impact on clocks when entering this mode	VR mode during low-power operation (controlled by VR_MODE)		Clock requirements for this mode/steps to exit this mode	VR mode after wake-up (hardware automatically switches or maintains based on VR_MODE configuration before entry)	
						VDDD VR	VDDK VR		VDDD VR	VDDK VR
Sleep	WFI or return from interrupt	Any interrupt	Same as before entry	No requirement	CPU HCLK clock OFF; no effect on other clocks or analog clock sources	MR Note 1	OFF	No requirement	MR	OFF
	WFE	Wake up event								
Low-power run	System clock selects MSI, and LPMS=110.	Mode switching Note 2	Switch clock to HSI, subsequent clocks depend on software configuration Note 3	Clock requirements: 1. Software must enable MSI 2. System clock selects MSI and supports MSIDIV division Refer to the "Software Flow" section for steps.	System clock reduced to 2MHz or below	MR	OFF	Clock requirements: 1. Software must enable HSI 2. System clock selects HSI and supports HSIDIV division Refer to the "Software Flow" section for steps.	NA Note 3	OFF
						LPR	OFF		NA Note 3	OFF
Low power sleep	Current mode is "Low Power Run," and SLEEPING=1, and SLEEPDEEP=0, and WFI	Any interrupt	Same as before entering (MSI)	Clock requirements: 1. Software must enable MSI 2. System clock selects MSI and supports MSIDIV division	CPU HCLK clock OFF; no effect on other clocks or analog clock sources	MR	OFF	Clock requirements: 1. Meet entry requirements 2. Wake-up clock is MSI	MR	OFF
	Current mode is "Low Power Run," and SLEEPING=1, and SLEEPDEEP=0, and WFE	Wake up event	Same as before entering (MSI)			LPR	OFF		LPR	OFF
Stop	Current mode is Low Power Run, and LPMS=000/001/010/011; and SLEEPDEEP=1; and	Any EXTI Line configured as interrupt (EXTI	Same as before entering (MSI)	Clock requirements: 1. Software must enable MSI 2. System clock selects	Hardware turns off MSI, HSI, HSE, PLL; LSI and LSE are turned on or off	MR	OFF	Clock requirements: 1. Meet entry requirements 2. Wake-up clock is MSI	MR	OFF
						LPR	OFF		LPR	OFF

Mode	Entry	Wake-up source	Wake-up clock	Clock requirements for this mode/steps to enter this mode	Impact on clocks when entering this mode	VR mode during low-power operation (controlled by VR_MODE)		Clock requirements for this mode/steps to exit this mode	VR mode after wake-up (hardware automatically switches or maintains based on VR_MODE configuration before entry)	
						VDDD VR	VDDK VR		VDDD VR	VDDK VR
	WFI or return from interrupt	register configuration), IWDG reset, NRST;		MSI and supports MSIDIV division	based on software configuration;	OFF	DLPR		LPR	OFF
	Current mode is Low Power Run, and LPMS=000/001/010/011; and SLEEPDEEP=1; and WFE	Any EXTI Line configured as event (EXTI register configuration), IWDG reset, NRST;	Same as before entering (MSI)	Clock requirements: 1. Software needs to enable MSI 2. System clock selects MSI and supports MSIDIV frequency division	Hardware turns off MSI, HSI, HSE, PLL; LSI and LSE are turned on or off based on software configuration;	MR	OFF	Clock requirements: 1. Meet entry requirements 2. Wake-up clock is MSI	MR	OFF
						LPR	OFF		LPR	OFF
						OFF	DLPR		LPR	OFF
	Current mode is normal operation, and LPMS=000/001/010/011; and SLEEPDEEP=1; and WFI or return from interrupt	Any EXTI Line configured as interrupt (EXTI register configuration), IWDG reset, NRST;	HSI	Clock requirements: 1. Software needs to enable HSI, and HSI clock frequency is configured to 8MHz	Hardware turns off MSI, HSI, HSE, PLL; LSI and LSE are turned on or off based on software configuration;	MR	OFF	Clock requirements: 1. Meet the entry requirements 2. Wake-up clock is HSI	MR	OFF
						LPR	OFF		MR	OFF
						OFF	DLPR		MR	OFF
	Current mode is normal operation, and LPMS=000/001/010/011; and SLEEPDEEP=1; and WFE	Any EXTI Line configured as event (EXTI register configuration), IWDG reset, NRST;	HSI	Clock requirements: 1. Software needs to enable HSI, and HSI clock frequency is configured to 8MHz	Hardware turns off MSI, HSI, HSE, PLL; LSI and LSE are turned on or off based on software configuration;	MR	OFF	Clock requirements: 1. Meet the entry requirements 2. Wake-up clock is HSI	MR	OFF
						LPR	OFF		MR	OFF
						OFF	DLPR		MR	OFF
Standby	Current mode is low-power operation, LPMS=101; and SLEEPDEEP=1, and WFI or return from interrupt, or WFE	WKUP IO, IWDG reset, RTC wake-up, TAMP event, NRST;	HSI	Requirements: 1. Software needs to enable MSI	LSE and LSI can be turned on or off; other clocks are turned off.	MR	OFF	Clock requirements: 1. Meet the entry requirements 2. Wake-up clock is HSI	MR	OFF
						LPR	OFF		MR	OFF
						OFF	DLPR		MR	OFF
	Current mode is normal operation, LPMS=101;	WKUP IO, IWDG reset,	HSI		LSE and LSI can be turned on or	MR	OFF		MR	OFF

Mode	Entry	Wake-up source	Wake-up clock	Clock requirements for this mode/steps to enter this mode	Impact on clocks when entering this mode	VR mode during low-power operation (controlled by VR_MODE)		Clock requirements for this mode/steps to exit this mode	VR mode after wake-up (hardware automatically switches or maintains based on VR_MODE configuration before entry)	
						VDDD VR	VDDK VR		VDDD VR	VDDK VR
	and SLEEPDEEP=1, and WFI or return from interrupt, or WFE	RTC wake-up, TAMP event, NRST;		Clock requirements: 1. Software needs to enable HSI, and HSI clock frequency is configured to 8MHz	off; other clocks are turned off.	LPR	OFF	Clock requirements: 1. Meet the entry requirements 2. Wake-up clock is HSI	MR	OFF
						OFF	DLPR		MR	OFF

Note 1: To ensure normal module operation, software must configure the VR state to MR mode before entering sleep mode.

Note 2: The mode switched to from low-power run mode depends on the CPU configuration. However, if the VR operating mode is LPR during LPRUN mode and changes to MR after state switching, more modules can only be enabled after waiting for MR_RDY.

Note 3: After switching from low-power run mode, the VR operating mode depends on the post-switch VR mode configuration.

Note: When VR_MODE is configured to DLPR mode, the VDDD LDO is turned off while the VDDK LDO remains active to generate the power supply required by digital circuits.

5.3.2. Reduce the system clock frequency

In run mode, the frequency of the system clock (SYSCLK, HCLK, PCLK) can be reduced by configuring the prescaler registers. These prescalers can also be used to lower peripheral frequencies before entering sleep or stop mode.

5.3.3. Software Flow

5.3.3.1. Switching from "Run" mode to "Low-Power Run" mode

When switching from RUN mode to LPRUN mode while running on the MSI clock, follow these steps:

- 1) Is the HSI clock enabled? If not enabled, the HSI must be enabled.
- 2) Configure `RCC_CR.MSION=1` in software to enable MSI, wait for `RCC_CR.MSIRDY=1`, then switch the system clock to MSI (configure `RCC_CFGR.SW=3'b101`, supporting MSIDIV prescaling).
- 3) Configure `PWR_CR1.LPMS=3'b110` in software.
- 4) Configure `PWR_CR1.VR_MODE` to LPR mode or MR mode in software (LPR mode has lower power consumption).
- 5) If `FLASH_ACR.LATENCY=0` needs to be configured or the HSI clock needs to be turned off, wait for `PWR_SR.LPRUN_RUN_SWF=0`.

5.3.3.2. Switching from "Low-Power Run" mode to "Run" mode

When switching from LPRUN mode to RUN mode for higher-frequency operation, follow these steps:

1. Configure `RCC_CR.HSION=1` in software to enable HSI.
2. Configure `PWR_CR1.VR_MODE` to MR mode in software.
3. Configure `PWR_CR1.LPMS=3'b000` in software.
4. Wait (with `FLASH_ACR.LATENCY=0` at this time):
 - 1) `RCC_CR.HSIRDY=1`
 - 2) `PWR_SR.MR_RDY=1` and `PWR_SR.LPRUN_RUN_SWF =1`
5. Configure `FLASH_ACR.LATENCY` based on the HSI frequency, then switch the system clock to HSI (configure `RCC_CFGR.SW=3'b000`, supporting HSIDIV prescaling for each HSI frequency). If needed, MSI can be turned off.
6. If switching to another system clock source is required, follow the system clock switching procedure.

5.3.3.3. Transition from "Run" mode to "Stop" low-power mode

The software configuration process for the system to enter Stopx mode from Run mode is as follows:

- 1) Software configures `RCC_CR.HSION=1` to enable HSI, sets HSI frequency to 8 MHz, and switches the system clock to 8 MHz after waiting for `RCC_CR.HSIRDY=1`
- 2) Software configures `PWR_CR1.LPMS=3'b001/3'b010/3'b011/3'b100`
- 3) Software configures `PWR_CR1.VR_MODE` to MR mode/LPR mode/DLPR mode (power consumption decreases in order)

- 4) Software configures PWR_CR1.HSION_CTRL=1 (default value) and sets PWR_CR1.FLS_SLPTIME=2'b01 (5μs)
- 5) Enter low-power mode
- 6) After wake-up, VR_MODE is in MR mode, configured according to PWR_CR1.MR_VSEL; the system clock after wake-up is HSI 8 MHz

5.3.3.4. Transition from "Low-Power Run" mode to "Stop" low-power mode

The software configuration process for the system to enter STOPx mode from LPRUN mode is as follows:

- 1) Currently in low-power run mode, the system clock is MSI;
- 2) Software configures PWR_CR1.LPMS=3'b001/3'b010/3'b011/3'b100;
- 3) Software configures PWR_CR1.VR_MODE to MR mode/LPR mode/DLPR mode (power consumption decreases in order);
- 4) Software configures PWR_CR1.HSION_CTRL=1 (default value) and sets PWR_CR1.FLS_SLPTIME=2'b01 (5μs);
- 5) Enter low-power mode;
- 6) After wake-up, VR_MODE is in LPR mode, configured according to PWR_CR1.LPR_VSEL; the system clock after wake-up is MSI;

5.3.3.5. Transition from "Run" mode to "Standby" low-power mode

The software configuration process for the system to enter STANDBY mode from RUN mode is as follows:

- 1) Software configures RCC_CR.HSION=1 to enable HSI, sets HSI frequency to 8MHz, and switches the system clock to 8MHz after waiting for RCC_CR.HSIRDY=1;
- 2) Software configures PWR_CR1.LPMS=3'b101;
- 3) Software configures PWR_CR1.VR_MODE to DLPR mode;
- 4) Enter low-power mode;
- 5) After wake-up, VR_MODE is in MR mode, configured according to PWR_CR1.MR_VSEL; the system clock after wake-up is HSI 8MHz;

5.3.3.6. Transition from "Low-Power Run" mode to "Standby" low-power mode

The software configuration process for the system to enter STOPx mode from LPRUN mode is as follows:

- 1) Currently in low-power operation mode, system clock is MSI
- 2) Software configures PWR_CR1.LPMS=3'b101
- 3) Software configures PWR_CR1.VR_MODE to DLPR mode
- 4) Enter low-power mode
- 5) After wake-up, VR_MODE is MR mode, MR mode is configured according to PWR_CR1.MR_VSEL; after wake-up, system clock is HSI 8MHz

5.4. PWR registers

5.4.1. Power Control Register 1 (PWR_CR1)

Offset address: 0x00

Reset value: 0x0200 0000

This register is reset upon exiting standby mode.

This register is reset by power reset (POR/BOR), system reset, and PWR soft reset

(RCC_APBSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Re s.	Res.	Res.	STDBY_MRRDY_WAIT[1:0]		HSION_CTRL	Re s.	Re s.	Re s.	DLPR_VSEL[1:0]		LPR_VSEL[1:0]		MR_VSEL[1:0]	
				RW	RW	RW				RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	Re s.	FLS_SLPTIME[1:0]		Res.	VR_MODE[1:0]		DBP	Re s.	Re s.	Res.	Res.	Res.	LPMS[2:0]		
		RW	RW		RW	RW	RW						RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:26	STDBY_MRRDY_WAIT[1:0]	RW	2'b0	After standby mode wake-up, MR ready and HSI clock stable, the required waiting time. 00: 0μs 01: 5μs 10: 10μs 11: 20μs This register is reset to 00 by system reset.
25	HSION_CTRL	RW	1	When waking up from all stop modes (Stop0/1/2/3), the HSI enable time control. 0: Turn on at the same time as VR, that is, HSI is enabled immediately when waking up. 1: After waiting for MR to stabilize, enable HSI; This register is reset to 1 by system reset.
00:22	Reserved	-	-	Reserved
21:20	DLPR_VSEL[1:0]	RW	2'b0	DLPR mode voltage selection. 00: DLPR mode 1, high voltage mode 01: DLPR mode 2, low voltage mode 10: Reserved 11: Reserved
19:18	LPR_VSEL[1:0]	RW	2'b0	LPR mode voltage selection. 00: LPR mode 1, high voltage mode 01: LPR mode 2, medium voltage mode 10: LPR mode 3, medium-low voltage mode 11: LPR mode 4, low voltage mode Note: Voltage values decrease from mode 1->mode 4, from high to low
17:16	MR_VSEL[1:0]	RW	2'b0	MR mode voltage selection. 00: MR mode 1, standard voltage mode (supports maximum frequency 72 MHz) 01: Reserved 10: MR mode 3, low voltage mode (supports maximum frequency 8MHz) 11: Reserved
15:14	Reserved	-	-	Reserved
13:12	FLS_SLPTIME	RW	2'b0	In Stop0/1/2/3 mode wake-up sequence, wait time is required after HSI stabilization and before FLASH operation. 2'b00: 3μs 2'b01: 5μs (recommended configuration) 2'b10: 1μs 2'b11: 0μs
11	Reserved	-	-	Reserved
10:9	VR_MODE[1:0]	RW	2'b0	Regulator operating mode

				00: MR mode 01: LPR mode 10: DLPR mode Others: Reserved (MR mode) Notes: 1. This register resets to 00 after standby mode wake-up. 2. For all stop modes, this register can be configured as 00~10; 3. For standby mode, this register can be configured as 00~10; 4. This register is reset to 00 (MR mode) by system reset.
8	DBP	RW	0	RTC and backup domain register write protection After reset, RTC and backup domain registers are write-protected to prevent accidental writes. This bit must be set to 1 to access RTC and backup domain registers. 0: Access to RTC and backup registers disabled 1: Access to RTC and backup registers enabled
7:3	Reserved	-	-	Reserved
2:0	LPMS[2:0]	RW	0	Low power mode selection 000: Run mode 001: Stop mode 0 (all modules remain powered) 010: Stop mode 1 (SRAM2 powered down) 011: Stop mode 2 (non-wakeup capable modules powered down) 100: Stop mode 3 (SRAM2 and non-wakeup capable modules powered down) 101: Standby mode 110: Low-power run mode This register is reset to 3'b000 (RUN mode) by system reset.

5.4.2. Power control register 2 (PWR_CR2)

Offset address: 0x04

Reset value: 0x0000 0000

Note: This register is a PVD function related register.

This register is reset upon exiting standby mode.

This register is reset by power reset (POR/BOR), system reset, and PWR soft reset

(RCC_APBSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res.	Res.	Res	Res	Res	Res	Res	Res	Res.
.	.	.	.	.	.	.			.	.	.	.	.	.	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	FLT_TIME [2: 0]			FLTE N	PVD_OUT_SE L	PLS[2:0]			Res	Res	Res	PVD E
.	.	.	.	RW	RW	RW	RW	RW	RW	RW	RW		.	.	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:9	FLT_TIME [2: 0]	RW	3'b0	PVD output digital filter time configuration 110: Filter time approximately 30.7ms (1024 LSI or LSE clock cycles) 101: Filter time approximately 3.8ms (128 LSI or LSE clock cycles) 100: Filter time approximately 1.92ms (64 LSI or LSE clock cycles) 011: Filtering time is approximately 480μs (16 LSI or LSE clocks) 010: Filtering time is approximately 120μs (4 LSI or LSE clocks) 001: Filtering time is approximately 60μs (2 LSI or LSE clocks)

				000: Filtering time is approximately 30μs (1 LSI or LSE clock)
8	FLTEN	RW	0	PVD output digital filtering function enabled 0: Disabled 1: Enabled
7	PVD_OUT_SEL	RW	0	PVD digital output control. 0: Comparator digitally selects analog PVD output without filtering and synchronization 1: Comparator digitally selects synchronized and filtered analog PVD output
6:4	PLS[2:0]	RW	3'b0	Voltage rising edge detection threshold (falling edge detection threshold is reduced by 0.1V accordingly). 000: VPVD0 (around 1.8 V) 001: VPVD1 (around 2.0 V) 010: VPVD2 (around 2.2 V) 011: VPVD3 (around 2.4 V) 100: VPVD4 (around 2.8 V) 101: VPVD5 (around 3.1 V) 110: VPVD6 (around 3.7 V) 111: VPVD7 (around 4.2 V) Note: Configure this register when PVD_EN is 0. If SYSCFG_CFGR2.PVDL=1, this register is write-protected. Write protection is reset only when the system is reset.
3:1	Reserved	-	-	Reserved
0	PVDE	RW	0	Voltage detection enable bit 0: Voltage detection disabled 1: Voltage detection enabled If SYSCFG_CFGR2.PVDL=1, PVDE is write-protected. Write protection is reset only when the system is reset.

5.4.3. Power control register 3 (PWR_CR3)

Offset address: 0x08

Reset value: 0x0000 0000

This register is not reset after exiting standby mode.

This register is only reset by POR, not by BOR or system reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Re s.	Re s.	Re s.	Re s.	Res.	Re s.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	Re s.	Re s.	Re s.	Re s.	SRAM1PD STD	AP C	LD_EN	EWU P7	EWU P6	EWU P5	EWU P4	EWU P3	EWU P2	EWU P1	EWU P0
					RW	R W	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10	SRAM1PDSTD	RW	0	SRAM1 power-down control in standby mode. 0: SRAM1 content retained in standby mode 1: SRAM1 powered down in standby mode
9	APC	RW	0	Apply pull-up and pull-down configuration This bit determines whether the I/O pull-up and pull-down configurations defined in the PWR_PUCRx and PWR_PDCRx registers are applied. 0: Do not apply 1: Apply
8	LD_EN	RW	0	Enable power-on loading after waking from standby mode. 0: No power-on loading after waking from standby mode 1: Perform power-on loading after waking from standby mode Note: When this register is set to 1, the wake-up time from standby mode will be extended.
7	EWUP7	RW	0	Enable WKUP7 pin (PB4). This bit is set and cleared by software.

				0: WKUP7 pin is used for general-purpose I/O. Events on the WKUP7 pin do not wake the device from standby mode. 1: WKUP7 pin is used to wake from standby mode, with the wake-up edge determined by WP7[1:0].
6	EWUP6	RW	0	Enable WKUP6 pin (PA13). This bit is set and cleared by software. 0: WKUP6 pin is used for general-purpose I/O. Events on the WKUP6 pin do not wake the device from standby mode. 1: WKUP6 pin is used to wake from standby mode, with the wake-up edge determined by WP6[1:0].
5	EWUP5	RW	0	Enable WKUP5 pin (PA5). This bit is set and cleared by software. 0: The WKUP5 pin is for general purpose I/O. Events on the WKUP5 pin do not wake up the device from standby mode. 1: WKUP5 pin is used to wake from standby mode, with the wake-up edge determined by WP5[1:0].
4	EWUP4	RW	0	Enable WKUP4 pin (PA1). This bit is set and cleared by software. 0: WKUP4 pin is used for general-purpose I/O. Events on the WKUP4 pin do not wake up the device from standby mode. 1: WKUP4 pin is used to wake up from standby mode, with the wake-up edge determined by WP4[1:0].
3	EWUP3	RW	0	Enable the WKUP3 pin (PD4). This bit is set and cleared by software. 0: WKUP3 pin is used for general-purpose I/O. Events on the WKUP3 pin do not wake up the device from standby mode. 1: WKUP3 pin is used to wake up from standby mode, with the wake-up edge determined by WP3[1:0].
2	EWUP2	RW	0	Enable the WKUP2 pin (PC10). This bit is set and cleared by software. 0: WKUP2 pin is used for general-purpose I/O. Events on the WKUP2 pin do not wake up the device from standby mode. 1: WKUP2 pin is used to wake up from standby mode, with the wake-up edge determined by WP2[1:0].
1	EWUP1	RW	0	Enable the WKUP1 pin (PC5). This bit is set and cleared by software. 0: WKUP1 pin is used for general-purpose I/O. Events on the WKUP1 pin do not wake up the device from standby mode. 1: WKUP1 pin is used to wake up from standby mode, with the wake-up edge determined by WP1[1:0].
0	EWUP0	RW	0	Enable the WKUP0 pin (PB10). This bit is set and cleared by software. 0: WKUP0 pin is used for general-purpose I/O. Events on the WKUP pin do not wake up the device from standby mode. 1: WKUP0 pin is used to wake up from standby mode, with the wake-up edge determined by WP0[1:0].

5.4.4. Power control register 4 (PWR_CR4)

Offset address: 0x0C

Reset value: 0x0000 0000 (reset by POR)

This register is not reset after exiting standby mode.

This register is only reset by POR, not by BOR or system reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WP7 [1:0]		WP6 [1:0]		WP5 [1:0]		WP4 [1:0]		WP3 [1:0]		WP2 [1:0]		WP1 [1:0]		WP0 [1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:14	WP7 [1:0]	RW	2'b0	WKUP7 pin (PB4) polarity selection This bit is set and cleared by software. 00: WKUP7 pin rising edge wake-up 01: WKUP7 pin falling edge wake-up 10, 11: Reserved
13:12	WP6 [1:0]	RW	2'b0	WKUP6 pin (PA13) polarity selection This bit is set and cleared by software. 00: WKUP6 pin rising edge wake-up 01: WKUP6 pin falling edge wake-up 10, 11: Reserved
11:10	WP5 [1:0]	RW	2'b0	WKUP5 pin (PA5) polarity selection This bit is set and cleared by software. 00: WKUP5 pin rising edge wake-up 01: WKUP5 pin falling edge wake-up 10, 11: Reserved
9:8	WP4 [1:0]	RW	2'b0	WKUP4 pin (PA1) polarity selection This bit is set and cleared by software. 00: WKUP4 pin rising edge wake-up 01: WKUP4 pin falling edge wake-up 10, 11: Reserved
7:6	WP3 [1:0]	RW	2'b0	WKUP3 pin (PD4) polarity selection This bit is set and cleared by software. 00: WKUP3 pin rising edge wake-up 01: WKUP3 pin falling edge wake-up 10, 11: Reserved
5:4	WP2 [1:0]	RW	2'b0	WKUP2 pin (PC10) polarity selection This bit is set and cleared by software. 00: WKUP2 pin rising edge wake-up 01: WKUP2 pin falling edge wakeup 10, 11: Reserved
3:2	WP1 [1:0]	RW	2'b0	WKUP1 pin (PC5) polarity selection This bit is set and cleared by software. 00: WKUP1 pin rising edge wakeup 01: WKUP1 pin falling edge wakeup 10, 11: Reserved
1:0	WP0 [1:0]	RW	2'b0	WKUP0 pin (PB10) polarity selection This bit is set and cleared by software. 00: WKUP0 pin rising edge wakeup 01: WKUP0 pin falling edge wakeup 10, 11: Reserved

5.4.5. Power status register (PWR_SR)

Offset address: 0x10

Reset value: 0x0000 4200 (reset by POR)

This register is not reset after exiting standby mode.

Except for PVDO, other bits in this register are only reset by POR, and are not reset by BOR or system reset.

The PVDO bit in this register is reset by power reset (POR/BOR), system reset, and PWR soft reset (RCC_APBSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Res.	Res.	Res.	Res.	Re s.	Res.	Re s.	Res	Res	Res	Res	Res	Res	Res	PV DO.
															R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	VCC_P ORF	PINRST WUF	IWDGRST WUF	LPRUN_RU N SWF	Re s.	MR_RDY	S BF	WU F7	WU F6	WU F5	WU F4	WU F3	WU F2	WU F1	WU F0
	R	R	R	R		R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	PVDO	R	0	PVD detection result output.

				0: The detected V _{CC} exceeds the PVD-selected comparison threshold 1: The detected V _{CC} is below the PVD-selected comparison threshold
15	Reserved	-	-	Reserved
14	VCC_PORF	R	1	V _{CC} power-on reset flag. 0: No V _{CC} power-on reset occurred 1: V _{CC} power-on reset occurred
13	PINRST_WUF	R	0	PIN reset wake-up standby mode flag. This bit is set to 1 by hardware. To clear it, only POR or setting the CPINRSTWUF bit in the PWR_SCR register to 1 can be used. 0: No wake-up event occurred 1: Wake-up event occurred
12	IWDGRST_WUF	R	0	IWDG reset wake-up standby mode flag. This bit is set to 1 by hardware. To clear it, only POR or setting the CIWDGRSTWUF bit in the PWR_SCR register to 1 can be used. 0: No wake-up event occurred 1: Wake-up event occurred
11	LPRUN_RUN_SWF	R	0	Mode switch flag between LPRUN and RUN. Set to 1 indicates RUN mode. During the switching process, the value changes as described below: --When switching from LPRUN to RUN: 0: System is in progress of switching from LPRUN mode to RUN mode 1: System has completed switching from LPRUN mode to RUN mode --When switching from RUN to LPRUN: 0: System has completed switching from RUN mode to LPRUN mode 1: System is in progress of switching from RUN mode to LPRUN mode
10	Reserved	-	-	Reserved
9	MR_RDY	R	1	Used to indicate the working status of MR, 1: MR mode is operating stably; 0: MR mode disabled; When VR_MODE is in MR mode, this register remains set. When switching to other modes (LPR or DLPR), this register is cleared.
8	SBF	R	0	Standby mode flag. This bit is set by hardware and can only be cleared by POR or setting the CSBF bit in the PWR_SCR register. 0: System has not entered standby mode 1: System has entered standby mode
7	WUF7	R	0	WKUP7 standby mode wake-up flag. This bit is set by hardware and can only be cleared by POR or setting the CWUF7 bit in the PWR_SCR register. 0: No wake-up event occurred 1: Wake-up event received
6	WUF6	R	0	WKUP6 standby mode wake-up flag. This bit is set by hardware and can only be cleared by POR or setting the CWUF6 bit in the PWR_SCR register. 0: No wake-up event occurred 1: Wake-up event received
5	WUF5	R	0	WKUP5 standby mode wake-up flag. This bit is set by hardware and can only be cleared by POR or setting the CWUF5 bit in the PWR_SCR register. 0: No wake-up event occurred 1: Wake-up event received
4	WUF4	R	0	WKUP4 standby mode wake-up flag. This bit is set by hardware and can only be cleared by POR or setting the CWUF4 bit in the PWR_SCR register. 0: No wake-up event occurred 1: Wake-up event received
3	WUF3	R	0	WKUP3 standby mode wake-up flag. This bit is set to 1 by hardware and can only be cleared by POR or setting the CWUF3 bit in the PWR_SCR register.

				0: No wake-up event occurred 1: Wake-up event received
2	WUF2	R	0	WKUP2 wake-up standby mode flag. This bit is set to 1 by hardware and can only be cleared by POR or setting the CWUF2 bit in the PWR_SCR register. 0: No wake-up event occurred 1: Wake-up event received
1	WUF1	R	0	WKUP1 wake-up standby mode flag. This bit is set to 1 by hardware and can only be cleared by POR or setting the CWUF1 bit in the PWR_SCR register. 0: No wake-up event occurred 1: Wake-up event received
0	WUF0	R	0	WKUP0 wake-up standby mode flag. This bit is set to 1 by hardware and can only be cleared by POR or setting the CWUF0 bit in the PWR_SCR register. 0: No wake-up event occurred 1: Wake-up event received

5.4.6. Power status clear register (PWR_SCR)

Offset address: 0x14.

Reset value: 0x0000 0000

This register is only reset by POR, not by BOR or system reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Res.	Res.	Res.	Re s.	Re s.	Re s.	Re s.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	CVCCP ORF	CPINRST WUF	CIWDGRS TWUF	Re s.	Re s.	Re s.	CS BF	CW UF7	CW UF6	CW UF5	CW UF4	CW UF3	CW UF2	CW UF1	CW UF0
	W	W	W				W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	Reserved
14	CVCCPORF	W	0	VCC power-on reset flag clear signal. Set this bit to 1 to clear the VCC_PORF signal in PWR_SR.
13	CPINRSTWUF	W	0	PINRST_WUF wakeup flag clear signal. Set this bit to 1 to clear the PINRST_WUF signal in PWR_SR.
12	CIWDGRSTWUF	W	0	IWDGRST_WUF wakeup flag clear signal. Set this bit to 1 to clear the IWDGRST_WUF signal in PWR_SR.
11:9	Reserved	-	-	Reserved
8	CSBF	W	0	SBF wakeup flag clear signal. Set this bit to 1 to clear the SBF signal in PWR_SR.
7	CWUF7	W	0	WKUP7 wakeup flag clear signal. Set this bit to 1 to clear the WUF7 signal in PWR_SR.
6	CWUF6	W	0	WKUP6 wakeup flag clear signal. Set this bit to 1 to clear the WUF6 signal in PWR_SR.
5	CWUF5	W	0	WKUP5 wakeup flag clear signal. Set this bit to 1 to clear the WUF5 signal in PWR_SR.
4	CWUF4	W	0	WKUP4 wakeup flag clear signal. Set this bit to 1 to clear the WUF4 signal in PWR_SR.
3	CWUF3	W	0	WKUP3 wakeup flag clear signal. Set this bit to 1 to clear the WUF3 signal in PWR_SR.
2	CWUF2	W	0	WKUP2 wakeup flag clear signal. Set this bit to 1 to clear the WUF2 signal in PWR_SR.
1	CWUF1	W	0	WKUP1 wakeup flag clear signal. Set this bit to 1 to clear the WUF1 signal in PWR_SR.
0	CWUF0	W	0	WKUP0 wakeup flag clear signal. Set this bit to 1 to clear the WUF0 signal in PWR_SR.

Registers from PWR_PUCRA to PWR_PDCRD control the pull-up/pull-down of each port in standby mode. To enable these registers, first configure PWR_CR3.APC=1.

When configuring these registers, note the following:

1. If PD13 is used as a reset pin, configure PWR_PUCRD.PU13=1 before entering standby mode.

5.4.7. Power PortA Pull-Up Control Register (PWR_PUCRA)

Offset address: 0x20

Reset value: 0x0000 0000

This register is not reset after exiting standby mode.

This register is reset by power reset (POR/BOR) and PWR soft reset (RCC_APBRRSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PU13	PU12	PU11	PU10	PU9	Res.	Res.	Res.	PU5	Res.	PU3	PU2	PU1	PU0
		RW	RW	RW	RW	RW				RW		RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13:0	PUy	RW	0	PortA Pull-Up Control Bit (y=0~15) Setting this bit to 1 while APC bit in PWR_CR3 is 1 activates the pull-up for PA[y]. If the corresponding PDy bit is also set to 1, the pull-up is not activated. Notes: 1. PA4, used as SWDIO, is configured as pull-up by GPIO and is not affected by this register. 2. PA6~8 are VBKP domain IOs and are not controlled by this register.

5.4.8. Power PortA Pull-Down Control Register (PWR_PDCRA)

Offset Address: 0x24

Reset value: 0x0000 0000

This register is not reset after exiting standby mode.

This register is reset by power reset (POR/BOR) and PWR soft reset (RCC_APBRRSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PD13	PD12	PD11	PD10	PD9	Res.	Res.	Res.	Res.	PD4	PD3	PD2	PD1	PD0
		RW	RW	RW	RW	RW					RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13:0	PDy	RW	0	PortA Pull-Down Control Bit (y=0~15) Setting this bit to 1 while APC bit in PWR_CR3 is 1 activates the pull-down for PA[y]. Notes: 1. PA5, used as SWCLK, is configured as pull-down by GPIO and is not affected by this register. 2. PA6~8 are VBKP domain IOs and are not controlled by this register.

5.4.9. Power PortB pull-up control register (PWR_PUCRB)

Offset address: 0x28

Reset value: 0x0000 0000

This register is not reset after exiting standby mode.

This register is reset by power reset (POR/BOR) and PWR soft reset (RCC_APBSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PU15	PU14	PU13	PU12	PU11	PU10	PU9	PU8	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PUy	RW	0	Port B pull-up control bits (y=0~15) Setting this bit to 1 when the APC bit in the PWR_CR3 register is set to 1 activates the pull-up on PB[y]. If the corresponding PDy bit is also set to 1, the pull-up is not activated.

5.4.10. Power PortB pull-down control register (PWR_PDCRB)

Offset address: 0x2C

Reset value: 0x0000 0000

This register is not reset after exiting standby mode.

This register is reset by power reset (POR/BOR) and PWR soft reset (RCC_APBSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PDy	RW	0	Port B pull-down control bits (y=0~15) Setting this bit to 1 when the APC bit in the PWR_CR3 register is set to 1 activates the pull-down on PB[y].

5.4.11. Power PortC pull-up control register (PWR_PUCRC)

Offset address: 0x30

Reset value: 0x0000 0000

This register is not reset after exiting standby mode.

This register is reset by power reset (POR/BOR) and PWR soft reset (RCC_APBSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PU15	PU14	PU13	PU12	PU11	PU10	PU9	PU8	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PUy	RW	0	Port C pull-up control bits (y=0~15) Setting this bit to 1 when the APC bit in the PWR_CR3 register is set to 1 activates the pull-up on PC[y]. If the corresponding PDy bit is also set to 1, the pull-up is not activated.

5.4.12. Power PortC Pull-Down Control Register (PWR_PDCRC)

Offset address: 0x34

Reset value: 0x0000 0000

This register is not reset after exiting standby mode.

This register is reset by power reset (POR/BOR) and PWR soft reset (RCC_APBSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PDy	RW	0	Port C Pull-Down Control Bits (y=0~15) Setting this bit to 1 when the APC bit in the PWR_CR3 register is set to 1 activates the pull-down on PC[y].

5.4.13. Power PortD Pull-Up Control Register (PWR_PUCRD)

Offset Address: 0x38

Reset Value: 0x0000 2000

This register is not reset after exiting standby mode.

This register is reset by power reset (POR/BOR) and PWR soft reset (RCC_APBRRSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PU15	PU14	PU13	PU12	PU11	PU10	PU9	PU8	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PUy	RW	0x2000	Port D Pull-Up Control Bits (y=0~13) Setting this bit to 1 when the APC bit in the PWR_CR3 register is set to 1 activates the pull-up on PD[y]. If the corresponding PDy bit is also set to 1, the pull-up is not activated. Note: Before entering Standby mode, the pull-up on the reset PIN (PD13) needs to be configured.

5.4.14. Power PortD Pull-Down Control Register (PWR_PDCRD)

Offset Address: 0x3C

Reset value: 0x0000 0000

This register is not reset after exiting standby mode.

This register is reset by power reset (POR/BOR) and PWR soft reset (RCC_APBRRSTR1.PWRRST).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PDy	RW	0	Port D pull-down control bit (y=0~15) Setting this bit while APC bit in PWR_CR3 register is set to 1 activates the pull-down of PD[y].

6. Reset and Clock Control (RCC)

6.1. Reset

Two resets are designed: power reset and system reset.

6.1.1. Power reset

A power reset occurs in the following situations:

- Power-on/power-down reset (POR/PDR)
- Brown-out reset (BOR)

Power-on reset and under-voltage reset will set all other registers to their reset values except registers in the RTC domain and some registers of the PWR controller.

When exiting standby mode, all registers in the VDDD_CORE, VDDD_PERI, VDDD_SRAM2, and VDDD_SRAM1 domains are set to their reset values. Registers in other domains (RTC, TAMP, WKUP, IWDG, and standby mode control) remain unaffected.

6.1.2. System reset

A system reset sets all registers to their reset values except the reset flags in the clock control/status register and the registers in the RTC domain.

System reset is generated when one of the following events occurs:

- NRST Pin Reset
- Windowed watchdog reset (WWDG)
- Independent watchdog reset (IWDG)
- SYSRESETREQ software reset
- Option byte loading reset (OBL)
- Low-power mode security reset

The reset source can be identified by checking the reset identification bit of the RCC_CSR register.

Note: The term "system reset" appearing in subsequent module descriptions includes the aforementioned "power reset" and "system reset."

6.1.2.1. PIN pin reset (external reset)

Through option byte (NRST_MODE bit) loading, the NRST pin can be configured into the following modes (see option byte description for specific configurations):

- Reset input

In this mode, any valid reset signal on the NRST pin is passed to internal logic, but resets generated internally by the chip are not output on the NRST pin.

In this mode, the GPIO functionality (PD13) is not available.

The NRST pin has a glitch filter. The design ensures that the NRST signal must be at least 40us wide; signals shorter than this width will be filtered out. The glitch filter circuit uses HSI_10M for

counting. By default, this clock source is turned off. A low level on NRST turns on this clock, while a high level turns it off.

■ GPIO

In this mode, the pin can be used as a standard GPIO, i.e., PD13. The reset function on the pin is disabled. Reset is only possible from device internal reset sources and it is not propagated to the pin.

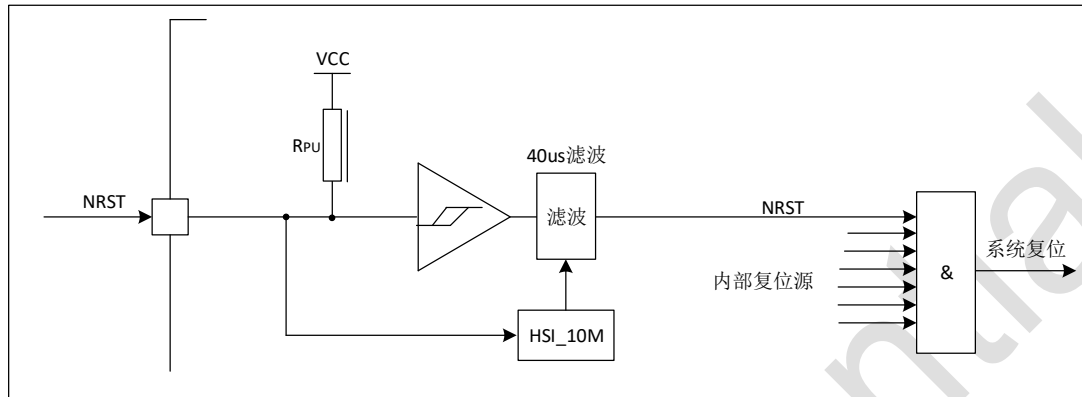


Figure 6-1 reset circuit

6.1.2.2. Watchdog reset

Refer to the "Independent Watchdog (IWDG)" and "System Window Watchdog (WWDG)" sections.

6.1.2.3. Software reset

A software reset can be achieved by setting the SYSRESETREQ bit in the ARM Cortex®-M0+ Interrupt and Reset Control Register.

6.1.2.4. Option Byte Load Reset

The software can generate an option byte loading reset by configuring FLASH_CR.OBL_LAUNCH=1, thereby initiating option byte loading.

6.1.2.5. Low-power mode security reset

To prevent that critical applications mistakenly enter a low-power mode, three low-power mode security resets are available. If enabled in option bytes, the resets are generated in the following conditions:

■ Entering Standby mode

This type of reset can be enabled by clearing the nRST_STDBY bit in the user option byte. In this case, whenever a Standby mode entry sequence is successfully executed, the device is reset instead of entering Standby mode.

■ Entering stop mode

This type of reset can be enabled by clearing the nRST_STOP bit in the user option byte. In this case, whenever a Stop mode entry sequence is successfully executed, the device is reset instead of entering Stop mode.

6.1.3. RTC domain reset

The RTC domain has two special resets. A RTC domain reset is generated when one of the following events occurs:

- A software reset, triggered by setting the BDRST bit in the RTC domain control register (RCC_BDCR) to 1.
- After the power supply V_{CC} has been powered down, it is powered on again.

The RTC domain reset only affects the LSE oscillator, RTC, backup registers, and RCC RTC domain control register.

6.2. Clocks

6.2.1. Clock sources

The system clock sources for this product include HSI, MSI, HSE, PLL, LSI, and LSE. It also includes HSI10M, which is used for pin reset filtering and controlling the FM_SLEEP signal during Flash low-power operation (system clock is LSI/LSE).

6.2.1.1. HSI clock

The HSI clock signal is generated from an internal RC oscillator. Frequencies include 8MHz, 16MHz, 24MHz, 48MHz, and 64MHz.

The advantage of the HSI RC oscillator is its lower cost (no external components required). It also has a shorter startup time compared to the HSE crystal oscillator. However, even after calibration, it is less accurate than an oscillator using a frequency reference such as quartz crystal or ceramic resonator.

After waking from Stop mode, the HSISYS clock derived from HSI can be selected as the system clock. It can also be used as a backup clock source (auxiliary clock) if the HSE crystal oscillator fails. Refer to "Clock Security System (CSS)."

The HSIRDY flag in the clock control register (RCC_CR) indicates whether the HSI RC is stable. At startup, the HSI becomes available only after the hardware sets this bit to 1.

The HSI RC can be turned on or off via the HSION bit in the clock control register (RCC_CR). It is turned on by default after power-on reset.

HSI calibration

Due to variations in manufacturing processes, the RC oscillator frequency differs across chips. To compensate for this variation, each device is factory-calibrated, and the accuracy under various conditions can be found in the datasheet.

After POR/BOR reset, the HSI frequency and corresponding calibration values are loaded into the {HSI_FS_OPCR[2:0], HSI_TRIMCR[12:0]} bits of the internal clock source calibration register (RCC_ICSCR).

Voltage or temperature changes in the application may affect the HSI frequency of the RC oscillator. Software can adjust it by configuring the HSI_TRIMCR[8:0] bits in the internal clock source calibration register (RCC_ICSCR).

For more details on how to measure HSI frequency variations, refer to "TIM16/TIM17 Measuring Internal and External Clocks."

6.2.1.2. MSI clock

The MSI 2 MHz clock signal is generated from an internal RC oscillator. The MSI RC oscillator provides a low-cost (no external components) and low-power clock source. This oscillator serves as the clock in LPRUN low-power mode to reduce power consumption.

The MSIRDY flag in the clock control register (RCC_CR) indicates whether the MSI RC is stable. At startup, the MSI can only be used after the hardware sets this bit to 1. If MSIRDYIE is enabled in the clock interrupt enable register (RCC_CIER), an interrupt can be generated.

The MSI RC can be turned on or off via the MSION bit in the clock control register (RCC_CR).

MSI calibration

Due to variations in manufacturing processes, the RC oscillator frequency differs across chips. To compensate for this variation, each device is factory-calibrated, and the accuracy under various conditions can be found in the datasheet.

After POR/BOR reset, the MSI calibration value is loaded into the MSI analog circuitry.

6.2.1.3. HSE clock

The high-speed external clock signal (HSE) has two clock sources:

- HSE external crystal/ceramic resonator
- HSE user external clock

The resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and startup stabilization time.

The loading capacitance values must be adjusted according to the selected oscillator.

External crystal/ceramic resonator (HSE crystal)

The 4 ~ 32 MHz external oscillator offers the advantage of very high accuracy.

Refer to the electrical characteristics section of the datasheet for more details.

The HSERDY flag in the clock control register (RCC_CR) indicates whether the HSE oscillator is stable. At startup, this clock can only be used after the hardware sets this bit to 1. If HSERDYIE is enabled in the clock interrupt enable register (RCC_CIER), an interrupt can be generated.

The HSE_DRIVER[1:0] bits in the external clock source configuration register (RCC_ECSCR) allow configuring the crystal driver strength before operation to achieve the optimal balance between robustness, short startup time, and low power consumption.

The HSE crystal can be turned on or off via the HSEON bit in the clock control register (RCC_CR).

External source (HSE bypass)

In this mode, an external clock source must be provided. The maximum frequency must not exceed 32 MHz. This mode is selected by setting the HSEBYP and HSEON bits in the clock control register (RCC_CR) to 1. On devices with OSC_IN and OSC_OUT pins, an external clock signal (square, sine, or triangular wave) with a duty cycle of ~40-60% must drive the OSC_IN pin, depending on the frequency (refer to the datasheet). The OSC_OUT pin can be used as a GPIO.

For details on pin availability, refer to the pinout section in the datasheet.

6.2.1.4. PLL clock

The internal PLL multiplies the HSI- or HSE-based clock frequency acquired at its input to generate independent clock outputs. Refer to the datasheet for the allowable input frequency range of the reference clock.

To modify the PLL configuration, follow these steps:

1. Disable the PLL by setting the PLLON bit to 0 in the clock control register (RCC_CR).
2. Wait until PLLRDY is cleared. The PLL is now completely stopped.
3. Modify the reference clock and multiplication factor as needed.
4. Re-enable the PLL by setting the PLLON bit to 1.

If interrupts are enabled in the clock interrupt enable register (RCC_CIER), an interrupt can be generated when the PLL is ready.

6.2.1.5. LSI clock

The internal low-speed RC clock LSI can operate as a low-power clock source in Stop and Standby modes for modules such as the independent watchdog (IWDG) and RTC. The clock frequency is 32.768 kHz. Refer to the electrical characteristics section of the datasheet for more details.

The LSI RC can be turned on or off via the LSION bit in the RTC domain control register (RCC_BDCR).

The LSIRDY flag in the RTC domain control register (RCC_BDCR) indicates whether the LSI oscillator is stable. At startup, this clock can only be used after the hardware sets this bit to 1. If interrupts are enabled in the clock interrupt enable register (RCC_CIER), an interrupt can be generated.

6.2.1.6. LSE clock

The LSE crystal is a 32.768 kHz crystal or ceramic resonator that serves as a clock source for the real-time clock (RTC) to provide clock/calendar or other timing functions, offering low power consumption and high accuracy.

The LSE crystal is turned on and off via the LSEEN bit in the RTC domain control register (RCC_BDCR). The LSE_DRIVER[1:0] bits in the RTC domain control register (RCC_BDCR) allow configuring the crystal drive strength before operation to achieve the best balance between robustness, short startup time, and low power consumption.

The LSE RDY flag in the RTC domain control register (RCC_BDCR) indicates whether the LSE crystal is stable. After hardware sets this bit during startup, the LSE crystal output clock signal becomes available. If interrupts are enabled in the clock interrupt enable register (RCC_CIER), an interrupt can be generated.

External source (LSE bypass)

In this mode, an external clock source must be provided. The maximum frequency must not exceed 1 MHz. This mode is selected by setting the LSEBYP bit to 1 in the RTC domain control register (RCC_BDCR) and configuring the LSEEN bit as enabled in the RTC domain control register

(RCC_BDCR). An external clock signal (square, sine, or triangular wave) with approximately 50% duty cycle must be used to drive the OSC32_IN pin.

6.2.1.7. HSI10M clock

This clock serves as a low-accuracy, low-power clock for filtering the NRST pin count and controlling the FM_SLEEP signal during Flash low-power operation (when the system clock is LSI/LSE). This clock is enabled and disabled by hardware.

6.2.2. Clock structure

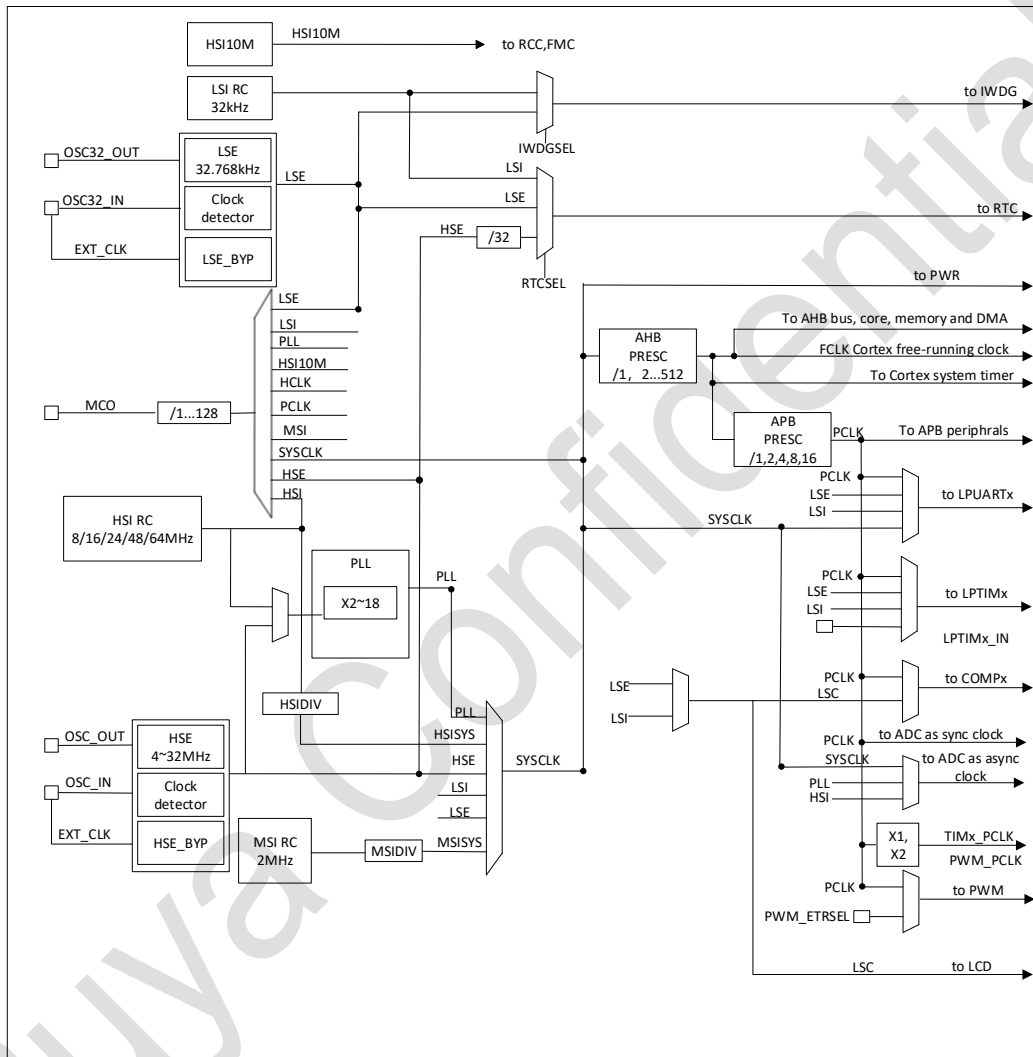


Figure 6-2 Clock Structure

6.2.3. System Clock (SYSCLK) Selection

One of the following clocks can be selected as the system clock (SYSCLK):

- LSI
- LSE
- HSISYS
- HSE
- PLL

The system clock maximum frequency is 72 MHz. After a system reset, the HSISYS clock derived from the HSI oscillator is selected as the system clock. When directly using HSI or using a clock source via PLL as the system clock, and when switching to low-power modes, this clock source cannot be stopped.

Switching from one clock source to another is only possible when the target clock source is ready (the clock is stable after startup delay or PLL locking). The original clock source can be disabled (if needed) only after the switch is successful (the SWS in the clock configuration register RCC_CFGR has been updated to the target clock source).

6.2.4. Clock Security System (CSS)

6.2.4.1. HSE CSS

The HSE clock security system can be activated by software via the configuration of RCC_CR.HSECSSON. In this case, after the HSE starts, the clock detection function is turned on. When the HSE is turned off, the clock detection function is turned off.

If a clock failure is detected on HSE, the HSE is automatically disabled, the clock failure event is sent to the brake inputs of TIM1 (advanced timer), TIM15/TIM16/TIM17 (general-purpose timers), and PWM, and an interrupt is generated to notify the software of the failure (Clock Security System Interrupt CSSI), allowing the MCU to perform recovery operations. CSSI is linked to the NMI (Non-Maskable Interrupt) exception vector of the Cortex®-M0+.

Note: Once CSS is enabled and if an HSE clock failure occurs, a CSS interrupt is generated, and an NMI is automatically triggered. The NMI will be executed indefinitely unless the CSS interrupt pending bit is cleared. As a consequence, in the NMI ISR user must clear the CSS interrupt by setting the CSSC bit in the Clock interrupt register (RCC_CIR).

If HSE is directly or indirectly used as the system clock (indirect meaning: used as the input to the PLL, and the PLL is used as the system clock), a clock failure will cause the system clock to automatically switch to HSI, and the HSE will be disabled. If the HSE is the input clock for the PLL when a clock failure occurs, the PLL will also be disabled.

6.2.4.2. LSE CSS

The LSE clock security system can be activated by software by configuring RCC_BDCR.LSECSSON. In this case, after the LSE starts, the clock detection function is turned on. When the LSE is turned off, the clock detection function is turned off.

If a clock failure is detected on LSE, the LSE is automatically disabled, the clock failure event is sent to the brake inputs of TIM1 (advanced timer) and TIM16/TIM17 (general-purpose timers), and an interrupt is generated to notify the software of the failure (Clock Security System Interrupt CSSI), allowing the MCU to perform recovery operations. CSSI is linked to the NMI (Non-Maskable Interrupt) exception vector of the Cortex®-M0+.

Note: Once LSECSS is enabled and if an LSE clock failure occurs, a CSS interrupt is generated, and an NMI is automatically triggered. The NMI will be executed indefinitely unless the LSECSS interrupt pending bit is cleared. Therefore, in the NMI handler, the LSECSS interrupt must be cleared by setting the LSECSSC bit in the clock interrupt register (RCC_CICR).

If the LSE is used as the system clock, a clock failure will cause the system clock to automatically switch to the LSI, while the LSE is turned off.

6.2.5. TIM16/TIM17 measures internal and external clocks

Due to factors such as temperature, voltage, process and production, the frequency of internal clock sources (such as HSI, LSI, etc.) drift. Therefore, it is necessary to take some necessary measures to calibrate the frequency drift according to the changes of the external working environment of the system.

The basic idea for handling clock drift is: when the external environment of the system changes, dynamically measure the internal clock of the chip in real-time to detect and identify issues. Then, the internal clock calibration parameters are fine-tuned by software, so as to achieve the purpose of dynamic calibration.

The frequencies of all on-board clock sources can be indirectly measured through the input capture of TIM16 and TIM17 channel 1.

TIM16

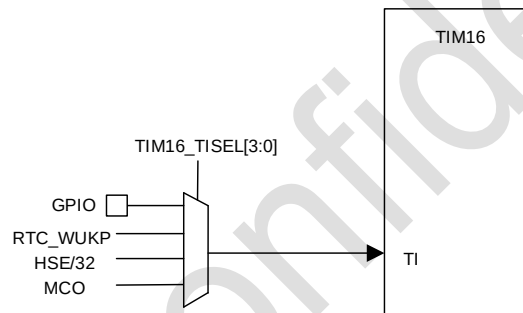


Figure 6-3 Measuring clock through TIM16 sampling

By configuring the TI1SEL[3:0] bit field in the TIM16_TISEL register, the clock selected for TIM16's input capture channel 1 can be one of the following:

- GPIO (refer to the multiplex function mapping in the device datasheet)
- LSI clock
- MCO output clock
- RTC wake-up interrupt signal

The last option requires enabling the RTC interrupt. The MCO output clock is controlled by the MCOSEL[3:0] bit field in the clock configuration register (RCC_CFGR). All clock sources can be selected for the MCO pin.

TIM17

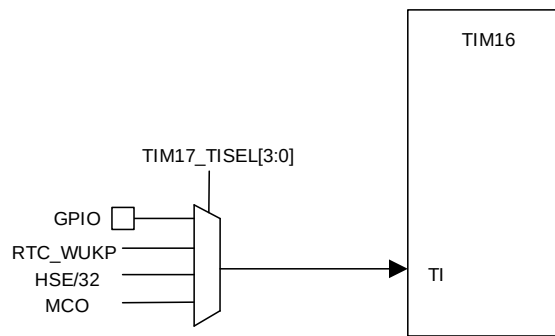


Figure 6-4 Measuring clock through TIM17 sampling

By configuring the TI1SEL[3:0] bit field in the TIM16_TISEL register, the clock selected for TIM16's input capture channel 1 can be one of the following:

- GPIO (refer to the multiplex function mapping in the device datasheet)
- LSI clock
- MCO output clock
- RTC wake-up interrupt signal

The last option requires enabling the RTC interrupt. The MCO output clock is controlled by the MCOSEL[3:0] bit field in the clock configuration register (RCC_CFGR). All clock sources can be selected for the MCO pin.

6.2.5.1. HSI calibration

HSI clock calibration is divided into two parts: clock detection and clock calibration.

Clock detection

For TIM15, TIM16, and TIM17, the primary purpose of connecting the LSE to channel 1 input capture is to enable precise measurement of the HSISYS clock (derived from HSI and selected as the system clock). Counting the HSISYS clock pulses between consecutive edges of the LSE clock (time reference) allows measurement of the HSISYS (and HSI) clock period. This measurement determines the HSI oscillator frequency with accuracy nearly equal to that of the 32.768 kHz quartz crystal used for the LSE oscillator (typically within tens of ppm). The HSI oscillator can then be adjusted to compensate for deviations from the target frequency caused by manufacturing, process, temperature, and/or voltage variations.

The HSI oscillator has dedicated calibration bits for this purpose, which are user-accessible. The fundamental principle is based on relative measurement (e.g., HSISYS/LSE ratio): thus, the measurement accuracy is closely tied to the ratio of the two clock sources. Increasing the ratio allows improving the measurement accuracy.

The HSE clock (divided by 32), generated by the HSE oscillator and used as a time reference, is another optimal method to achieve excellent HSI frequency measurement accuracy. It is recommended for use when the LSE clock is unavailable.

To further improve HSI oscillator calibration accuracy, the following measures are recommended to enhance frequency measurement precision, either individually or in combination:

- Set the HSISYS divider value to 1, making the HSISYS frequency equal to the HSI frequency.
- Average the results of multiple consecutive measurements.

- Use the timer's input capture prescaler value (capturing once every eight cycles at most).
- Use the LSE clock from the RTC and RTC wake-up interrupt signal as the time reference.

The last point significantly increases the reference period for counting HSI clock pulses, thereby improving the accuracy of a single measurement. For this, the RTC wake-up interrupt must be enabled.

Clock calibration

Once an HSI clock anomaly is detected, notify the software for handling via an interrupt. The software fine-tunes the internal clock trimming parameters (by updating `RCC_ICSCR.HSI_TRIMCR[8:0]`) to achieve dynamic calibration.

Note: When fine-tuning the HSI clock calibration value, ensure it matches the frequency value of `RCC_ICSCR.HSI_FS_OPCR`.

6.2.5.2. LSI calibration

Like the HSI, the LSI clock frequency is also subject to drift due to voltage, temperature, process, and manufacturing variations. The calibration of LSI is performed by using HSE or HSI with a large frequency difference, and the calibration method is similar to HSI.

The calibration of LSI involves selecting the LSI clock for input capture of TIM16/TIM17, with HSE as the system clock source. The number of HSE clock pulses between consecutive edges of the LSI signal is counted by TIM16/TIM17, representing the LSI clock period.

In principle, it still relates to the relative frequency ratio, i.e., the HSE/LSI frequency ratio: calibration accuracy is closely tied to this, and a larger ratio value yields better measurement results.

6.2.6. Clock Output

In order to facilitate board-level applications, save BOM costs, and meet debugging requirements, the device needs to provide clock output functions. That is, the MCO signal (parallel frequency division) in the table below is used to realize the clock output function through the multiplexing function of GPIO.

Table 6-1 Output clock selection

Clock Source/Internal Clock	MCO output clock source
HSI	√
HSE	√
PLL	√
LSE	√
LSI	√
HSI_10M	√
SYSCLK	√
HCLK	√
PCLK	√

Note: When the MCO clock source is switched and the GPIO AF function is selected as the initial stage of the MCO, the MCO may generate glitches and need to be avoided for this period of time.

6.2.7. Module clock types

Clock types:

- SYSCLK: The clock configured by `RCC_CFGR.SW` as the clock source, up to 72 MHz
- HCLK: The clock after HPRE division of SYSCLK, up to 72 MHz
- PCLK: The clock after PPRE division of HCLK, up to 72 MHz

- **RTCCLK:** The core clock of the RTC module, with the clock source configured by the `RCC_BDCR.RTCSEL` register, up to 1 MHz, commonly 32 kHz
- **PWM_CLK:** The core clock of the PWM module, with the clock source selected internally by the PWM module
- **LPTIM_CLK:** The core clock of the LPTIM module, with the clock source configured by the `RCC_CCIPR.LPTIM*SEL` register, up to 72 MHz, commonly 32 kHz
- **IWDG_CLK:** The core clock of the IWDG module, with the clock source configured by the `RCC_CCIPR.IWDGSEL` register
- **LPUART_CLK:** The core clock of the LPUART module, with the clock source configured by the `RCC_CCIPR.LPUART*SEL` register, up to 72 MHz
- **CMP_CLK:** The core clock of the COMP1/COMP2 module, with the clock source configured by the `RCC_CCIPR.COMP*SEL` register, up to 72 MHz, commonly 32 kHz
- **PVD_CLK:** The core clock of the PVD module, with the clock source configured by the `RCC_CCIPR.PVDSEL` register, up to 72 MHz, commonly 32 kHz
- **ADC_CLK:** The core clock of the SARADC module, with the clock source and division factor configured by the `ADC_CCR.CKMODE` and `ADC_CCR.PRESC` registers
- **LCD_CLK:** The core clock of the LCD module, with the clock source configured by the `RCC_BDCR.LSCSEL` register to select LSI or LSE, at a frequency of 32 kHz

6.3. RCC registers

The registers of this module can be accessed in word (32-bit), half-word (16-bit), and byte (8-bit) formats.

6.3.1. RCC Clock Control Register (RCC_CR)

Offset address: 0x00

Reset value: 0x0000 0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Re s.	Re s.	Re s.	Re s.	Res.	PLL RDY	PLL ON	Res.	Res.	Re s.	Re s.	HSECSS ON	HSEB YP	HSER DY	HSEO N
						R	RW					RS	RW	R	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	Re s.	HSIDIV[2:0]			HSIR DY	MSIR DY	HSIO N	MSIO N	HSIKER ON	Re s.	Re s.	Res.	MSIDIV[2:0]		
		RW	RW	RW	R	R	RW	RW	RW				RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	PLLRDY	R	0	PLL clock ready flag. Set by hardware to indicate that the PLL clock is locked. 0: PLL not locked 1: PLL locked
24	PLLON	RW	0	PLL enabled Set and cleared by software. This bit is cleared by hardware when the system enters any stop or standby mode. This bit cannot be cleared when the PLL clock is being used (SWS is PLL) or is about to be used as the system clock (SW selects PLL). 0: PLL off 1: PLL enabled
23: 20	Reserved	-	-	Reserved

19	HSECSSON	RS	0	Clock security system enable. Set by software to enable the clock security system. When this bit is set, if the HSE is ready, hardware performs clock detection. If a clock failure is detected, hardware disables clock detection. This bit can only be set; clearing is only possible via reset. 0: Clock safety system off (clock detection off) 1: Clock security system enabled (clock detection is on if HSE is ready, otherwise off)
18	HSEBYP	RW	0	Bypass the HSE external crystal and select the pin input clock. Set and cleared by software to bypass the oscillator with an external clock. The external clock must be enabled using HSEON. The HSEBYP bit can only be set when the HSE external crystal is disabled. 0: HSE external crystal not bypassed 1: HSE external crystal bypassed, external pin input clock selected
17	HSERDY	R	0	HSE clock ready flag Set by hardware to indicate that the HSE is ready. 0: HSE not ready 1: HSE ready Note: When HSEON is cleared, HSERDY is immediately cleared.
16	HSEON	RW	0	HSE clock enabled Set and cleared by software. Enter stop mode, hardware clears this bit. This bit cannot be reset if the HSE oscillator is used directly or indirectly as the system clock. 0: HSE off 1: HSE on
15:14	Reserved	-	-	Reserved
13:11	HSIDIV[2:0]	RW	0	HSI clock division factor. The software controls these bits to set the frequency division coefficient of the HSI, generating the HSI SYS clock 000: 1 001: 2 010: 4 011: 8 100: 16 101: 32 110: 64 111: 128
10	HSIRDY	R	0	HSI clock ready flag. Set by hardware to indicate HSI oscillator is stable. This bit is only valid if HSION = 1. 0: HSI oscillator not ready; 1: HSI oscillator ready; When HSION is cleared, HSIRDY will be pulled low after 3 HSI cycles.
9	MSIRDY	R	0	MSI clock ready flag. Set by hardware to indicate MSI oscillator is stable. This bit is only valid if MSION = 1. 0: MSI oscillator not ready; 1: MSI oscillator ready; When MSION is cleared, MSIRDY will be pulled low after 3 MSI cycles.
8	HSION	RW	1	HSI clock enable Set and cleared by software. When entering any stop mode or standby mode, hardware clears this bit to stop HSI. When HSI is directly or indirectly used as the system clock (also when exiting stop or standby mode, or when HSE is the system clock and fails), hardware sets this register. 0: HSI OFF 1: HSI ON

				Note: When HSION is cleared, HSI will generate 3 additional cycles, and HSIRDY will be cleared after 3 HSI cycles.
7	MSION	RW	0	MSI clock enable Set and cleared by software. When entering LPRUN mode and software selects MSI as the system clock, this register must be enabled. When entering a stop mode from LPRUN mode, and upon waking from this stop mode, if LPRUN mode is required again, hardware will enable MSI upon wake-up if MSI clock is selected by software configuration. 0: MSI OFF 1: MSI ON Note: When MSION is cleared, MSI will generate 3 additional cycles, and MSIRDY will be cleared after 3 HSI cycles.
6	HSIKERON	RW	0	HSI is always enabled for peripheral cores. When I2C is used as a wake-up source from stop mode, to ensure communication speed, this register can be set to keep HSI on during stop mode, avoiding communication speed reduction due to HSI startup time. This bit has no effect on the HSION value. 0: No effect on the HSI oscillator. 1: HSI oscillator is forcibly enabled even in stop mode.
5:3	Reserved	-	-	Reserved
2:0	MSIDIV[2:0]	RW	0	MSI clock division factor. The software controls these bits to set the divider coefficient of MSI, generating the MSISYS clock as the system clock for LPRUN mode. 000: 1 001: 2 010: 4 011: 8 100: 16 101: 32 110: 64 111: 128

6.3.2. RCC Internal Clock Source Calibration Register (RCC_ICSCR)

Offset address: 0x04

Reset value: 0x0000 1100

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved		-	Reserved
15:13	HSI_FS_OPCR	RW	3'b000	HSI frequency: 000: 8 MHz 001: 8 MHz 010: 16 MHz 011: Reserved 100: 24 MHz 101: 48 MHz 110: 64 MHz 111: 8 MHz After power-on, 8 MHz is selected by default. After the option byte loading is completed, the hardware updates the Trim value corresponding to 8 MHz by loading the Flash trimming page data. After a system reset, the hardware also switches to 8 MHz.
12:0	HSI_TRIMCR[12:0]	RW	13'h1100	HSI clock frequency calibration value.

				After power-on, the hardware uses the default calibration value of HSI 8 MHz. During power-on trimming loading, the factory information (stored at 0x1FFF 1F2C) is written into this register. The software reads the data stored at the corresponding address in the information area and writes it into this register to achieve calibration for the HSI at a specific output frequency. Saved in the following address in Flash: 64 MHz calibration value storage address: 0x1FFF 1E18 48 MHz calibration value storage address: 0x1FFF 1E14 24 MHz calibration value storage address: 0x1FFF 1E10 16 MHz calibration value storage address: 0x1FFF 1E08 8 MHz calibration value storage address: 0x1FFF 1E04 The register value can also be modified by writing the calibration value into the register, which is the center value. For every increase (decrease) of 1, the output frequency of the HSI increases (decreases) by about 0.1%. HSI_TRIM[12:9]: Coarse adjustment bits, HSI_TRIM[8:0]: Fine adjustment bits. During trimming, it needs to be controlled in two stages, such as keeping the coarse adjustment bits unchanged while changing the fine adjustment bits.

6.3.3. RCC clock configuration register (RCC_CFGR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCOPRE[3:0]				MCOSEL[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PPRE[2:0]			HPRE[3:0]				Res.	Res.	SWS[2:0]			SW[2:0]		
	RW	RW	RW	RW	RW	RW	RW			R	R	R	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	MCOPRE[2:0]	RW	0	MCO (microcontroller clock output) frequency division factor. The software controls these bits and sets the frequency division factor of the MCO output: 0000: 1 1000: 2 1001: 4 1010: 8 1011: 16 1100: 32 1101: 64 1110: 128 1111: 256 It is recommended to set these bits before the MCO output is enabled.
27:24	MCOSEL[3:0]	RW	0	MCO Selection 0000: No clock, MCO output disabled 0001: SYSCLK 0010: HSI10M 0011: HSI 0100: HSE 0101: PLL CLK 0110: LSI 0111: LSE 1000: HCLK 1001: PCLK 1010: MSI 1011: LCD_HSI10M Others: No clock Note: The output clock may be incomplete during the clock startup or switching phase.
23:15	Reserved	-	-	Reserved
14:12	PPRE[2:0]	RW	0	This bit is controlled by software. To generate the PCLK clock, it sets the division coefficients of HCLK as follows: 0xx: 1 100: 2 101: 4

				110: 8 111: 16
11:8	HPRE[3:0]	RW	0	AHB clock division factor. The software controls this bit. To generate the HCLK clock, it sets the division coefficients of SYSCLK as follows: 0000: 1 1000: 2 1001: 4 1010: 8 1011: 16 1100: 64 1101: 128 1110: 256 1111: 512 Others: Reserved To ensure the normal operation of the system, the appropriate frequency needs to be configured according to the VR power supply situation. Note: It is recommended to switch the frequency division coefficients gradually.
7:6	Reserved	-	-	Reserved
5:3	SWS[2:0]	R	0	System clock switching status bit These bits are controlled by hardware and indicate which clock source is currently being used as the system clock: 000: HSISYS 001: HSE 010: PLL CLK 011: LSI 100: LSE 101: MSI Others: Reserved
2:0	SW[2:0]	RW	0	System clock source select bit. These bits are controlled by software and hardware and are used to select the system clock: 000: HSISYS 001: HSE 010: PLL CLK 011: LSI 100: LSE 101: MSI Others: Reserved Scenarios where the hardware is configured as HSISYS include: 1) System exits from Stop mode 2) Software configuration 001 (HSE), HSE failure occurs (HSE is the system clock source, or HSE is used as PLL input with PLL as the system clock source)

6.3.4. RCC PLL Configuration Register (RCC_PLLCFGR)

Offset address: 0x0C

Reset value: 0x0000 0000

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

Note: This register must be configured before enabling the PLL (PLLON=1). If reconfiguration is needed, the PLL must first be disabled, reconfigured, and then re-enabled.

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6:2	PLLMUL[4:0]	RW	5'b0	PLL frequency doubling factor 00000: x2 00001: x3 00010: x4 00011: x5

				00100: x6 00101: x7 00110: x8 00111: x9 01000: x10 01001: x11 01010: x12 01011: x13 01100: x14 01101: x15 01110: x16 01111: x17 10000: x18 Others: Reserved (writable)
1	Reserved	-	-	Reserved
0	PLLSRC	RW	0	PLL clock source selection: 0: HSI 1: HSE

6.3.5. RCC external clock source configuration register (RCC_ECSCR)

Offset address: 0x10

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSE_FILT_ENB	HSE_RDYSEL[1:0]	HSE_DRIVER[1:0]		
											RW	RW	RW	RW	RW

Note: This register must be configured before enabling the HSE (HSEON=1). If reconfiguration is needed, the HSE must first be disabled, reconfigured, and then re-enabled.

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	Reserved
4	HSE_FILT_ENB	RW	0	Analog HSE filter control. 0: Enable filter 1: Disable filter
3:2	HSE_RDYSEL[1:0]	RW	2'b0	HSE crystal oscillator stabilization time selection. HSEBYP=0: 00: 4096 LSE clock cycles; 01: 2048 LSE clock cycles; 10: 8192 LSE clock cycles; 11: Direct output regardless of stabilization time; HSEBYP=1: 00: 2048 LSE clock cycles; 01: 1024 LSE clock cycles; 10: 4096 LSE clock cycles; 11: Direct output regardless of stabilization time;
1:0	HSE_DRV [1: 0]	RW	2'b0	HSE drive capability configuration. 00: gm 3.5 mA/V 01: gm 5.0 mA/V 10: gm 7.5 mA/V 11: gm 10 mA/V

6.3.6. RCC clock interrupt enable register (RCC_CIER)

Offset address: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL RDYIE	HSE RDYIE	HSI RDYIE	MSIRDYIE	LSE RDYIE	LSI RDYIE
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	PLLRDYIE	RW	0	PLL ready interrupt enable.

				0: Disabled 1: Enabled
4	HSERDYIE	RW	0	HSE clock ready interrupt enable. 0: Disabled 1: Enabled
3	HSIRDYIE	RW	0	HSI clock ready interrupt enable. 0: Disabled 1: Enabled
2	MSIRDYIE	RW	0	MSI clock ready interrupt enable. 0: Disabled 1: Enabled
1	LSERDYIE	RW	0	LSE clock ready interrupt enable. 0: Disabled 1: Enabled
0	LSIRDYIE	RW	0	LSI clock ready interrupt enable. 0: Disabled 1: Enabled

6.3.7. RCC clock interrupt flag register (RCC_CIFR)

Offset address: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	LSE CSSF	CSSF	Res.	Res.	PLL RDYF	HSE RDYF	HSI RDYF	MSI RDYF	LSE RDYF	LSI RDYF
						R	R			R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	LSECSSF	R	0	LSE Clock security system interrupt flag. This register is set when hardware detects an LSE oscillator failure. Software clears this bit by writing 1 to the LSECSSC register. 0: LSE clock failure interrupt not generated 1: LSE clock failure interrupt generated
8	CSSF	R	0	HSE clock safe system interrupt flag bit. This register is set when hardware detects an HSE oscillator failure. Software clears this bit by writing 1 to the CSSC register. 0: HSE clock failure interrupt not generated 1: HSE clock failure interrupt generated
7:6	Reserved	-	-	Reserved
5	PLLRDYF	R	0	PLL ready interrupt flag Set by hardware when PLL is locked and PLLRDYIE bit is set to 1. Cleared by software setting the PLLRDYC bit. 0: Clock ready interrupt due to PLL lock not generated 1: Clock ready interrupt due to PLL lock generated
4	HSERDYF	R	0	HSE ready interrupt flag Set by hardware when the HSE clock becomes stable and HSERDYIE is enabled. Cleared by software setting the HSERDYC bit. 0: No clock ready interrupt caused by HSE 1: Clock ready interrupt caused by HSE generated
3	HSIRDYF	R	0	HSI ready interrupt flag Set by hardware when the HSI clock becomes stable and HSIRDYIE is enabled. Cleared by software setting the HSIRDYC bit. 0: No clock ready interrupt caused by HSI 1: Clock ready interrupt caused by HSI generated
2	MSIRDYF	R	0	MSI ready interrupt flag This bit is set by hardware when MSI is stable and MSIRDYIE is enabled. The software sets the MSIRDYC bit and clears this bit. 0: No clock ready interrupt caused by MSI 1: Clock ready interrupt caused by MSI exists
1	LSERDYF	R	0	LSE ready interrupt flag bit

				Set by hardware when the LSE clock becomes stable and LSERDYIE is enabled. Cleared by software setting the LSERDYC bit. 0: No clock ready interrupt caused by LSE 1: Clock ready interrupt caused by LSE exists
0	LSIRDYF	R	0	LSI ready interrupt flag bit Set by hardware when the LSE clock becomes stable and LSERDYIE is enabled. Cleared by software setting the LSERDYC bit. 0: No clock ready interrupt caused by LSI 1: Clock ready interrupt caused by LSI exists

6.3.8. RCC clock interrupt clear register (RCC_CICR)

Offset address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	LSECSSC	CSSC	Res.	Res.	PLL RDYC	HSE RDYC	HSI RDYC	MSI RDYC	LSE RDYC	LSI RDYC
						W	W			W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	LSECSSC	W	0	LSE clock security system interrupt flag. 0: No effect; 1: Clear LSECSSF flag
8	CSSC	W	0	HSE clock security interrupt clear bit. 0: No effect 1: Clear CSSF flag bit
7:6	Reserved	-	-	Reserved
5	PLLRDYC	W	0	PLLS ready flag clear. 0: No effect 1: Clear the PLLRDYF bit
4	HSERDYC	W	0	HSE ready flag clear. 0: No effect 1: Clear HSERDYF bit
3	HSIRDYC	W	0	HSI ready flag clear. 0: No effect 1: Clear HSIRDYF bit
2	MSIRDYC	W	0	MSI ready flag clear. 0: No effect 1: Clear the MSIRDYF bit
1	LSERDYC	W	0	LSE ready flag clear. 0: No effect 1: Clear the LSERDYF bit
0	LSIRDYC	W	0	LSI ready flag clear. 0: No effect 1: Clear LSIRDYF bit

6.3.9. RCC I/O port reset register (RCC_IOPRSTR)

Offset address: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIOD RST	GPIOC RST	GPIOB RST	GPIOA RST
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved

3	GPIODRST	RW	0	I/O PortD reset. 0: No effect 1: PortD I/O reset Set and cleared by software.
2	GPIOCRST	RW	0	I/O PortC reset. 0: No effect 1: PortC I/O reset Set and cleared by software.
1	GPIOBRST	RW	0	I/O PortB reset. 0: No effect 1: I/O PortB reset. Set and cleared by software.
0	GPIOARST	RW	0	I/O PortA reset. 0: No effect 1: I/O PortA reset Set and cleared by software.

6.3.10. RCC AHB peripheral reset register (RCC_AHBRSTR)

Offset address: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRC RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA RST
			RW												RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	CRCRST	RW	0	CRC reset 0: No effect 1: CRC reset Set and cleared by software.
11:1	Reserved	-	-	Reserved
0	DMARST	RW	0	DMA reset. 0: No effect 1: DMA reset Set and cleared by software.

Note: The system reset or module reset described in each module is, unless otherwise specified, a system reset + module soft reset.

6.3.11. RCC APB peripheral reset register 1 (RCC_APBSTR1)

Offset address: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTI M1 RST	LPTI M2 RST	DACR ST	PW R RST	DBG RST	Re s.	Re s.	Re s.	OPAR ST	I2C 2 RST	I2C RS T	LPUART1 RST	UAR T2 RST	UAR T1 RST	USAR T2 RST	LPUART2 RST
RW	RW	RW	RW	RW					RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SPI2 RST	Res.	Re s.	WWDG RST	Re s	Re s.	Re s.	Res.	PW M RS T	TI M7 RS T	TIM6 RST	Res.	Res.	TIM3 RST	TIM2 RST
	RW			RW					RW	RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIM1RST	RW	0	LP Timer1 module reset. 0: No effect 1: The module is reset Set and cleared by software.
30	LPTIM2RST	RW	0	LP Timer2 module reset.

				0: No effect 1: The module is reset Set and cleared by software.
29	DACRST	RW	0	DAC reset 0: No effect 1: The module is reset Set and cleared by software.
28	PWRRST	RW	0	Power interface reset 0: No effect 1: The module is reset Set and cleared by software.
27	DBG RST	RW	0	DBG reset 0: No effect 1: The module is reset Set and cleared by software.
26:24	Reserved	-	-	Reserved
23	OPARST	RW	0	OPA reset 0: No effect 1: The module is reset Set and cleared by software.
22	I2C2RST	RW	0	I2C2 reset 0: No effect 1: The module is reset Set and cleared by software.
21	I2C1RST	RW	0	I2C1 reset. 0: No effect 1: The module is reset Set and cleared by software.
20	LPUART1RST	RW	0	LPUART1 reset 0: No effect 1: The module is reset Set and cleared by software.
19	UART2RST	RW	0	USART2 reset 0: No effect 1: The module is reset Set and cleared by software.
18	UART1RST	RW	0	UART1 reset 0: No effect 1: The module is reset Set and cleared by software.
17	USART2RST	RW	0	USART2 reset 0: No effect 1: The module is reset Set and cleared by software.
16	LPUART2RST	RW	0	LPUART2 reset 0: No effect 1: The module is reset Set and cleared by software.
15	Reserved	-	-	Reserved
14	SPI2RST	RW	0	SPI2 reset 0: No effect 1: The module is reset Set and cleared by software.
13:12	Reserved	-	-	Reserved
11	WWDGRST	RW	0	WWDG reset 0: No effect 1: The module is reset Set and cleared by software.
10:7	Reserved	-	-	Reserved
6	PWMRST	RW	0	PWM reset 0: No effect 1: The module is reset Set and cleared by software.
5	TIM7RST	RW	0	TIM7 reset 0: No effect 1: The module is reset Set and cleared by software.

4	TIM6RST	RW	0	TIM6 reset 0: No effect 1: The module is reset Set and cleared by software.
3:2	Reserved	-	-	Reserved
1	TIM3RST	RW	0	TIM3 reset 0: No effect 1: The module is reset Set and cleared by software.
0	TIM2RST	RW	0	TIM2 reset 0: No effect 1: The module is reset Set and cleared by software.

6.3.12. RCC APB peripheral reset register 2 (RCC_APBSTR2)

Offset address: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	VREFBUF RST	Res.	LCD RST	Res.	COMP2 RST	COMP1 RST	ADCRST	Res.	TIM17 RST	TIM16 RST	TIM15 RST
					RW		RW		RW	RW	RW		RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	USART1 RST	Res.	SPI1 RST	TIM1 RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYS CFG RST
	RW		RW	RW											RW

Bit	Name	R/W	Reset Value	Function
31:27	Reserved	-	-	Reserved
26	VREFBUF RST	RW	0	VREFBUF module reset. 0: No effect 1: The module is reset Set and cleared by software.
25	Reserved	-	-	Reserved
24	LCD RST	RW	0	LCD reset 0: No effect 1: The module is reset Set and cleared by software.
23	Reserved	-	-	Reserved
22	COMP2 RST	RW	0	COMP2 reset 0: No effect 1: The module is reset Set and cleared by software.
21	COMP1 RST	RW	0	COMP1 reset 0: No effect 1: The module is reset Set and cleared by software.
20	ADCRST	RW	0	ADC reset. 0: No effect 1: The module is reset Set and cleared by software.
19	Reserved	-	-	Reserved
18	TIM17 RST	RW	0	TIM17 reset 0: No effect 1: The module is reset Set and cleared by software.
17	TIM16 RST	RW	0	TIM16 reset 0: No effect 1: The module is reset Set and cleared by software.
16	TIM15 RST	RW	0	TIM15 reset 0: No effect 1: The module is reset Set and cleared by software.
15	Reserved	-	-	Reserved

14	USART1RST	RW	0	USART1 reset 0: No effect 1: The module is reset Set and cleared by software.
13	Reserved	-	-	Reserved
12	SPI1RST	RW	0	SPI1 reset 0: No effect 1: The module is reset Set and cleared by software.
11	TIM1RST	RW	0	TIM1 reset 0: No effect 1: The module is reset Set and cleared by software.
10:1	Reserved	-	-	Reserved
0	SYSCFG_RST	RWs	0	SYSCFG reset 0: No effect 1: The module is reset Set and cleared by software.

Note: The system reset or module reset described in each module is, unless otherwise specified, a system reset + module soft reset.

6.3.13. RCC I/O port clock enable register (RCC_IOPENR)

Offset Address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIOA EN	GPIOB EN	GPIOC EN	GPIOA EN
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3	GPIOAEN	RW	0	I/O PortD clock enable. 0: Clock disabled 1: clock enabled; Set and cleared by software.
2	GPIOBEN	RW	0	IO port C clock enable 0: Clock disabled 1: clock enabled; Set and cleared by software.
1	GPIOCEN	RW	0	IO port B clock enable 0: Clock disabled 1: clock enabled; Set and cleared by software.
0	GPIOAEN	RW	0	IO port A clock enable 0: Clock disabled 1: clock enabled; Set and cleared by software.

6.3.14. RCC AHB peripheral clock enable register (RCC_AHBENR)

Offset Address: 0x38

Reset Value: 0x0000 0300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRC EN	Res.	Res.	SRAMEN	FLASH EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA EN
			RW			RW	RW								RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved

12	CRCEN	RW	0	Enable CRC clock 0: Disabled 1: Enabled Set and cleared by software.
11:10	Reserved	-	-	Reserved
9	SRAMEN	RW	1	SRAM clock enable control in sleep mode 0: Module clock is disabled in sleep mode 1: Module clock is enabled in sleep mode Note: This bit only affects the module clock enable in sleep mode. In RUN mode, the module clock will not be disabled.
8	FLASHEN	RW	1	FLASH clock enable control in sleep mode 0: Module clock is disabled in sleep mode 1: Module clock is enabled in sleep mode Note: This bit only affects the module clock enable in sleep mode. In RUN mode, the module clock will not be disabled.
7:1	Reserved	-	-	Reserved
0	DMAEN	RW	0	DMA clock enable 0: Disabled 1: Enabled Set and cleared by software.

6.3.15. RCC APB Peripheral Clock Enable Register 1 (RCC_APBENR1)

Offset Address: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIM1EN	LPTIM2EN	DACEN	PWREN	DBGEN	Res.	Res.	Res.	OPAEN	I2C2EN	I2C1EN	LP UAR T1EN	UART2EN	UART1EN	USART2EN	LPUART2EN
RW	RW	RW	RW	RW				RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SPI2EN	Res.	Res.	WWDGEN	Res.	Res.	Res.	Res.	PWMEN	TIM7EN	TIM6EN	Res.	Res.	TIM3EN	TIM2EN
	RW			RW					RW	RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIM1EN	RW	0	LPTIM clock enable 0: Disabled 1: Enabled Set and cleared by software.
30	LPTIM2EN	RW	0	LP Timer2 module clock enable. 0: Disabled 1: Enabled Set and cleared by software.
29	DACEN	RW	0	DAC clock enable 0: Disabled 1: Enabled Set and cleared by software.
28	PWREN	RW	0	Power interface module clock enable. 0: Disabled 1: Enabled Set and cleared by software.
27	DBGEN	RW	0	DBG enable 0: Disabled 1: Enabled Set and cleared by software.
26:24	Reserved	-	-	Reserved
23	OPAEN	RW	0	OPA clock enable 0: Disabled 1: Enabled Set and cleared by software.
22	I2C2EN	RW	0	I2C2 enable 0: Disabled 1: Enabled Set and cleared by software.
21	I2C1EN	RW	0	I2C1 enable

				0: Disabled 1: Enabled Set and cleared by software.
20	LPUART1EN	RW	0	LP UART1 module clock enable. 0: Disabled 1: Enabled Set and cleared by software.
19	UART2EN	RW	0	UART2 enable 0: Disabled 1: Enabled Set and cleared by software.
18	UART1EN	RW	0	UART1 enable 0: Disabled 1: Enabled Set and cleared by software.
17	USART2EN	RW	0	USART2 enable 0: Disabled 1: Enabled Set and cleared by software.
16	LPUART2EN	RW	0	LP UART2 module clock enable. 0: Disabled 1: Enabled Set and cleared by software.
15	Reserved	-	-	Reserved
14	SPI2EN	RW	0	SPI2 enable 0: Disabled 1: Enabled Set and cleared by software.
13:12	Reserved	-	-	Reserved
11	WWDGEN	RW	0	WWDG clock enable 0: Disabled 1: Enabled Set and cleared by software.
10:7	Reserved	-	-	Reserved
6	PWMEN	RW	0	PWM clock enable 0: Disabled 1: Enabled Set and cleared by software.
5	TIM7EN	RW	0	TIM7 enable 0: Disabled 1: Enabled Set and cleared by software.
4	TIM6EN	RW	0	TIM6 enable 0: Disabled 1: Enabled Set and cleared by software.
3:2	Reserved	-	-	Reserved
1	TIM3EN	RW	0	TIM3 enable 0: Disabled 1: Enabled Set and cleared by software.
0	TIM2EN	RW	0	TIM2 enable 0: Disabled 1: Enabled Set and cleared by software.

6.3.16. RCC APB peripheral clock enable register 2 (RCC_APBENR2)

Offset address: 0x40

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	VERBUFE N	Res.	LCDE N	Res.	COMP 2 EN	COMP 1 EN	AD C EN	Res.	TIM1 7 EN	TIM1 6 EN	TIM1 5 EN
					RW		RW		RW	RW	RW		RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	USART 1 EN	Res.	SPI 1 EN	TIM 1 EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYS CFG EN
	RW		RW	RW											RW

Bit	Name	R/W	Reset Value	Function
31:27	Reserved	-	-	Reserved
26	VREFBUFEN	RW	0	VREFBUF module clock enable. 0: Disabled 1: Enabled Set and cleared by software.
25	Reserved	-	-	Reserved
24	LCDEN	RW	0	LCD clock enable 0: Disabled 1: Enabled Set and cleared by software.
23	Reserved	-	-	Reserved
22	COMP2EN	RW	0	COMP2 enable 0: Disabled 1: Enabled Set and cleared by software.
21	COMP1EN	RW	0	COMP1 enable 0: Disabled 1: Enabled Set and cleared by software.
20	ADCEN	RW	0	ADC enable 0: Disabled 1: Enabled Set and cleared by software.
19	Reserved	-	-	Reserved
18	TIM17EN	RW	0	TIM17 enable 0: Disabled 1: Enabled Set and cleared by software.
17	TIM16EN	RW	0	TIM16 enable 0: Disabled 1: Enabled Set and cleared by software.
16	TIM15EN	RW	0	TIM15 enable 0: Disabled 1: Enabled Set and cleared by software.
15	Reserved	-	-	Reserved
14	USART1EN	RW	0	USART1 enable 0: Disabled 1: Enabled Set and cleared by software.
13	Reserved	-	-	Reserved
12	SPI1EN	RW	0	SPI1 enable 0: Disabled 1: Enabled Set and cleared by software.
11	TIM1EN	RW	0	TIM1 enable 0: Disabled 1: Enabled Set and cleared by software.
10:1	Reserved	-	-	Reserved
0	SYSCFGEN	RW	1	SYSCFG clock enable 0: Disabled 1: Enabled Set and cleared by software.

6.3.17. RCC peripheral clock configuration register (RCC_CCIPR)

Offset address: 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCSEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPTIM2SEL [1:0]		LPTIM1SEL [1:0]	
RW												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Res.	LPUART2SEL	LPUART1SEL	Res.	COMP2SEL	COMP1SEL	PVDSEL	Res.	Res.	Res.	Res.	Res.	Res.	TIMCLKCTRL
	RW	RW		RWs	RW	RW							RW

Bit	Name	R/W	Reset Value	Function
31:30	ADCSEL[1:0]	RW	2'b00	ADC clock source selection. 00: SYSCLK 01: PLL 10: HSI 11: Reserved Note: Before switching the clock source, disable the ADC module clock enable first, then re-enable it after switching is completed.
29:20	Reserved	-	-	Reserved
19:18	LPTIM2SEL [1:0]	RW	2'b00	LPTIM2 clock source selection 00: PCLK 01: LSI 10: Reserved 11: LSE Note: Before switching the clock source, disable the LPTIM2 module clock enable first, then re-enable it after switching is completed.
17:16	LPTIM1SEL [1:0]	RW	2'b00	LPTIM1 clock source selection 00: PCLK 01: LSI 10: Reserved 11: LSE Note: Before switching the clock source, disable the LPTIM1 module clock enable first, then re-enable it after switching is completed.
15	Reserved	-	-	Reserved
14:13	LPUART2SEL [1:0]	RW	0	LPUART2 clock source selection. 00: PCLK 01: SYSCLK 10: LSI 11: LSE Note: Before switching the clock source, disable the LPUART2 module clock enable first, then re-enable it after switching is completed.
12:11	LPUART1SEL [1:0]	RW	0	LPUART1 Clock source selection. 00: PCLK 01: SYSCLK 10: LSI 11: LSE Note: Before switching the clock source, disable the LPUART1 module clock enable first, then re-enable it after switching is completed.
10	Reserved	-	-	Reserved
9	COMP2SEL	RW	0	COMP2 clock source selection 0: PCLK 1: LSC (clock selected by RCC_BDCR.LSCOSEL) Note: 1. Configure the LSC clock selection before enabling COMP2_FR.FLTEN. 2. Before switching the clock source, disable the COMP2 module clock enable first, then re-enable it after switching is completed.
8	COMP1SEL	RW	0	COMP1 clock source selection 0: PCLK 1: LSC (clock selected by RCC_BDCR.LSCOSEL) Note: 1. Configure this register to select the clock before enabling COMP2_FR2.FLTEN. 2. Before switching the clock source, disable the clock enable first, then re-enable it after the switch is completed.
7	PVDSEL	RW	0	PVD detection clock source selection. 0: PCLK 1: LSC (clock selected by RCC_BDCR.LSCOSEL)

				Note: Before switching the clock source, disable the COMP1 module clock enable first, then re-enable it after the switch is completed.
6:1	Reserved	-	-	Reserved
0	TIMCLKCTRL	RW	0	TIMER PCLK frequency control. 0: TIMER PCLK is system PCLK * 2, but the frequency will not exceed HCLK 1: TIMER PCLK is system PCLK * 1

6.3.18. RCC RTC Field Control Register (RCC_BDCR)

Offset address: 0x5C

Reset value: 0x0200 2800

1. This register can only be written when PWR_CR1.DBP is set to 1.
2. BDRST, LSE_STARTUP, LSE_DRIVER, LSI_TRIMCR can only be reset by BPOR. Other register bits are reset by BPOR and RTC soft reset.
3. Clock configuration must be performed when the clock is disabled, i.e., LSE_DRIVER [1:0] and LSE_STARTUP[1:0] must be configured when LSE is disabled. LSI_TRIMCR[8:0] must be configured when LSI is disabled.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	IWDGSEL	LSE_STARTUP [1:0]		LSIRDY	LSION	LSI_TRIMCR[8:0]								BDRST	
	RW	RW	RW	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RT C EN	LSCSEL	LSEEN[3:0]				RTCSEL[1:0]		Res.	LSECSSD	LSECSSON	LSE_DRIVER [1:0]		LSEBY P	LSERDY	Res.
RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	R	

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30	IWDGSEL	RW	0	IWDG internal clock source selection. 0: LSI 1: LSE
29:28	LSE_STARTUP [1:0]	RW	0	LSE crystal oscillator stabilization time selection. LSEBYP=0: 00: 4096 LSE clock cycles 01: 2048 LSE clock cycles 10: 8192 LSE clock cycles 11: Direct output regardless of stabilization time LSEBYP=1: 00: 2048 LSE clock cycles 01: 1024 LSE clock cycles 10: 4096 LSE clock cycles 11: Direct output regardless of stabilization time
27	LSIRDY	R	0	LSI clock ready flag. 0: LSI not ready 1: LSI ready
26	LSION	RW	0	LSI clock enable. 0: Disabled 1: Enabled Set and cleared by software. This bit is set by hardware when IWDG is enabled (via option bytes) and LSECSSON is enabled by software.
25:17	LSI_TRIMCR[8:0]	RW	9'h100	LSI clock frequency calibration. After power-on, the chip hardware writes the factory information (stored at 0x1FFF 1F24) into this register, enabling LSI to output an accurate 32.768 kHz frequency. By modifying the register value, the software increases (decreases) the LSI output frequency by approximately 0.2% for each increment (decrement) of 1.
16	BDRST	RW	0	RTC domain software reset.

				0: No effect 1: Reset
15	RTCCEN	RW	0	RTC clock enabled. 0: Disabled 1: Enabled
14	LSCSEL	RW	0	Low speed clock output selection 0: LSI 1: LSE
13:10	LSEEN[3:0]	RW	4'hA	Enable register, LSE enabled by default upon power-up 1010: Disable LSE 0101: Enable LSE The software must write 1010 to disable LSE and must write 0101 to enable LSE.
9:8	RTCSEL[1:0]	RW	0	RTC clock source selection. 00: No clock 01: LSE 10: LSI 11: HSE clock divided by 32 Once the RTC clock source is selected, it cannot be changed unless: ■ RTC domain is reset to 00 ■ Selected as LSE (LSECSSD=1) but no LSE present ■ BDRST soft reset resets this register to 00
7	Reserved	-	-	Reserved
6	LSECSSD	R	0	LSE CSS (Clock Security System) fault detection. This bit is set by hardware to indicate that the CSS has detected a fault in the 32.768 kHz crystal oscillator (LSE). 0: No LSE fault detected 1: LSE fault detected
5	LSECSSON	RW	0	CSS on LSE enable 0: Disabled 1: Enabled LSECSSON must be enabled after the LSE oscillator is enabled (LSEON=1) and ready (LSERDY=1). Once enabled this bit cannot be disabled, except after an LSE failure detection (LSECSSD=1).
4:3	LSE_DRIVER[1:0]	RW	0	LSE drive capability settings 00: Icc 200nA, gm 2.5μA/V 01: Icc 300nA, gm 3.5μA/V 10: Icc 600nA, gm 7.5μA/V 11: Icc 1000nA, gm 10μA/V
2	LSEBYP	RW	0	LSE oscillator bypass 0: Not bypassed, low-speed external clock selects crystal oscillator 1: Bypassed, low-speed external clock selects external interface input clock Note: This bit can only be written when the external 32.768 kHz oscillator is disabled (LSEON=0 and LSERDY=0).
1	LSERDY	R	0	LSE oscillator ready flag. Set and cleared by hardware to indicate the external low-speed oscillator is ready (stable). 0: Not ready 1: Ready
0	Reserved	-	-	Reserved

6.3.19. RCC Control/Status Register (RCC_CSR)

Offset address: 0x60

Reset value: 0x0000 0000

Reset by POR (flag bit), reset by system reset excluding NRST (NRST_FLTIDS)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPW R RSTF	WWD G RSTF	IWD G RSTF	SFT RST F	PWR RST F	PIN RST F	OBL RST F	Res.	RMV F	SPERST F	Res. .	Res. .	Res. .	Res. .	Res. .	Res. .
R	R	R	R	R	R	R		RW	R						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	NRST_ FLTID S	Res.	Res.	Res. .	Res. .	Res. .	Res. .	Res. .	Res. .

7.0	Reserved		

7. General-purpose I/Os (GPIO)

7.1. Introduction to GPIO

Each general-purpose I/O port has:

Four 32-bit configuration registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, and GPIOx_PUPDR)

Two 32-bit data registers (GPIOx_IDR and GPIOx_ODR)

A 32-bit set/reset register (GPIOx_BSRR)

A 32-bit locking register (GPIOx_LCKR)

2 multiplexing function selection registers (GPIOx_AFRH and GPIOx_AFRL).

7.2. GPIO main features

- Registers support Fast IO Port and AHB bus read/write
- Output state: Push-pull or open-drain output + pull-up/pull-down
- Output data from output data register (GPIOx_ODR) or peripheral (alternate function output)
- Speed selection for each I/O
- Input status: floating, pull-up/pull-down, analog
- Input data to input data register (GPIOx_IDR) or peripheral (alternate function input)
- Set/reset register (GPIOx_BSRR), allowing bitwise write to GPIOx_ODR
- Locking mechanism (GPIOx_LCKR) provided to freeze the I/O port configurations
- Analog function
- Alternate function selection registers (at most 16 AFs possible per I/O)
- Fast toggle
- Highly flexible I/O multiplexing function, allowing the I/O port to function as GPIO or as various peripheral interface functions

7.3. GPIO functional description

Each port bit of the general-purpose I/O (GPIO) ports can be individually configured by software in several modes:

- Input floating
- Input pull-up
- Input-pull-down
- Analog
- Output open-drain with pull-up or pull-down capability
- Output push-pull with pull-up or pull-down capability
- Multiplexed function push-pull with pull-up or pull-down
- Multiplexed function open-drain with pull-up or pull-down

Each I/O port can be freely programmed, but I/O port registers must be accessed as 32-bit words. The purpose of the GPIOx_BSRR register is to allow atomic read/modify accesses to any of the GPIOx_ODR registers. This ensures no hazards occur when interrupt requests arise between read and modify accesses.

The following figure shows the basic structure of a standard I/O port.

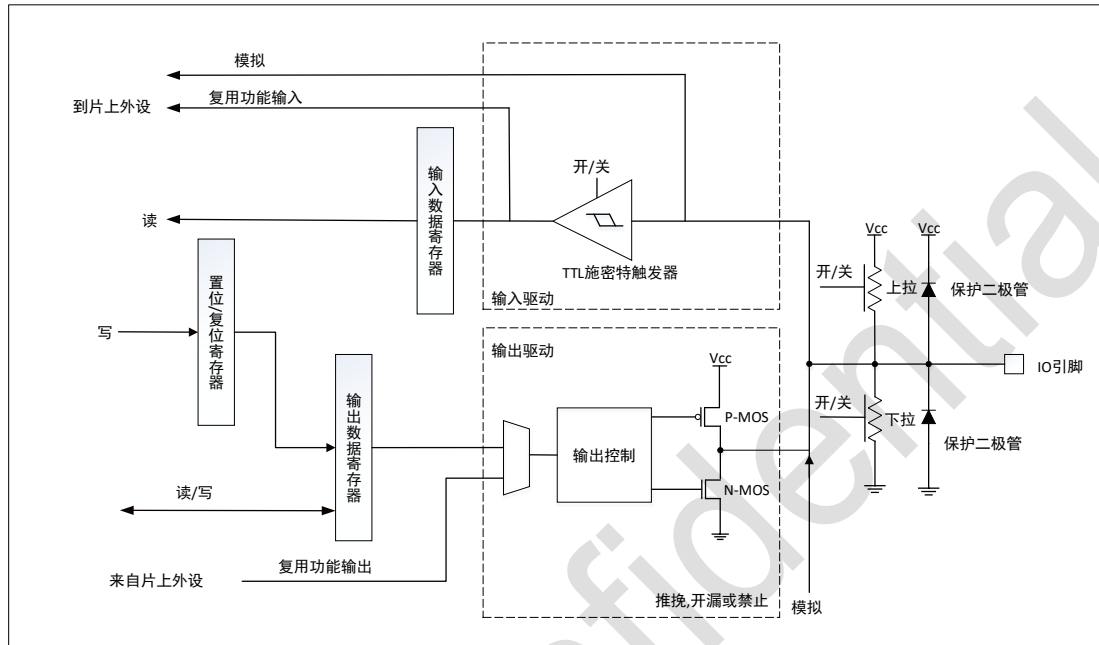


Figure 7-1 Basic structure of an I/O port bit

Table 7-1 Port bit configuration table

MODE(i) [1:0]	OTYPE	OSPEED(i) [1:0]		PUPD(i) [1:0]		IO configuration	
01	0	SPEED[1:0]		0	0	the GP output	PP
	0			0	1	the GP output	PP+PU
	0			1	0	the GP output	PP+PD
	0			1	1	Reserved	
	1			0	0	the GP output	OD
	1			0	1	the GP output	OD+PU
	1			1	0	the GP output	OD+PD
	1			1	1	Reserved (GP output OD)	
10	0	SPEED[1:0]		0	0	AF	PP
	0			0	1	AF	PP+PU
	0			1	0	AF	PP+PD
	0			1	1	Reserved	
	1			0	0	AF	OD
	1			0	1	AF	OD+PU
	1			1	0	AF	OD+PD
	1			1	1	Reserved	
00	x	x	x	0	0	I	Floating
	x	x	x	0	1	I	PU
	x	x	x	1	0	I	PD
	x	x	x	1	1	Reserved (input floating)	
11	x	x	x	0	0	I/O	Analog
	x	x	x	0	1	Reserved (PUPD cannot be configured as pull-up/down)	
	x	x	x	1	0		
	x	x	x	1	1		

GP = General Purpose, PP = Push-Pull, PU = Pull-Up, PD = Pull-Down, OD = Open-Drain, AF = Alternate Function.

7.3.1. General-purpose I/Os (GPIO)

During and just after reset, the alternate functions are not active and most of the I/O ports are configured in analog mode.

Debug pins are defaulted to alternate function pull-up or pull-down mode. The chip contains two sets of debug interface mappings, with configuration options as shown in the table below:

SWD_MODE[1:0]	PA4	PA5	PB6	PB5
00(default)	SWDIO	SWCLK	GPIO	GPIO
01	GPIO	GPIO	SWDIO	SWCLK
10	GPIO	SWCLK	SWDIO	GPIO
11	SWDIO	GPIO	GPIO	SWCLK

When PA5 and PB5 are configured as SWCLK via SWD_OPTION, they are set to pull-down mode;

When PA4 and PB6 are configured as SWDIO via SWD_OPTION, they are set to pull-up mode;

The Boot pin (PD8) is defaulted to input mode with pull-down mode;

When the pin is configured as output, the value written to the output data register (GPIOx_ODR) is output on the I/O pin. Push-pull or open-drain output mode may be used (low level is output, high level is high-impedance state).

The input data register (GPIOx_IDR) captures the data present on the I/O pin at every AHB clock cycle.

All GPIO pins have weak internal pull-up and pull-down resistors, which can be activated or not depending on the value in the GPIOx_PUPDR register.

7.3.2. I/O pin alternate function

The device I/O pins are connected to on-board peripherals/modules through a multiplexer that allows only one peripheral alternate function (AF) connected to an I/O pin at a time. This avoids conflicts between available peripherals on the same I/O port.

Each I/O port has a multiplexer with up to 16 alternate function inputs (AF0 to AF15), configurable via registers GPIOx_AFRL (alternate functions 0 to 7) and GPIOx_AFRH (alternate functions 8 to 15).

- After reset, the multiplexer defaults to AF0. The I/Os are configured in alternate function mode through GPIOx_MODER register
- The alternate functions for each pin are described in the corresponding datasheet.

In addition to this flexible I/O multiplexing architecture, each peripheral has alternate functions mapped onto different I/O pins to optimize the number of peripherals available in smaller packages.

To use an I/O in a given configuration, the user has to proceed as follows:

- Debug function: after each device reset these pins are assigned as alternate function pins immediately usable by the debugger host
- GPIO: configure the desired I/O as output, input or analog in the GPIOx_MODER register.
- Peripheral alternate function:
 - The I/O corresponding to the register GPIOx_AFRL or GPIOx_AFRH configuration is multiplexing function x (x = 0... 15)

- Select the type, pull-up/pull-down and output speed via the GPIOx_OTYPER, GPIOx_PUPDR and GPIOx_OSPEEDER registers, respectively.
- Configure the desired I/O as an alternate function in the GPIOx_MODER register.
- Additional functions:
 - Regardless of the I/O port configuration mode, ADC, LCD, and TK functions are enabled in their respective module registers. When using I/O ports for ADC, LCD, or TK, it is recommended to configure the port in analog mode via the GPIOx_MODER register.
 - For OPA and COMP functions, in addition to enabling them in the OPA and COMP module registers, the ANA2EN register in the SYSCFG module must also be configured. When using I/O ports for OPA and COMP functions, it is recommended to configure the port in analog mode via the GPIOx_MODER register.
 - For additional functions such as RTC, TAMP, WKUPx, and crystal oscillators, configure their respective functions in the corresponding RTC, TAMP, PWR, and RCC module registers. These functions have priority over the configuration in the standard GPIO registers.

7.3.3. I/O port control registers

Each GPIO port has four 32-bit memory-mapped control registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, and GPIOx_PUPDR), which can configure up to 16 I/O ports. The GPIOx_MODER register is used to select the I/O mode (input, output, AF, analog). The GPIOx_OTYPER and GPIOx_OSPEEDR registers are used to select the output type (push- pull or open-drain) and speed. The GPIOx_PUPDR register is used to select pull-up/pull-down.

7.3.4. I/O data registers

Each GPIO has two 16-bit memory-mapped data registers: input and output data registers (GPIOx_IDR and GPIOx_ODR). GPIOx_ODR stores the data to be output, it is read/write accessible. The data input through the I/O are stored into the input data register (GPIOx_IDR), a read-only register.

7.3.5. I/O data bit manipulation

The bit set reset register (GPIOx_BSRR) is a 32-bit register which allows the application to set and reset each individual bit in the output data register (GPIOx_ODR). The bit set reset register has twice the size of GPIOx_ODR.

To each bit in GPIOx_ODR, correspond two control bits in GPIOx_BSRR: BS(i) and BR(i). When written to 1, bit BS(i) sets the corresponding ODR(i) bit. When written to 1, bit BR(i) resets the ODR(i) corresponding bit.

Writing any bit to 0 in GPIOx_BSRR does not have any effect on the corresponding bit in GPIOx_ODR. If there is an attempt to both set and reset a bit in GPIOx_BSRR, the set action takes priority.

Using the GPIOx_BSRR register to change the values of individual bits in GPIOx_ODR is a “one-shot” effect that does not lock the GPIOx_ODR bits. The GPIOx_ODR bits can always be accessed directly. The GPIOx_BSRR register provides a way of performing atomic bitwise handling.

When software manipulates bits in GPIOx_ODR, it is not necessary to disable interrupts: one or more bits may be modified during a single AHB write access.

7.3.6. GPIO locking mechanism

It is possible to freeze the GPIO control registers by applying a specific write sequence to the GPIOx_LCKR register. The frozen registers are GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR, GPIOx_AFRL and GPIOx_AFRH.

To write the GPIOx_LCKR register, a specific write / read sequence has to be applied. When the right LOCK sequence is applied to bit 16 in this register, the value of LCKR [15:0] is used to lock the configuration of the I/Os (during the write sequence the LCKR [15:0] value must be the same). When the LOCK sequence has been applied to a port bit, the value of the port bit can no longer be modified until the next MCU reset or peripheral reset. Each bit in GPIOx_LCKR freezes the corresponding bit in the control registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR, GPIOx_AFRL, and GPIOx_AFRH).

The LOCK sequence can only be performed with a word (32-bit long) access to the GPIOx_LCKR register, because setting bit 16 of GPIOx_LCKR also sets bits [15:0].

7.3.7. I/O alternate function input/output

Two registers are provided to select one of the alternate function inputs/outputs available for each I/O. With these registers, the user can connect an alternate function to some other pin as required by the application.

This means that a number of possible peripheral functions are multiplexed on each GPIO using the GPIOx_AFRL and GPIOx_AFRH alternate function registers. The application can thus select any one of the possible functions for each I/O. The AF selection signal being common to the alternate function input and alternate function output, a single channel is selected for the alternate function input/output of a given I/O.

7.3.8. External interrupt line/wake-up line

All ports have external interrupt capability. To use external interrupt lines, the port must not be configured in analog mode or as a crystal oscillator pin, and trigger input must be enabled.

7.3.9. Input configuration

When the I/O port is programmed as input:

- The output buffer is disabled
- The Schmitt trigger input is activated
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- The data present on the I/O pin are sampled into the input data register every AHB clock cycle
- A read access to the input data register provides the I/O state

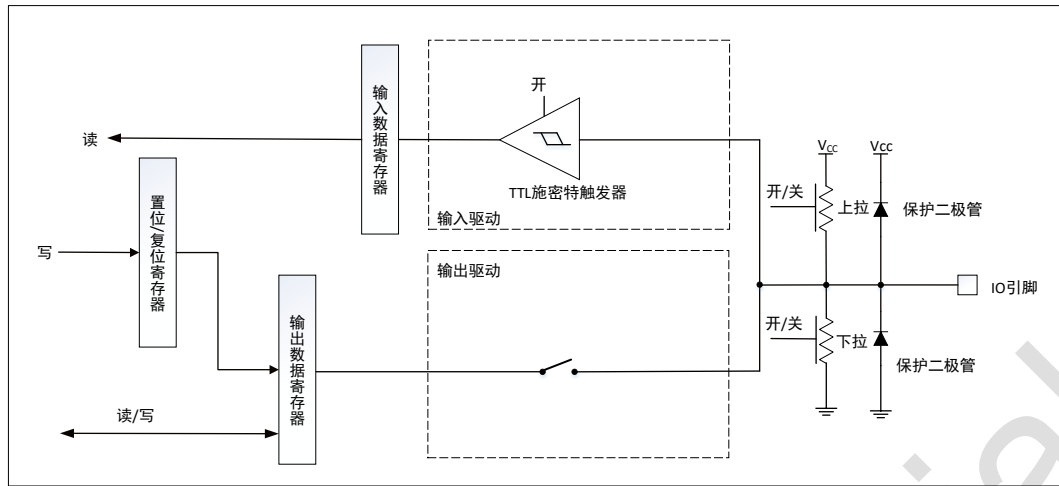


Figure 7-2 Input floating/pull-up/pull-down configuration

7.3.10. Output configuration

When the I/O port is programmed as output:

- The output buffer is activated.
 - Open drain mode: A “0” in the Output register activates the N-MOS whereas a “1” in the output register leaves the port in Hi-Z (the P-MOS is never activated).
 - Push-pull mode: A “0” in the Output register activates the N-MOS whereas a “1” in the Output register activates the P-MOS.
- The Schmitt trigger input is activated
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- The data present on the I/O pin are sampled into the input data register every AHB clock cycle
- A read access to the input data register provides the I/O state
- A read access to the output data register retrieves the last written value.

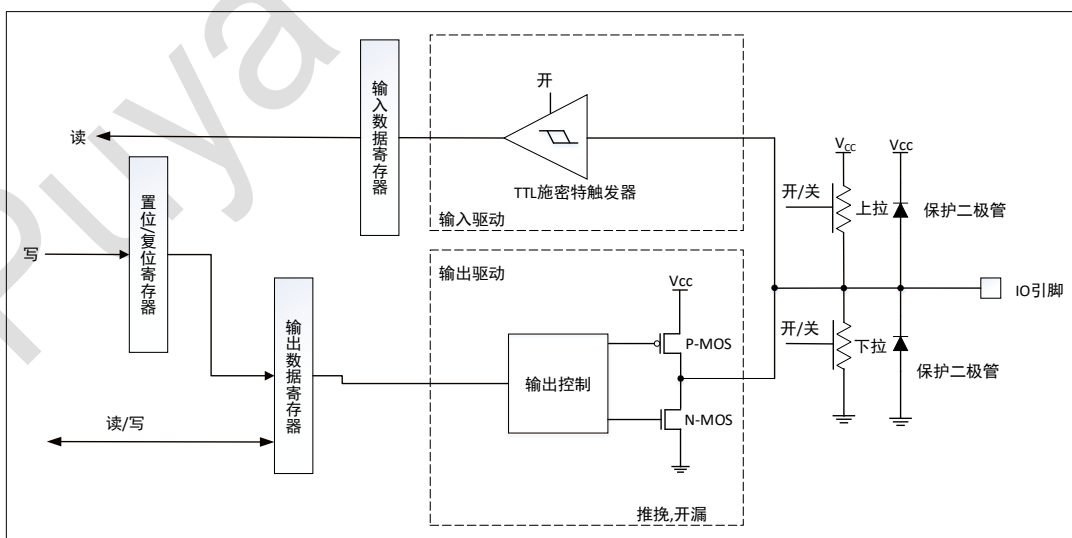


Figure 7-3 Output configuration

7.3.11. Alternate function configuration

When the I/O port is programmed as alternate function:

- The output buffer can be configured in open-drain or push-pull mode
- The output buffer is driven by the signals coming from the internal peripheral (alternate function output)
- The Schmitt trigger input is activated
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- The data present on the I/O pin are sampled into the input data register every AHB clock cycle
- A read access to the input data register provides the I/O state

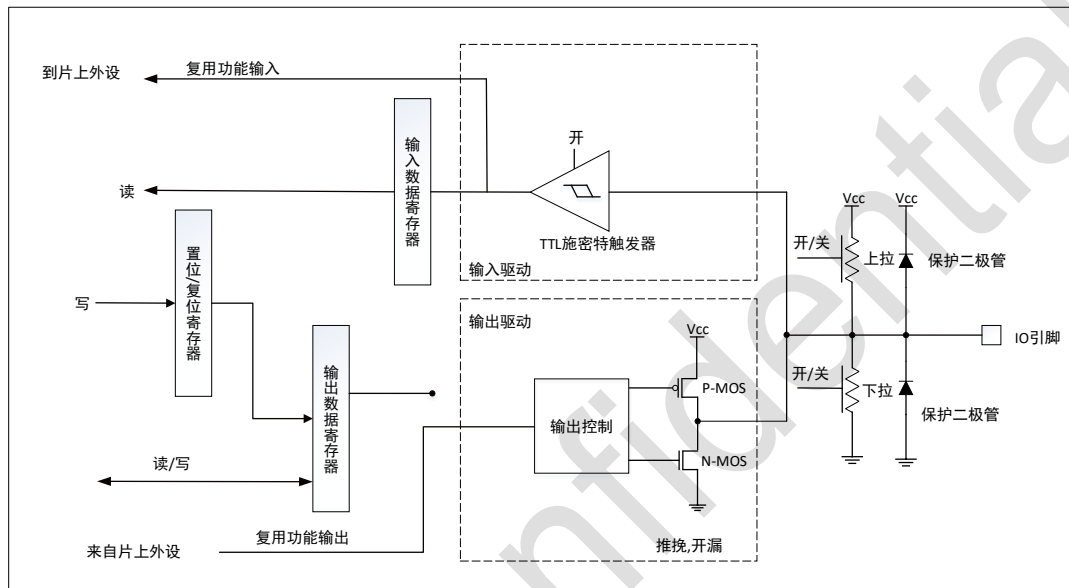


Figure 7-4 Alternate function configuration

7.3.12. Analog configuration

When the I/O port is programmed as analog configuration:

- The output buffer is disabled;
- Disabling the Schmitt trigger input reduces the power consumption of each analog input on the I/O pin to zero. The output of the Schmitt trigger is forced to a constant value (0);
- Weak pull-up and pull-down resistors are disabled (software setting is required);
- Read access to the input data register gets the value "0".

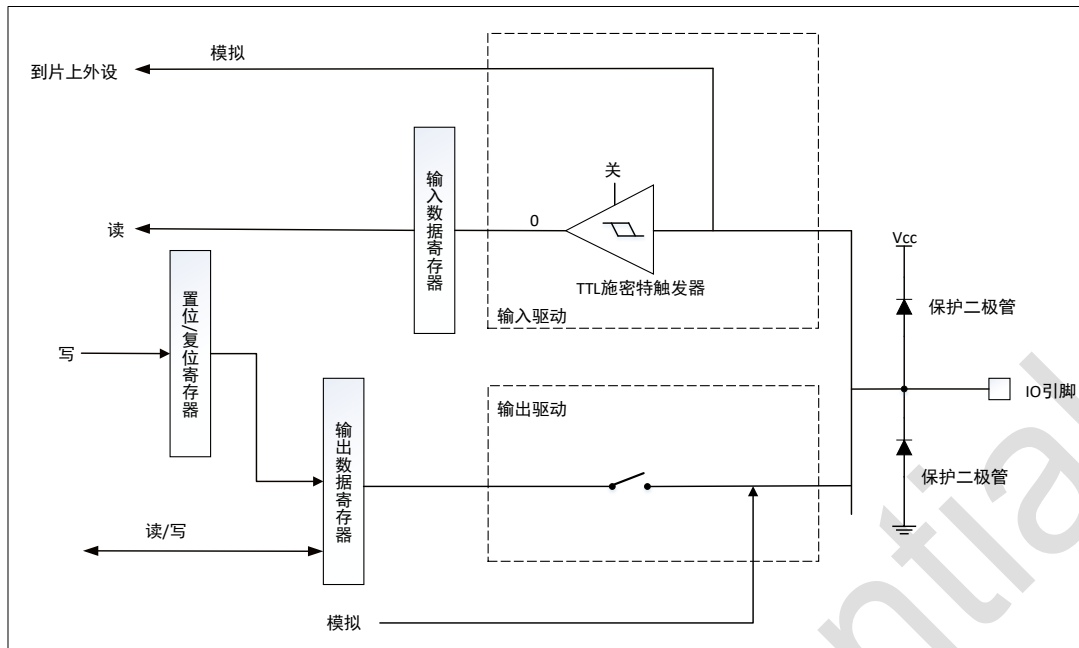


Figure 7-5 High impedance-analog configuration

7.3.13. Using HSE/LSE pins as GPIO

When the HSE or LSE function is disabled (default after reset), the corresponding pins can be used as normal GPIO.

When the HSE or LSE function is enabled (HSEON set in the RCC_CR register, LSEEN configured in the RCC_BDCR register), the corresponding port must be configured as an analog port by software.

When the oscillator is configured in a user external clock mode, only the pin OSC_IN or OSC32_IN is reserved for clock input and the OSC_OUT or OSC32_OUT pin can still be used as normal GPIO.

7.3.14. Backup power domain GPIO pins

When the core power domain is powered off (device enters standby mode), the PA6/PA7/PA8 GPIO functionality will be lost. In this case, if the GPIO is not configured for RTC-related alternate functions, these pins will be in analog input mode.

7.4. GPIO registers

7.4.1. GPIO port mode register (GPIOx_MODER) (x=A,B,C,D)

Offset address: 0x00

Reset value:

- GPIOA :
 - SWD_MODE=00: 0xFFFF FAFF
 - SWD_MODE=01: 0xFFFF FFFF
 - SWD_MODE=10: 0xFFFF FBFF
 - SWD_MODE=11: 0xFFFF FEFF
- GPIOB:
 - SWD_MODE=00: 0xFFFF FFFF
 - SWD_MODE=01: 0xFFFF EBFF

--SWD_MODE=10: 0xFFFF EFFF

--SWD_MODE=11: 0xFFFF FBFF

■ 0xFFFF FFFF for GPIOC

■ 0x0FFC FFFF For GPIOD

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE15 [1:0]	MODE14 [1:0]	MODE13 [1:0]	MODE12 [1:0]	MODE11 [1:0]	MODE10 [1:0]	MODE9 [1:0]	MODE8 [1:0]	MODE7 [1:0]	MODE6 [1:0]	MODE5 [1:0]	MODE4 [1:0]	MODE3 [1:0]	MODE2 [1:0]	MODE1 [1:0]	MODE0 [1:0]
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE7 [1:0]	MODE6 [1:0]	MODE5 [1:0]	MODE4 [1:0]	MODE3 [1:0]	MODE2 [1:0]	MODE1 [1:0]	MODE0 [1:0]								
RW	RW	RW	RW	RW	RW	RW	RW								

Bit	Name	R/W	Reset Value	Function
31:0	MODEy[1:0]	RW		y = 15..0 These bits are written by software to configure the I/O mode. 00: Input mode 01: General purpose output mode 10: Alternate function mode 11: Analog mode (reset state for most IOS)

7.4.2. GPIO port output type register (GPIOx_OTYPER) (x = A,B,C,D)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	OTy[1:0]	RW	0	These bits are written by software to configure the I/O output type. 0: Output push-pull (reset state) 1: Output open-drain

7.4.3. GPIO port output drive register (GPIOx_OSPEEDR) (x = A,B,C, D)

Offset address: 0x08

Reset value:

■ GPIOA :

--SWD_MODE=00: 0x0000 0100

--SWD_MODE=01: 0x0000 0000

-- SWD_MODE=10: 0x0000 0000

-- SWD_MODE=11: 0x0000 0100

■ GPIOB :

--SWD_MODE=00: 0x0000 1000

--SWD_MODE=01: 0x0000 0000

-- SWD_MODE=10: 0x0000 1000

-- SWD_MODE=11: 0x0000 0000

■ Other ports: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSPEED15 [1:0]	OSPEED14 [1:0]	OSPEED13 [1:0]	OSPEED12 [1:0]	OSPEED11 [1:0]	OSPEED10 [1:0]	OSPEED9 [1:0]	OSPEED8 [1:0]								

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSPEED7 [1:0]		OSPEED6 [1:0]		OSPEED5 [1:0]		OSPEED4 [1:0]		OSPEED3 [1:0]		OSPEED2 [1:0]		OSPEED1 [1:0]		OSPEED0 [1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	OSPEEDy[1:0]	RW		Y = 15.. 0 These bits are written by software to configure the I/O output speed. 00: Very low 01: Low speed 10: high speed 11: Very high speed

7.4.4. GPIO port pull-up/pull-down register (GPIOx_PUPDR) (x = A,B,C, D)

Offset address: 0x0C

Reset value:

■ GPIOA:

--SWD_MODE=00: 0x0000 0900
--SWD_MODE=01: 0x0000 0000
--SWD_MODE=10: 0x0000 0800
--SWD_MODE=11: 0x0000 0100

■ GPIOB:

--SWD_MODE=00: 0x0000 0000
--SWD_MODE=01: 0x0000 1800
--SWD_MODE=10: 0x0000 1000
--SWD_MODE=11: 0x0000 0800

■ 0x0000 0000(Port C)

■ 0x0002 0000(Port D)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUPD15 [1:0]		PUPD14 [1:0]		PUPD13 [1:0]		PUPD12 [1:0]		PUPD11 [1:0]		PUPD10 [1:0]		PUPD9 [1:0]		PUPD8 [1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUPD7 [1:0]		PUPD6 [1:0]		PUPD5 [1:0]		PUPD4 [1:0]		PUPD3 [1:0]		PUPD2 [1:0]		PUPD1 [1:0]		PUPD0 [1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	PUPDy [1: 0]	RW		Y = 15.. 0 These bits are written by software to configure the I/O pull-up or pull-down 00: No pull-up, pull-down 01: Pull-up 10: Pull-down 11: Reserved (no pull-up/pull-down)

7.4.5. GPIO port input data register (GPIOx_IDR) (x = A,B,C, D)

Offset address: 0x10

Reset value: 0x0000 XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	IDy	R		y = 15..0 This is read-only, the read value corresponds to the state of the I/O port.

7.4.6. GPIO port output data register (GPIOx_ODR) (x = A,B,C, D)

Offset address: 0x14.

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OD15	OD14	OD13	OD12	OD11	OD10	OD9	OD8	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	ODy [1: 0]	RW	0	y = 15..0 These bits can be read and written by software. Note: By writing to the GPIOx_BSRR or GPIOx_BRR registers (x=A,B,C,D), each ODR bit can be independently set/reset.

7.4.7. GPIO port bit set/reset register (GPIOx_BSRR) (x = A,B,C, D)

Offset address: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	BRy	W	0	y = 15..0 Software writable, reading these bits returns 0. 0: No action on the corresponding ODy bit 1: Clears the corresponding ODy bit Note: If both BSy and BRy bits are set, the BSy bit takes effect.
15:0	BSy	W	0	y = 15..0 Software writable, reading these bits returns 0. 0: No action on the corresponding ODy bit 1: Set the corresponding ODy bit

7.4.8. GPIO port configuration lock register (GPIOx_LCKR) (x = A,B,C, D)

This register is used to lock the configuration of the port bits when a correct write sequence is applied to bit 16 (LCKK). The value of bits [15:0] is used to lock the configuration of the GPIO. During the write sequence, LCKR[15:0] cannot be changed. When the LOCK sequence has been applied on a port bit, the value of this port bit can no longer be modified until the next system reset.

Note: A specific write sequence is used to write to the GPIOx_LCKR register. Only word access (32-bit long) is allowed during this locking sequence.

Each lock bit freezes a specific configuration register (control register and alternate function register)

Offset address: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LCK K

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	RW
LCK1 5	LCK1 4	LCK1 3	LCK1 2	LCK1 1	LCK1 0	LCK 9	LCK 8	LCK 7	LCK 6	LCK 5	LCK 4	LCK 3	LCK 2	LCK 1	LCK0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	LCKK	RW	0	This bit can be read any time. It can only be modified using the lock key write sequence. 0: Port configuration lock key not active 1: The port configuration lock key is activated, and the GPIOx_LCKR register is locked until the next system reset. Write timing of lock key: Write {LCKR[16]=1, LCKR[15:0]} Write {LCKR[16]=0, LCKR[15:0]} Write {LCKR[16]=1, LCKR[15:0]} Read LCKR Read LCKR[16] as 1. The last read can be omitted, but this read operation can be used to confirm that the lock key has been activated. Note: - During the write sequence for the lock key operation, the value of LCK[15:0] must not be changed. - Any error in the lock sequence aborts the lock. - After the first lock sequence on any port bit, any read access to the LCKK bit will return '1' until the next system reset or GPIO peripheral soft reset.
15:0	LCKy	RW	0	y = 15..0 These bits are read/write but can only be written when the LCKK bit is '0'. 0: Port configuration is not locked 1: Port configuration is locked

7.4.9. GPIO alternate function low register (GPIOx_AFRL) (x = A, B, C, D)

Offset address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL7 [3:0]				AFSEL6 [3:0]				AFSEL5 [3:0]				AFSEL4 [3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFSEL3 [3:0]				AFSEL2 [3:0]				AFSEL1 [3:0]				AFSEL0 [3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function																
31:0	AFSELY[3:0](y= 7 to 0)	RW	0	These bits are written by software to configure alternate function I/Os. AFSELY selection: <table><tr><td>0000: AF0</td><td>1000: AF8</td></tr><tr><td>0001: AF1</td><td>1001: AF9</td></tr><tr><td>0010: AF21010</td><td>: AF10</td></tr><tr><td>0011: AF31011</td><td>: AF11</td></tr><tr><td>0100: AF41100</td><td>: AF12</td></tr><tr><td>0101: AF51101</td><td>: AF13</td></tr><tr><td>0110: AF61110</td><td>: AF14</td></tr><tr><td>0111: AF71111</td><td>: AF15</td></tr></table>	0000: AF0	1000: AF8	0001: AF1	1001: AF9	0010: AF21010	: AF10	0011: AF31011	: AF11	0100: AF41100	: AF12	0101: AF51101	: AF13	0110: AF61110	: AF14	0111: AF71111	: AF15
0000: AF0	1000: AF8																			
0001: AF1	1001: AF9																			
0010: AF21010	: AF10																			
0011: AF31011	: AF11																			
0100: AF41100	: AF12																			
0101: AF51101	: AF13																			
0110: AF61110	: AF14																			
0111: AF71111	: AF15																			

7.4.10. GPIO alternate function high register (GPIOx_AFRH) (x = A, B, C, D)

Offset Address: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL15 [3:0]				AFSEL14 [3:0]				AFSEL13 [3:0]				AFSEL12 [3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFSEL11 [3:0]				AFSEL10 [3:0]				AFSEL9 [3:0]				AFSEL8 [3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	AFSELY[3:0](y= 8 to 15)	RW	0	These bits are written by software to configure alternate function I/Os. AFSELY selection: 0000:AF0 1000: AF8 0001:AF1 1001: AF9 0010: AF2 1010: AF10 0011: AF3 1011: AF11 0100: AF4 1100: AF12 0101: AF5 1101: AF13 0110: AF6 1110: AF14 0111: AF7 1111: AF15

7.4.11. GPIO Port Bit Reset Register (GPIOx_BRR) (x = A,B,C,D)

Offset address: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	2	11	10	9	8	7	6	5	4	3	2	1	0
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	BRy	W	0	y = 15..0 These bits are write-only. A read to these bits returns the value 0. 0: No action on the corresponding ODRy bit 1: Clears the corresponding ODy bit

8. Peripheral Interconnection

8.1. Introduction

Several peripherals have direct connections between them.

This allows autonomous communication and/or synchronization between peripherals, saving CPU resources thus power consumption.

In addition, these hardware connections remove software latency and allow design of predictable systems.

These interconnects can work in Run, Sleep, Low Power Run, Low Power Sleep, Stop modes, depending on the peripheral.

8.2. Interconnection Details

8.2.1. Timer Input Trigger (ITR)

Purpose

Some TIMx timers are connected together internally for synchronization or connection of timers.

When a timer is in master mode, it can reset, start, stop the slave mode timer.

Triggering signals

After the configurable timer event occurs, the output (from the master device) appears on the TIMx_TRGO signal.

For TIM16 and TIM17 timers that do not have a trigger output, output comparison 1 will be used instead.

Inputs (to slave) are on TIMx_ITR0/ITR1/ITR2/ITR3/ITR4/ITR5 signals.

Table 8-1 timer input trigger

TIMx Internal Trigger	Timer input trigger source (ITR)			
	TIM1	TIM2	TIM3	TIM15
ITR0(TIMx_SMCR.TS=0x0)	-	TIM1_TRGO	TIM1_TRGO	TIM1_TRGO
ITR1(TIMx_SMCR.TS=0x1)	TIM2_TRGO	-	TIM2_TRGO	TIM2_TRGO
ITR2(TIMx_SMCR.TS=0x2)	TIM3_TRGO	TIM3_TRGO	-	TIM3_TRGO
ITR3(TIMx_SMCR.TS=0x3)	TIM15_TRGO	TIM15_TRGO	TIM15_TRGO	-
ITR4(TIMx_SMCR.TS=0x8)	TIM16_OC1	TIM16_OC1	TIM16_OC1	TIM16_OC1
ITR5(TIMx_SMCR.TS=0x9)	TIM17_OC1	TIM17_OC1	TIM17_OC1	TIM17_OC1

Relevant power modes

These interconnections operate in Run, Sleep, Low-power run, and Low-power sleep power modes.

8.2.2. Timer External Trigger (ETR)

Purpose

The ADC analog watchdog signal output, the output of the comparator, and the timer ETR pin can be connected to the timer's external trigger ETR.

Triggering signals

ADC analog watchdog output ADC_AWD, comparator outputs COMP1_OUT, COMP2_OUT.

Table 8-2 Timer External Trigger

TIMx External Trigger (TIMx_AF1.ETRSEL)	Timer external trigger source (ETR)				
	TIM1	TIM2	TIM3	TIM15	PWM
ETR0	TIM1_ETR	TIM2_ETR	TIM3_ETR	TIM15_ETR	PWM_ETR
ETR1	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP1_OUT
ETR2	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP2_OUT
ETR3	-	-	-	-	-
ETR4	-	-	-	-	SYSCLK
ETR5	ADC_AWD	TIM3_ETR	TIM2_ETR	ADC_AWD	PWM_ETR_HS

Relevant power modes

These interconnections operate in Run, Sleep, Low-power run, and Low-power sleep power modes.

8.2.3. Clear timer OCxREF signal**Purpose**

The output of the comparator can be used to clear the OCxREF signals of the advanced timers TIM1, the general timers TIM2, and TIM3.

Triggering signals

Comparator outputs COMP1_OUT, COMP2_OUT.

Table 8-3 Clear Timer OCxREF Signal

OCxREF_CLR	Clear Timer OCxREF (OCxREF_CLR) Source		
	TIM1	TIM2	TIM3
OCREF_CLR0	COMP1_OUT	COMP1_OUT	COMP1_OUT
OCREF_CLR1	COMP2_OUT	COMP2_OUT	COMP2_OUT

Relevant power modes

These interconnections operate in Run, Sleep, Low-power run, and Low-power sleep power modes.

8.2.4. Timer input capture**Purpose**

The output of the comparator can be used as the input capture channel for the advanced timer TIM1 and the general purpose timers TIM2/TIM3/TIM15.

The general purpose timers TIM15/TIM16 can implement clock measurements using input capture channel 1.

Table 8-4 Input Capture Channel 1 (TI1) Source

TI1 (TIMx_TI1SEL.TI1SEL)	Input capture channel 1 (TI1) source					
	TIM1	TIM2	TIM3	TIM15	TIM16	TIM17
TI1_IN0	TIM1_CH1	TIM2_CH1	TIM3_CH1	TIM15_CH1	TIM16_CH1	TIM17_CH1
TI1_IN1	COMP1_OUT	COMP1_OUT	COMP1_OUT	LSECSS	-	-
TI1_IN2	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP1_OUT	MCO	MCO
TI1_IN3	-	-	-	COMP2_OUT	HSE_DIV32	HSE_DIV32
TI1_IN4	-	-	-	-	RTC_WKUP	RTC_WKUP
TI1_IN5	-	-	-	-	LSECSS	LSECSS
TI1_IN6	-	-	-	-	LSI	LSI

Table 8-5 Input Capture Channel 2 (TI2) Source

TI2 (TIMx_TI2SEL.TI2SEL)	Input Capture Channel 2 (TI2) Source			
	TIM1	TIM2	TIM3	TIM15
TI2_IN0	TIM1_CH2	TIM2_CH2	TIM3_CH2	TIM15_CH2
TI2_IN1	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP2_OUT
TI2_IN2	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP1_OUT

Table 8-6 Input Capture Channel 3 (TI3) Source

TI3 (TIMx_TI3SEL.TI3SEL)	Input Capture Channel 3 (TI3) Source		
	TIM1	TIM2	TIM3
TI3_IN0	TIM1_CH3	TIM2_CH3	TIM3_CH3
TI3_IN1	COMP1_OUT	COMP1_OUT	COMP1_OUT
TI3_IN2	COMP2_OUT	COMP2_OUT	COMP2_OUT

Table 8-7 Input Capture Channel 4 (TI4) Source

TI4 (TIMx_TI4SEL.TI4SEL)	Input Capture Channel 4 (TI4) Source		
	TIM1	TIM2	TIM3
TI3_IN0	TIM1_CH4	TIM2_CH4	TIM3_CH4
TI3_IN1	COMP1_OUT	COMP1_OUT	COMP1_OUT
TI3_IN2	COMP2_OUT	COMP2_OUT	COMP2_OUT

Relevant power modes

These interconnections operate in Run, Sleep, Low-power run, and Low-power sleep power modes.

8.2.5. Timer brake**Purpose**

The output of the comparator can be used as the advanced timer TIM1, the general purpose timers TIM15/TIM16/TIM17, and the brake of the PWM.

Triggering signals

The comparator outputs COMP1_OUT, COMP2_OUT, and the brake pin input TIMx_BRK.

Table 8-8 timer brake input

TIMx Brake	Timer Brake (BRK) Source				
	TIM1	TIM15	TIM16	TIM17	PWM
BRK0	TIM1_BRK	TIM2_BRK	TIM3_BRK	TIM15_BRK	PWM_BRK
BRK1	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP1_OUT
BRK2	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP2_OUT

Relevant power modes

These interconnections operate in Run, Sleep, Low-power run, and Low-power sleep power modes.

8.2.6. Timer System Brake**Purpose**

HSE and LSE CSS, CPU LOCKUP, SRAM Parity Error, FLASH ECC Detection Error, and PVD Alarm can generate system errors as timer brakes for TIM1, TIM15, TIM16, TIM17, and PWM modules.

The purpose of the break function is to protect power switches driven by PWM signals from the timers.

Relevant power modes

These interconnections operate in Run, Sleep, Low-power run, and Low-power sleep power modes.

8.2.7. Low-Power Timer LPTIM Trigger**Purpose**

RTC Alarm A/B, TAMP Input Detection, COMP1/2_OUT, and LPTIM1/2_ETR pins can be used as trigger signals to start the LPTIM counter LPTIM1/2.

Table 8-9 LPTIM External Trigger

LPTIMx_CFGR.TRIGSEL[2:0]	Low-Power Timer External Trigger	
	LPTIM1	LPTIM2
EXT_TRIG0	LPTIM1_ETR	LPTIM2_ETR
EXT_TRIG1	RTC_ALARMA	RTC_ALARMA
EXT_TRIG2	RTC_ALARMB	RTC_ALARMB
EXT_TRIG3	TAMP_TRG	TAMP_TRG
EXT_TRIG4	COMP1_OUT	COMP1_OUT
EXT_TRIG5	COMP2_OUT	COMP2_OUT

Relevant power modes

These interconnections operate in run, sleep, low-power run, low-power sleep, and all stop power modes.

8.2.8. Infrared output waveform control (IRTIM)**Purpose**

The TIMx_OC1 output channel of the TIM16 or TIM17 timer generates an infrared output waveform.

Table 8-10 IRTIM control signal assignment

IRTIM waveform control	IRTIM waveform control signal source
Modulated envelope signal	TIM16_OC1
Carrier signal	TIM17_OC1

Relevant power modes

These interconnections operate in Run, Sleep, Low-power run, and Low-power sleep power modes.

8.2.9. Comparator blanking**Purpose**

The advanced-control timer TIM1 and general-purpose timers TIM2, TIM3, TIM15, and PWM can be used as blanking window inputs for COMP1 and COMP2.

Triggering signals

The timer output signal TIMx_OCx/PWM_OC3 is the blanking source input for COMP1/COMP2.

Table 8-11 comparator blanking input

COMPx_CSR.BLANKSEL [2: 0]	Comparator blanking input	
	COMP1	COMP2
BLANK0	-	-
BLANK1	TIM1_OC4	TIM1_OC4
BLANK2	TIM2_OC3	TIM2_OC3
BLANK3	TIM3_OC3	TIM3_OC3

BLANK4	TIM15_OC2	TIM15_OC2
BLANK5	PWM_OC3	PWM_OC3

Relevant power modes

These interconnections operate in Run, Sleep, Low-power run, and Low-power sleep power modes.

8.2.10. ADC hardware trigger input

Purpose

General-purpose timers TIM2, TIM3, and TIM15, advanced timer TIM1, and EXTI can be used to generate ADC trigger events.

Triggering signals

The output (from the timer) is on the TIMx_TRGO, or TIMx_OCx signal events, and the EXTI Line11, EXTI Line15 events.

The input (to the ADC) is on the EXT [15: 0] and JEXT [15: 0] signals, and the selection is selected by registers ADC_CFGR.EXTSEL [3: 0] and ADC_JSQR.JEXTSEL [3: 0], respectively.

Table 8-12 ADC hardware trigger sources

EXTSEL[3:0]/JEXTSEL[3:0]	ADC hardware trigger source	
	Rule trigger	Injection trigger
0	TIM1_TRGO	TIM1_TRGO
1	TIM1_OC1	TIM1_OC1
2	TIM1_OC2	TIM1_OC2
3	TIM1_OC3	TIM1_OC3
4	TIM1_OC4	TIM1_OC4
5	TIM2_TRGO	TIM2_TRGO
6	TIM2_OC1	TIM2_OC1
7	TIM3_OC1	TIM3_OC1
8	TIM3_TRGO	TIM3_TRGO
9	TIM15_TRGO	TIM15_TRGO
10	-	-
11	-	-
12	EXTI_EVT11	EXTI_EVT11
13	EXTI_EVT15	EXTI_EVT15
14	-	-
15	-	-

8.2.11. ADC internal channel source

Purpose

Internal temperature sensor output voltage VTS, internal reference voltage V_{REFINT}, V_{CC}/3, DAC and OPA outputs connected to ADC input channels.

Table 8-13 ADC internal channel sources

ADC_CFGR.AWDCH[5:0]	ADC internal channel source
6'b11011	VTS
6'b11100	V _{REFINT}
6'b11101	V _{CC} / 3
6'b11110	DAC_OUT
6'b11111	OPA_OUT

8.2.12. DAC hardware trigger input

Purpose

General-purpose timers TIM2, TIM3, and TIM15, basic timers TIM6 and TIM7, and EXTI can be used to generate DAC external trigger events.

Triggering signals

Output (from timers) occurs on TIMx_TRGO events and EXTI Line9 events.

Input (to DAC) is located on TSEL[7:0] signals, selected by the DAC_CR1.TSEL[2:0] register respectively.

Table 8-14 DAC trigger sources

DAC_CR1.TSEL[2:0]	DAC trigger source
0	TIM6_TRGO
1	TIM3_TRGO
2	TIM7_TRGO
3	TIM15_TRGO
4	TIM2_TRGO
5	-
6	EXTI_EVT9
7	SWTRIG (software trigger)

9. System configuration controller (SYSCFG)

9.1. SYSCFG main features

The chip has a set of configuration registers, and the main purposes of the system configuration controller are as follows:

- Enable and disable I2C type IO filter
- Enable and disable I2C Fm+ mode
- Map the initial program area according to different boot modes
- DMA peripheral channel selection control
- Analog PAD2 enable
- Enable and disable Noise filter for all GPIOs
- Enable and disable PVD lock
- Enable and disable Cortex®-M0+ LOCKUP
- Flash ECC Lock Enable and Disable
- Enable and disable SRAM parity check
- LED IO control

9.2. SYSCFG registers

9.2.1. SYSCFG configuration register 1 (SYSCFG_CFGR1)

The MEM_MODE bit of this register is used to configure the type of memory address 0x0000 0000 access. These two bits are used to select the software physical remapping and bypass the hardware boot selection. After power-on or NRST pin reset, the values of these two bits are determined by the actual boot mode configuration sampled by the hardware.

Address offset: 0x00

Reset value: 0x0000 000x (x is the memory mode selected by the actual boot mode configuration)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	GPIO_AHB_SE	Res	Res	Res	Res	Res	Res	Res.	Res.
.	.	.	.	.	.	.	L	.	.	.	.	.	.	.	.
							RW								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res.	Res	Res	Res	Res	Res	Res	MEM_MODE[1:0]	
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
														RW	RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	GPIO_AHB_SEL	RW	0	CPU FASTIO or CPU AHB bus access to GPIO register control. 0: CPU FASTIO bus accesses GPIO registers 1: CPU AHB bus accesses GPIO registers Note: DMA can access GPIO registers via the AHB bus by default and is not controlled by this bit
23:2	Reserved	-	-	Reserved
1:0	MEM_MODE [1:0]	RW		Memory mapping selection bit Set and cleared by software. This register controls the mapping of the 0x0000 0000 address in memory. After reset (including only power-on reset, NRST pin reset, and OBL reset), the hardware resamples the Boot pins and the definitions in the option bytes, adopting the actual boot mode configuration value.

				(Note: The timing of sampling the Boot method after reset varies depending on the reset type. For details, consult the application engineer.) X0: Logical address 0x0000 0000 is mapped to the Flash main memory area 01: Logical address 0x0000 0000 is mapped to the Flash system memory area 11: Logical address 0x0000 0000 is mapped to SRAM

9.2.2. SYSCFG configuration register 2 (SYSCFG_CFGR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCL	PVDL	SPL	CLL
												RS	RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3	ECCL	RS	0	ECC lock. This bit is set to 1 by software and can only be cleared by system reset. This bit can be used to enable and lock the connection of FLASH ECC errors to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM. Writing 0 to this bit in software does not change its value. 0: ECC errors are not connected to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM 1: ECC error is connected to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM
2	PVDL	RS	0	PVD lock enable bit Software set, system reset and clear. It can be used to enable and lock the PVD connection to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM, and also lock the PVDE and PLS[2:0] bits in the PWR_CR2 register. Writing 0 to this bit in software does not change its value. 0: PVD interrupt is not connected to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM. The PVDE bit and PLS[2:0] can be written by software. 1: PVD interrupt is connected to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM. The PVDE bit and PLS[2:0] are read-only.
1	SPL	RS	0	SRAM parity lock bit. This bit is set to 1 by software and can only be cleared by system reset. This bit can be used to enable and lock the SRAM1 parity error signal connection to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM. Writing 0 to this bit in software does not change its value. 0: SRAM parity error signal is not connected to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM 1: SRAM parity error signal is connected to the brake input of TIM1/TIM15/TIM16/TIM17/PWM
0	CLL	RS	0	Cortex®-M0+ LOCKUP bit enable bit Software set, system reset and clear. It can enable and lock the Cortex®-M0+ LOCKUP (HardFault) output to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM. Writing 0 to this bit in software does not change its value. 0: Cortex®-M0+ LOCKUP output is not connected to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM 1: Cortex®-M0+ LOCKUP output is connected to the system brake input of TIM1/TIM15/TIM16/TIM17/PWM

9.2.3. SYSCFG configuration register 3 (SYSCFG_CFGR3)

Address offset: 0x08

Reset value: 0x3F3F_3F3F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	DMA4_MAP [5:0]						Res.	Res.	DMA3_MAP [5:0]					
		RW	RW	RW	RW	RW	RW				RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	DMA2_MAP [5:0]						Res.	Res.	DMA1_MAP [5:0]					
		RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:24	DMA4_MAP [5:0]	RW	6'b111111	Channel 4 peripheral request mapping. Refer to the Channel 1 description for definitions.
23:22	Reserved	-	-	Reserved
21:16	DMA3_MAP [5:0]	RW	6'b111111	Channel 3 peripheral request mapping. Refer to Channel 1 description for definition.
15:14	Reserved	-	-	Reserved
13:8	DMA2_MAP [5:0]	RW	6'b111111	Channel 2 peripheral request mapping. Refer to the Channel 1 description for definitions.
7:6	Reserved	-	-	Reserved
5:0	DMA1_MAP [5:0]	RW	6'b111111	Channel 1 peripheral request mapping. 000000: ADC 000001: DAC 000010: Reserved 000011: SPI1_RX 000100: SPI1_TX 000101: SPI2_RX 000110: SPI2_TX 000111: USART1_RX 001000: USART1_TX 001001: USART2_RX 001010: USART2_TX 001011: UART1_RX 001100: UART1_TX 001101: LPUART1_RX 001110: LPUART1_TX 001111: I2C1_RX 010000: I2C1_TX 010001: I2C2_RX 010010: I2C2_TX 010011: TIM1_CH1 010100: TIM1_CH2 010101: TIM1_CH3 010110: TIM1_CH4 010111: TIM1_COM 011000: TIM1_TRG 011001: TIM1_UP 011010: TIM2_CH1 011011: TIM2_CH2 011100: TIM2_CH3 011101: TIM2_CH4 011110: TIM2_UP 011111: TIM2_TRG 100000: TIM3_CH1 100001: TIM3_CH2 100010: TIM3_CH3 100011: TIM3_CH4 100100: TIM3_UP 100101: TIM3_TRG 100110: TIM6_UP 100111: TIM7_UP 101000: TIM15_CH1 101001: TIM15_CH2 101010: TIM15_UP 101011: TIM15_TRG 101100: TIM15_COM 101101: TIM16_CH1 101110: TIM16_UP 101111: TIM17_CH1 110000: TIM17_UP

				110001: Reserved 110010: LPUART2_RX 110011: LPUART2_TX 110100: PWM_CH1 110101: PWM_CH2 110110: PWM_CH3 110111: PWM_CH4 111000: PWM_UP 111001: UART2_RX 111010: UART2_TX

9.2.4. SYSCFG configuration register 4 (SYSCFG_CFGR4)

Address offset: 0x0C

Reset value: 0x003F_3F3F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA7_MAP [5:0]					
										RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	DMA6_MAP [5:0]						Res.	Res.	DMA5_MAP [5:0]					
		RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	Reserved
21:16	DMA7_MAP [5:0]	RW	6'b111111	Channel 7 peripheral request mapping. Refer to the Channel 1 description for definitions.
15:14	Reserved	-	-	Reserved
13:8	DMA6_MAP [5:0]	RW	6'b111111	Channel 6 peripheral request mapping. Refer to the Channel 1 description for definitions.
7:6	Reserved	-	-	Reserved
5:0	DMA5_MAP [5:0]	RW	6'b111111	Channel 5 peripheral request mapping. Refer to the Channel 1 description for definitions.

9.2.5. GPIOA noise filter enable register (PA_ENS)

Offset address: 0x20

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PA_ENS[x]	RW	0x0000	Noise filter enable for GPIOA0~15 0: Turn off noise filtering function 1: Turn on noise filtering function

9.2.6. GPIOB noise filter enable register (PB_ENS)

Offset address: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PB_ENS[x]	RW	0x0000	Noise filter enable for GPIOB0~15 0: Turn off noise filtering function 1: Turn on noise filtering function

9.2.7. GPIOC noise filter enable register (PC_ENS)

Offset address: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PC_ENS[x]	RW	0x0000	Noise filter enable for GPIOC0~15 0: Turn off noise filtering function 1: Turn on noise filtering function

9.2.8. GPIOD noise rate enable register (PD_ENS)

Offset address: 0x2C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PD_ENS[x]	RW	0x0000	Noise filter enable for GPIOD0~15 0: Turn off noise filtering function 1: Turn on noise filtering function

9.2.9. I2C type IO configuration register (SYSCFG_IOCFG)

Address offset: 0x38

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	PD_EIIC[3:0]				Res.	PC_EIIC[8:0]								
		RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD_EIIC_11	PD_EIIC_10	PB_EIIC[5:0]						Res.	Res.	Res.	Res.	PA_EIIC[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:26	PD_EIIC[3:0]	RW	0	PD EIIC signal control. Used as _I2C type IO. Bit0: control PD6 Bit1: control PD7 Bit2: Reserved Bit3: Reserved

25	Reserved	-	-	Reserved
24:16	PC_EIIC[8:0]	RW	0	PC EIIC signal control. Used as _I2C type IO. Bit0: control PC5 Bit1: control PC6 Bit2: control PC7 Bit3: control PC9 Bit4: control PC10 Bit5: control PC12 Bit6: control PC13 Bit7: control PC14 Bit8: control PC15
15	PD_EIIC_11	RW	0	PD11 EIIC signal control. Used as _I2C type IO.
14	PD_EIIC_10	RW	0	PD10 EIIC signal control. Used as _I2C type IO.
13:8	PB_EIIC[5:0]	RW	0	PB IIC signal control. Used as _I2C type IO. Bit0: control PB4 Bit1: control PB5 Bit2: control PB8 Bit3: control PB9 Bit4: control PB12 Bit5: control PB13
7:4	Reserved	-	-	Reserved
3:0	PA_EIIC[3:0]	RW	0	PA EIIC signal control. Used as _I2C type IO. Bit0: control PA11 Bit1: control PA12 Bit2: control PA13 Bit3: control PA14

9.2.10. LED IO configuration register (SYSCFG_LEDCFG)

Address offset: 0x3C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC_ENSEG[1:0]		PB_ENSEG [13: 0]													
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB_EHS[7:0]								Res.	Res.	Res.	Res.	SEGIS_EN[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	PC_ENSEG[1:0]	RW	0	LED SEG port constant current enable control. Bit0: PC0 (LED_SEG14) Bit1: PC1 (LED_SEG15)
29:16	PB_ENSEG [13: 0]	RW	0	LED SEG port constant current enable control. Bit0: PB2 (LED_SEG0) Bit1: PB3 (LED_SEG1) Bit2: PB4 (LED_SEG2) Bit3: PB5 (LED_SEG3) Bit4: PB6 (LED_SEG4) Bit5: PB7 (LED_SEG5) Bit6: PB8 (LED_SEG6) Bit7: PB9 (LED_SEG7) Bit8: PB10 (LED_SEG8) Bit9: PB11 (LED_SEG9) Bit10: PB12 (LED_SEG10) Bit11: PB13 (LED_SEG11) Bit12: PB14 (LED_SEG12) Bit13: PB15 (LED_SEG13)
15:8	PB_EHS[7:0]	RW	0	LED COM 120mA sink current control. Bit0: Controls PB2 (LED_COM0) Bit1: Controls PB3 (LED_COM1) Bit2: Controls PB4 (LED_COM2) Bit3: Controls PB5 (LED_COM3) Bit4: Controls PB6 (LED_COM4) Bit5: Controls PB7 (LED_COM5) Bit6: Controls PB8 (LED_COM6)

				Bit7: Controls PB9 (LED_COM7)
7:4	Reserved	-	-	Reserved
3:0	SEGIS_EN	RW	0	LED SEG port constant current source circuit enable. Bit0: Constant current source 0 circuit switch (for LED SEG0~2) Bit1: Constant current source 1 circuit switch (for LED SEG3~5) Bit2: Constant current source 2 circuit switch (for LED SEG6~10) Bit3: Constant current source 3 circuit switch (for LED SEG11~15)

9.2.11. GPIOA analog PAD2 enable register (PA_ANA2EN)

Offset address: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PA_ANA2EN[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PA_ANA2EN[x]	RW	0x0000	IO port A PAD_ANA2 enable 0: PAD_ANA2 Disabled 1: PAD_ANA2 Enabled Note 1: This register needs to be enabled when the IO is configured in analog mode. If the IO is configured in non-analog mode, this register should be set to 0. Note 2: Whether configuration is required, refer to the description below the PD_ANA2EN register.

9.2.12. GPIOB Analog PAD2 Enable Register (PB_ANA2EN)

Offset Address: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB_ANA2EN [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PB_ANA2EN[x]	RW	0x0000	IO Port B PAD_ANA2 Enable 0: PAD_ANA2 Disabled 1: PAD_ANA2 Enabled Note 1: This register needs to be enabled when the IO is configured in analog mode. If the IO is configured in non-analog mode, this register should be set to 0. Note 2: Whether configuration is required, refer to the description below the PD_ANA2EN register.

9.2.13. GPIOC Analog PAD2 Enable Register (PC_ANA2EN)

Offset Address: 0x48

Reset Value: 0x0000 0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC_ANA2EN [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PC_ANA2EN[x]	RW	0x0000	IO Port C PAD_ANA2 Enable 0: PAD_ANA2 Disabled 1: PAD_ANA2 Enabled Note 1: This register needs to be enabled when the IO is configured in analog mode. If the IO is configured in non-analog mode, this register should be set to 0. Note 2: Whether configuration is required, refer to the description below the PD_ANA2EN register.

9.2.14. GPIOD analog PAD2 enable register (PD_ANA2EN)

Offset address: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PD_ANA2EN [13:0]													
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13:0	PD_ANA2EN[x]	RW	0x000	IO Port D PAD_ANA2 enable 0: PAD_ANA2 Disabled 1: PAD_ANA2 Enabled Note 1: This register needs to be enabled when the IO is configured in analog mode. If the IO is configured in non-analog mode, this register should be set to 0. Note 2: Whether configuration is required, refer to the description below the PD_ANA2EN register.

For analog PAD_ANA2, the following requirements must be met:

- When used for analog functions, only comparator COMP inputs and operational amplifier OPA input/output require Px_ANA2EN=1. Other pins serving analog functions should be configured with Px_ANA2EN=0:
--COMP1 INP: PC15, PD0, PD1
--COMP1 INM: PA1, PA13, PC5
--COMP2 INP: PB14, PC3, PC4
--COMP2 INM: PA13, PB15, PC0
--OPA INP: PC2, PD3
--OPA INM: PC1, PD2
--OPA OUT: PC8, PC9, PD1, PD4

9.2.15. SRAM control status register (SYSCFG_SCSR)

Address offset: 0x70

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERR_RSTEN
							RC_W1								RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8	SPF	RC_W1	0	SRAM parity error flag bit. This bit is set to 1 by hardware when an SRAM parity error occurs. The software writes 1 to this bit to clear it 0: No parity error detected 1: Parity error detected
7:1	Reserved	-	-	Reserved
0	PERR_RSTEN	RW	0	Response control when SRAM parity error occurs. 0: Generate NMI interrupt when SRAM parity error occurs 1: Generate system reset when SRAM parity error occurs

9.2.16. SRAM write protection register (SYSCFG_SWPR)

Address offset: 0x74

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
P31 WP	P30 WP	P29 WP	P28 WP	P27 WP	P26 WP	P25 WP	P24 WP	P23 WP	P22 WP	P21 WP	P20 WP	P19 WP	P18 WP	P17 WP	P16 WP
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15 WP	P14 WP	P13 WP	P12 WP	P11 WP	P10 WP	P9W P	P8W P	P7W P	P6W P	P5W P	P4W P	P3W P	P2W P	P1W P	P0W P
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	SRAM_PxWP	RW	0	SRAM page x write protection These bits are set to 1 by software and can only be cleared by system reset. 0: SRAM page x write protection disabled 1: SRAM page x write protection enabled

10. Direct Memory Access (DMA)

10.1. Introduction to DMA

Direct memory access (DMA) is used to provide a high-speed data transfer between peripherals and memory as well as from memory to memory. Without CPU intervention, data can be moved quickly through DMA, saving CPU resources for other operations.

The DMA controller have 7 channels in total, each one dedicated to manage memory access requests from one or more peripherals. The DMA controller includes an arbiter to handle DMA requests, which manages the priority of each DMA request.

10.2. DMA main features

- 7 configurable channels
- Each channel can be configured to connect to hardware DMA requests from any peripheral, and each channel also supports software triggering.
- Priority between multiple requests on the same DMA module is software-programmable. For equal priorities, hardware resolves conflicts (lower channel number = higher priority).
- The transfer sizes of the source and destination are independent (byte, half word and word), simulating packing and unpacking. Source and destination addresses must be aligned to the transfer width.
- Programmable address modes: Increment, decrement, or fixed.
- Each channel has 4 event flags (Transfer Complete, Block Transfer Complete, Half Block Transfer Complete, Transfer Error), which are logically OR and become a separate interrupt request
- Support transfers between memory to memory, peripheral to memory, memory to peripheral and peripheral to peripheral.
- SRAM, APB, and AHB peripherals can serve as both source and destination for access, while FLASH can only be a source but not a destination
- Support single-trigger mode and four circular modes:
 - Peripheral address retained, memory address retained
 - Peripheral address reloaded, memory address retained
 - Peripheral address retained, memory address reloaded
 - Both peripheral and memory addresses reload
- Number of programmable transmissions in single mode 0 ~ 65535
- Loop mode supports infinite loop and finite loop (1 ~ 255)
- Support single transfer and bulk transfer
 - Single transmission: 1 data transfer and 1 ACK reply;
 - Batch transfer: reply to ACK once after transferring the configured data amount (release the bus after all data transfers are completed);
- Two transfer modes

- Fast mode: Holds the bus until all data is transferred.
- Round-robin mode: Releases the bus after each transfer for re-arbitration.
- In circular mode, supports pausing transfer after entering block transfer completion interrupt

10.3. DMA functional description

10.3.1. DMA block diagram

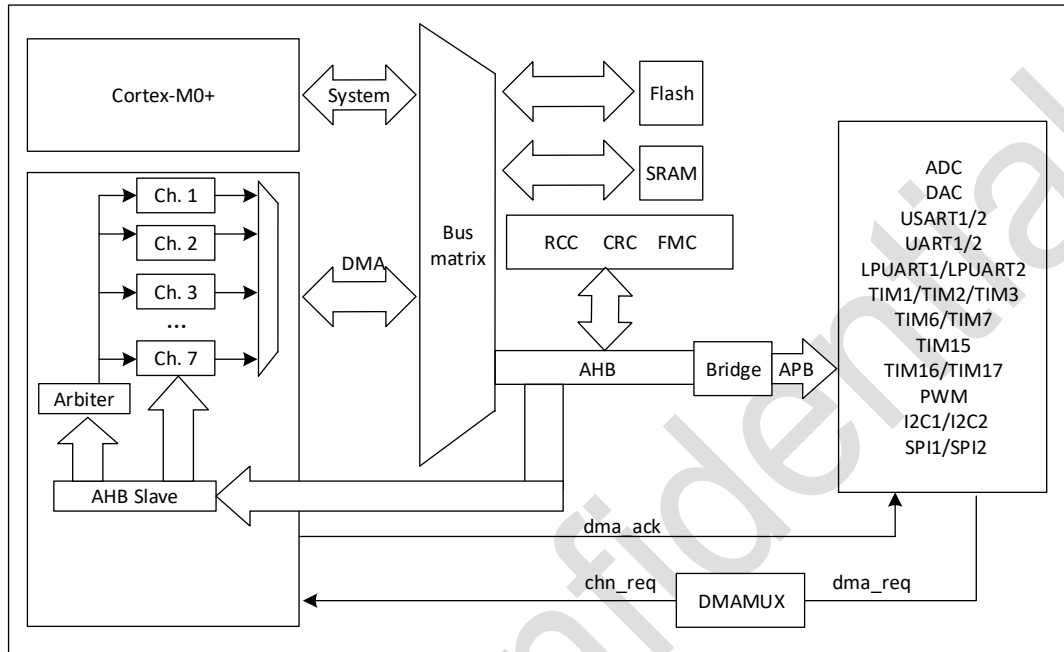


Figure 10-1 DMA block diagram

10.3.2. CPU and DMA bus sharing

The DMA controller and CPU share the AMBA data bus and perform direct memory data transfer. When the CPU and DMA access the same target (RAM or peripheral) at the same time, the DMA request will suspend the CPU from accessing the system bus for several cycles, and the bus arbiter performs cyclic scheduling to ensure that the CPU can get at least half of the system bus bandwidth.

10.3.3. DMA transmission

After completing an event, the peripheral sends a request signal to the DMA controller. The DMA controller serves the request, depending on the priority of the channel associated to this peripheral request. When the DMA controller starts to access the requesting peripheral, the DMA controller sends it an answer signal after the transmission is completed. The peripheral releases its request as soon as it gets the acknowledge from the DMA controller. Once the request is de-asserted by the peripheral, the DMA controller releases the acknowledge. The peripheral may order a further single request and initiate another single DMA transfer.

In summary, each DMA transfer consists of three operations:

- Fetch data from the peripheral data register or from the current peripheral/memory address register, where the address is the peripheral address or memory location specified by the DMA_CPARx or DMA_CMARx registers.

- Store data to the peripheral register or the memory address indicated by the current peripheral/memory address register, where the address is the peripheral address or memory location specified by the DMA_CPARx or DMA_CMARx registers.
- Performs a decrement of the DMA_CNDTRx register, which indicates the number of outstanding operations.

10.3.4. DMA Arbiter

The arbiter initiates access to the peripheral/memory according to the priority of the channel request.

The priorities are managed in two stages:

- Software: The priority of each channel can be set in the DMA_CCRx register, with 7 levels (the number of priority types equals the number of channels):

- 0
- 1
- 2
- 3
- 4
- 5
- 6

Priority 6>5>4>3>2>1>0.

- Hardware: If 2 requests have the same software priority, the lower-numbered channel has higher priority than the higher-numbered channel. For example, channel 2 gets priority over channel 3.

10.3.5. DMA channel

Each channel may handle a DMA transfer between a peripheral register located at a fixed address, and a memory address. The amount of data transferred by DMA is programmable, up to 65535. A register containing the amount of data to be transferred, decremented after each transfer.

10.3.5.1. Programmable data size

The transmission data width of peripherals and memory can be programmed by the MSIZE and PSIZE bits in the DMA_CCRx register.

10.3.5.2. Programmable address increment mode

By setting the PINC and MINC flag bits in the DMA_CCRx register, the pointers of peripherals and memory can selectively complete auto-increment after each transfer. When the bit delta mode is set, the next address to be transferred will be the previous address plus the delta value, which is 1, 2, 4 depending on the selected data width.

The first transfer address is stored in the DMA_CPARx/DMA_CMARx registers. During transfer, these registers hold their initial values and the software cannot change and read out the address currently being transferred (it is in the internal current peripheral/memory address register).

When the channel is configured for monolithic transmission, no DMA operation will be generated after the transmission is completed (i.e., the transmission count becomes 0). The DMA channel must be disabled in order to reload a new number of data items into the DMA_CNDTRx register.

In loop mode, at the end of the last transfer, the contents of the DMA_CNDTRx register are automatically reloaded to its initial value, and the internal current peripheral/memory address register holds the address or reloads the address according to the configuration of MNORLD and PNORLD.

10.3.5.3. the circular mode

Circular mode is used to handle circular buffers and continuous data transfers. This feature is enabled using the CIRC bit in the DMA_CCRx register. When the circular mode is started and the number of data transmissions becomes 0, the initial value set when the channel is configured will be automatically restored, and the DMA operation will continue.

There are four circular mode address regeneration methods:

- Peripheral address retained, memory address retained
- Peripheral address reloaded, memory address retained
- Peripheral address retained, memory address reloaded
- Both peripheral and memory addresses reload

The cycle mode is divided into the following two cycle times:

- Infinite loop: Channel enabled, DMA_CCRx.CIRC is 1, and DMA_CCCFGRx.NBT is 0, with single-block transfer data volume being DMA_CNDTRx.NDT;
- Finite loop: Channel enabled, DMA_CCRx.CIRC is 1, and DMA_CCCFGRx.NBT is not 0.
 - When DMA_CCCFGRx.NBT is greater than 0, the single-block transfer data volume is DMA_CNDTRx.NDT;
 - When DMA_CCCFGRx.NBT equals 0, if DMACCCEGRx.NDTL equals 0, the transfer ends; if DMACCCEGRx.NDTL is greater than 0, it continues to transfer DMA_CCCFGRx.NDTL number of data once before ending the transfer;

10.3.5.4. Memory-to-memory mode

The operation of the DMA channel can be performed without peripheral requests. This operation is memory-to-memory mode.

When the MEM2MEM bit in the DMA_CCRx register is set, the DMA transfer will start immediately after the EN bit in the DMA_CCRx register is set by software to enable the DMA channel. The transfer stops once the DMA_CNDTRx register reaches zero.

By configuring the DMC_CCRx.MEMARB register, the memory-to-memory mode supports the following two modes:

- Fast mode: After obtaining the arbitration, it always occupies the bus until the bus is released after all data is transferred. This mode cannot be used simultaneously with the loop mode;
- Rotation mode: After transmitting data once, the bus is released for re-arbitration. In this mode, other channels also have the opportunity to obtain arbitration;

10.3.5.5. Transmission Type

According to the transmission requirements of peripheral devices, the following two data transmission methods are supported

- Single transfer: Transfer data once and reply with one dma_ack;

- Bulk transfer: Transmit NDT sub-data and reply with one dma_ack. The channel using this transfer method will release the bus only after all data transmission is completed;

10.3.5.6. Pause after block transmission completion interrupt

In the cyclic mode, DMA_CCRx.BTCSUSP can be set to 1 so that the channel suspends data transmission after entering the block transmission completion interrupt and resumes data transmission after clearing the block transmission completion interrupt.

10.3.5.7. Channel configuration steps

Configure the DMA channel as follows:

- Set the peripheral register address in the DMA_CPARx register. When a peripheral data transfer request occurs, this address will be the source or destination of the data transfer.
- Set the memory address in the DMA_CMARx register. When a peripheral data transfer request occurs, the transferred data will be read from or written to this address.
- Configure the total number of data to transfer in the DMA_CNDTRx register. After each data transfer, this value is decremented.
- The priority of the channel is set in the PL bit of the DMA_CCRx register.
- In the DMA_CCRx register, the direction of data transfer, the cyclic mode, the incremental mode of the peripheral and memory, the data width of the peripheral and memory, the interrupt generated by half of the transfer or the interrupt generated by completion of the transfer are set.
- Set the number of cycle transfers in the DMA_CCCFGRx register. In a finite cycle, this value decrements after each block is transmitted.
- Activate the channel by setting the EN bit in the DMA_CCRx register.

A channel, as soon as enabled, may serve any DMA request from the peripheral connected to this channel.

When half of the data is transmitted, the Half Block Transfer Flag (HBTIF) is set to 1. When the Allow Half Block Transfer Interrupt Bit (HBTIE) is set, an interrupt request will be generated. After the single block data transmission is completed, the block transfer completion flag (BTCIF) is set to 1, and when the allow block transfer completion interrupt bit (BTCIE) is set, an interrupt request will be generated. After all the transmission of multiple blocks of data is completed, the transmission completion flag (TCIF) is set to 1, and when the transmission completion allow interrupt bit (TCIE) is set, an interrupt request will be generated.

10.3.6. Programmable data width, data alignment, and endianness

When PSIZE and MSIZE are not equal, the DMA controller performs some data alignments as described in the table below.

Table 10-2 Data width and endian (PINC = MINC = 1)

Source Width	Destination Width	Transmission Number	Source: address/data	DMA transfers	Destination: address/data
8	8	4	0x0/B0	1: Read B0 [7: 0] at 0x0, write B0 [7: 0] at 0x0	0x0/B0

			0x1/B1 0x2/B2 0x3/B3	2: Read B1 [7: 0] at 0x1, write B1 [7: 0] at 0x1 3: Read B2 [7: 0] at 0x2, write B2 [7: 0] at 0x2 4: Read B3 [7: 0] at 0x3, write B3 [7: 0] at 0x3	0x1/B1 0x2/B2 0x3/B3
8	16	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1: Read B0 [7: 0] at 0x0, write 00B0 [7: 0] at 0x0 2: Read B1 [7: 0] at 0x1, write 00B1 [7: 0] at 0x2 3: Read B2 [7: 0] at 0x2, write 00B2 [7: 0] at 0x4 4: Read B3 [7: 0] at 0x3, write 00B3 [7: 0] at 0x6	0x0/00B0 0x2/00B1 0x4/00B2 0x6/00B3
8	32	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1: Read B0 [7: 0] at 0x0, write 000000B0 [31: 0] at 0x0 2: Read B1 [7: 0] at 0x1, write 000000B1 [31: 0] at 0x4 3: Read B2 [7: 0] at 0x2, write 000000B2 [31: 0] at 0x8 4: Read B3 [7: 0] at 0x3, write 000000B3 [31: 0] at 0xC	0x0/000000B0 0x4/000000B1 0x8/000000B2 0xC/000000B3
16	8	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: Read B1B0 [15: 0] at 0x0, write B0 [7: 0] at 0x0 2: Read B3B2 [15: 0] at 0x2, write B2 [7: 0] at 0x1 3: Read B5B4 [15: 0] at 0x4, write B4 [7: 0] at 0x2 4: Read B7B6 [15: 0] at 0x6, write B6 [7: 0] at 0x3	0x0/B0 0x1/B2 0x2/B4 0x3/B6
16	16	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: Read B1B0 [15: 0] at 0x0, write B1B0 [15: 0] at 0x0 2: Read B3B2 [15: 0] at 0x2, write B3B2 [15: 0] at 0x2 3: Read B5B4 [15: 0] at 0x4, write B5B4 [15: 0] at 0x4 4: Read B7B6 [15: 0] at 0x6, write B7B6 [15: 0] at 0x6	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6
16	32	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: Read B1B0 [7: 0] at 0x0, write 0000B1B0 [31: 0] at 0x0 2: Read B3B2 [7: 0] at 0x2, write 0000B3B2 [31: 0] at 0x4 3: Read B5B4 [7: 0] at 0x4, write 0000B5B4 [31: 0] at 0x8 4: Read B7B6 [7: 0] at 0x6, write 0000B7B6 [31: 0] at 0xC	0x0/0000B1B0 0x4/0000B3B2 0x8/0000B5B4 0xC/0000B7B6
32	8	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1: Read B3B2B1B0 [31: 0] at 0x0, write B0 [7: 0] at 0x0 2: Read B7B6B5B4 [31: 0] at 0x4, write B4 [7: 0] at 0x1 3: Read BBBAB9B8 [31: 0] at 0x8, write B8 [7: 0] at 0x2 4: Read BFBEBDBC [31: 0] at 0xc, write BC [7: 0] at 0x3	0x0/B0 0x1/B4 0x2/B8 0x3/BC
32	16	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1: Read B3B2B1B0 [31: 0] at 0x0, write B1B0 [7: 0] at 0x0 2: Read B7B6B5B4 [31: 0] at 0x4, write B5B4 [7: 0] at 0x2 3: Read BBBAB9B8 [31: 0] at 0x8, write B9B8 [7: 0] at 0x4 4: Read BFBEBDBC [31: 0] at 0xc, write BDBC [7: 0] at 0x6	0x0/B1B0 0x2/B5B4 0x4/B9B8 0x6/BDBC
32	32	4	0x0/B3B2B1B0 0x4/B7B6B5B4	1: Read B3B2B1B0 [31: 0] at 0x0, write B3B2B1B0 [7: 0] at 0x0	0x0/B3B2B1B0 0x4/B7B6B5B4

			0x8/BBBAB9B8 0xC/BFBEBDBC	2: Read B7B6B5B4 [31: 0] at 0x4, write B7B6B5B4 [7: 0] at 0x2 3: Read BBBAB9B8 [31: 0] at 0x8, write BBBAB9B8 [7: 0] at 0x4 4: Read BFBEBDBC [31: 0] at 0xc, write BFBEBDBC [7: 0] at 0x6	0x8/BBBAB9B8 0xC/BFBEBDBC

10.3.6.1. Byte/half-word access is not supported

If DMA writes in bytes or half words to an AHB device that does not support byte or half word writing operation (i.e. HSIZE is not suitable for this module), no error occurs, DMA will write 32-bit HWDATA data according to the following two examples:

- To write the half-word 0xABCD, the DMA controller sets the HWDATA bus to 0xABCDABCD with a half-word data size (HSIZE = Half-word).
- To write the byte 0xAB, the DMA controller sets the HWDATA bus to 0xABABABAB with a byte data size (HSIZE = Byte).

Assuming the AHB/APB bridge is a 32-bit AHB slave, it does not handle the HSIZE parameter. It will transfer any byte or half-word on the AHB to the APB as 32-bit in the following manner:

- A write byte data '0xB0' operation on an AHB for address 0x0 (or 0x1, 0x2, or 0x3) will be translated to a write data '0xB0B0B0B0' operation on an APB for address 0x0.
- A write half-word data '0xB1B0' operation on an AHB to address 0x0 (or 0x2) will be translated to a write data '0xB1B0B1B0' operation on an APB to address 0x0.

10.3.7. DMA error management

A DMA transfer error is generated when reading from or writing to a reserved address space. When a DMA transfer error occurs during a DMA read or write access, the faulty channel x is automatically disabled through a hardware clear of its EN bit in the corresponding DMA_CCRx register. At this time, the Transfer Error Interrupt Flag bit (TEIF) corresponding to the channel in the DMA_IFR register will be set, and an interrupt will be generated if the Transfer Error Interrupt Allow bit is set in the DMA_CCRx register.

10.4. DMA interrupts

Each DMA channel can generate interrupts when DMA block transmission is halfway through, block transmission is complete, transmission is complete, and transmission errors. Separate interrupt enable bits are available for flexibility.

Table 10-3 DMA interrupt request

Interrupt event	Event flag	Enable control bit
More than half of block transmission	HBTIF	HBTIE
Block transfer complete	BTCIF	BTCIE
Transfer complete	TCIF	TCIE
Transfer error	TEIF	TEIE

Note: When the DMA_CNDTRx register is 1, the HBTIFx bit is not set

10.5. DMA peripheral request mapping

Table 10-4 DMA peripheral channel requests

Peripherals	Channel 1~7 (Each channel can be configured as the following requests)
ADC	ADC
DAC	DAC
SPI	SPI1_RX SPI1_TX SPI2_RX SPI2_TX
USART UART LPUART	USART1_RX USART1_TX USART2_RX USART2_TX UART1_RX UART1_TX UART2_RX UART2_TX LPUART1_RX LPUART1_TX LPUART2_RX LPUART2_TX
I2C	I2C1_RX I2C1_TX I2C2_RX I2C2_TX
TIM1	TIM1_CH1 TIM1_CH2 TIM1_CH3 TIM1_CH4 TIM1_UP TIM1_TRG TIM1_COM
TIM2	TIM2_CH1 TIM2_CH2 TIM2_CH3 TIM2_CH4 TIM2_UP TIM2_TRG
TIM3	TIM3_CH1 TIM3_CH2 TIM3_CH3 TIM3_CH4 TIM3_UP TIM3_TRG
TIM6	TIM6_UP
TIM7	TIM7_UP
TIM15	TIM15_CH1 TIM15_CH2 TIM15_COM TIM15_UP TIM15_TRG
TIM16	TIM16_CH1 TIM16_UP
TIM17	TIM17_CH1 TIM17_UP
PWM	PWM_CH1 PWM_CH2 PWM_CH3 PWM_CH4 PWM_UP

10.6. DMA registers

10.6.1. DMA interrupt status register (DMA_ISR)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TEIF 7	HBTIF 7	BTCIF 7	TCIF 7	TEIF 6	HBTIF 6	BTCIF 6	TCIF 6	TEIF 5	HBTIF 5	BTCIF 5	TCIF 5
				R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEIF 4	HBTIF 4	BTCIF 4	TCIF 4	TEIF 3	HBTIF 3	BTCIF 3	TCIF 3	TEIF 2	HBTIF 2	BTCIF 2	TCIF 2	TEIF 1	HBTIF 1	BTCIF 1	TCIF 1
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27	TEIF7	R	0	Channel 7 Transmission Error (TE) flag. Hardware set, software writes DMA_CTEIF7 = 1 to clear. 0: No TE event 1: TE event occurred
26	HBTIF7	R	0	Channel 7 Half Block Transfer (HBT) flag. Hardware set, software writes DMA_CHBTIF7 = 1 to clear. 0: No HBT event 1: HBT event occurred
25	BTCIF7	R	0	Channel 7 Block Transfer Complete (BTC) flag. Hardware set, software writes DMA_CBTCIF7 = 1 to clear. 0: No BTC event 1: BTC event occurs
24	TCIF7	R	0	Channel 7 All Block Transfer Complete (TC) interrupt flag. Hardware set, software writes DMA_CTCIF7 = 1 to clear. 0: No TC event 1: TC event occurred
23	TEIF6	R	0	Channel 6 Transmission Error (TE) flag. Hardware set, software writes DMA_CTEIF6 = 1 to clear. 0: No TE event 1: TE event occurred
22	HBTIF6	R	0	Channel 6 Half Block Transfer (HBT) completion flag. Hardware set, software writes DMA_CHBTIF6 = 1 to clear. 0: No HBT event 1: HBT event occurred
21	BTCIF6	R	0	Channel 6 Block Transfer Complete (BTC) flag. Hardware set, software writes DMA_CBTCIF6 = 1 to clear. 0: No BTC event 1: BTC event occurs
20	TCIF6	R	0	Channel 6 All Block Transfer Complete (TC) interrupt flag. Hardware set, software writes DMA_CTCIF6 = 1 to clear. 0: No TC event 1: TC event occurred
19	TEIF5	R	0	Channel 5 Transmission Error (TE) flag. Hardware set, software writes DMA_CTEIF5 = 1 to clear. 0: No TE event 1: TE event occurred
18	HBTIF5	R	0	Channel 5 Half Block Transfer (HBT) completion flag. Hardware set, software writes DMA_CHBTIF5 = 1 to clear. 0: No HBT event

				1: HBT event occurred
17	BTCIF5	R	0	Channel 5 Block Transfer Complete (BTC) flag. Hardware set, software writes DMA_CBTCIF5 = 1 to clear. 0: No BTC event 1: BTC event occurs
16	TCIF5	R	0	Channel 5 All Block Transfer Complete (TC) interrupt flag. Hardware set, software writes DMA_CTCIF5 = 1 to clear. 0: No TC event 1: TC event occurred
15	TEIF4	R	0	Channel 4 Transmission Error (TE) flag. Hardware sets the bit, software writes DMA_CTEIF4=1 to clear it. 0: No TE event 1: TE event occurred
14	HBTIF4	R	0	Channel 4 Half Block Transfer Complete (HBT) flag. Hardware set, software writes DMA_CHBTIF4 = 1 to clear. 0: No HBT event 1: HBT event occurred
13	BTCIF4	R	0	Channel 4 Block Transfer Complete (BTC) flag. Hardware set, software writes DMA_CBTCIF4 = 1 to clear. 0: No BTC event 1: BTC event occurs
12	TCIF4	R	0	Channel 4 All Block Transfer Complete (TC) interrupt flag. Hardware set, software writes DMA_CTCIF4 = 1 to clear. 0: No TC event 1: TC event occurred
11	TEIF3	R	0	Channel 3 Transmission Error (TE) flag. Hardware set, software writes DMA_CTEIF3 = 1 to clear. 0: No TE event 1: TE event occurred
10	HBTIF3	R	0	Channel 3 Half Block Transfer (HBT) flag. Hardware set, software writes DMA_CHBTIF3 = 1 to clear. 0: No HBT event 1: HBT event occurred
9	BTCIF3	R	0	Channel 3 Block Transfer Complete (BTC) flag. Hardware set, software writes DMA_CBTCIF3 = 1 to clear. 0: No BTC event 1: BTC event occurs
8	TCIF3	R	0	Channel 3 All Block Transfer Complete (TC) interrupt flag. Hardware set, software writes DMA_CTCIF3 = 1 to clear. 0: No TC event 1: TC event occurred
7	TEIF2	R	0	Channel 2 Transmission Error (TE) flag. Hardware set, software writes DMA_CTEIF2 = 1 to clear. 0: No TE event 1: TE event occurred
6	HBTIF2	R	0	Channel 2 Half Block Transfer (HBT) flag. Hardware set, software writes DMA_CHBTIF2 = 1 to clear. 0: No HBT event 1: HBT event occurred
5	BTCIF2	R	0	Channel 2 Block Transfer Complete (BTC) flag. Hardware set, software writes DMA_CBTCIF2 = 1 to clear. 0: No BTC event 1: BTC event occurs

4	TCIF2	R	0	Channel 2 All Block Transfer Complete (TC) interrupt flag. Hardware set, software writes DMA_CTICIF2 = 1 to clear. 0: No TC event 1: TC event occurred
3	TEIF1	R	0	Channel 1 Transmission Error (TE) flag. Hardware set, software writes DMA_CTEIF1 = 1 to clear. 0: No TE event 1: TE event occurred
2	HBTIF1	R	0	Channel 1 Half Block Transfer (HBT) completion flag. Hardware set, software writes DMA_CHBTIF1 = 1 to clear. 0: No HBT event 1: HBT event occurred
1	BTCIF1	R	0	Channel 1 Block Transfer Complete (BTC) flag. Hardware set, software writes DMA_CBTCIF1 = 1 to clear. 0: No BTC event 1: BTC event occurs
0	TCIF1	R	0	Channel 1 All Block Transfer Complete (TC) interrupt flag. Hardware set, software writes DMA_CTICIF1 = 1 to clear. 0: No TC event 1: TC event occurred

10.6.2. DMA Interrupt Flag Clear Register (DMA_IFCR)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	CTEI F7	CHBT IF7	CBTC IF7	CTCI F7	CTEI F6	CHBT IF6	CBTC IF6	CTCI F6	CTEI F5	CHBT IF5	CBTC IF5	CTCI F5
				W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTEI F4	CHBT IF4	CBTC IF4	CTCI F4	CTEI F3	CHBT IF3	CBTC IF3	CTCI F3	CTEI F2	CHBT IF2	CBTC IF2	CTCI F2	CTEI F1	CHBT IF1	CBTC IF1	CTCI F1
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27	CTEIF7	W	0	Transfer error flag clear for channel 7 0: No effect 1: Clear TEIF7
26	CHBTIF7	W	0	Channel 7 block half transfer flag cleared. 0: No effect 1: Clear HBTIF7
25	CBTCIF7	W	0	Channel 7 block transfer complete flag cleared. 0: No effect 1: Clear BTCIF7
24	CTCIF7	W	0	All block transmission completion flag of channel 7 is cleared. 0: No effect 1: Clear TCIF7
23	CTEIF6	W	0	Transfer error flag clear for channel 6 0: No effect 1: Clear TEIF6
22	CHBTIF6	W	0	Channel 6 block half transfer flag cleared. 0: No effect 1: Clear HBTIF6
21	CBTCIF6	W	0	Channel 6 block transmission completion flag is cleared. 0: No effect 1: Clear BTCIF6
20	CTCIF6	W	0	All block transmission completion flag of channel 6 is cleared. 0: No effect

				1: Clear TCIF6
19	CTEIF5	W	0	Transfer error flag clear for channel 5 0: No effect 1: Clear TEIF5
18	CHBTIF5	W	0	Channel 5 block half transfer flag cleared. 0: No effect 1: Clear HBTIF5
17	CBTCIF5	W	0	Channel 5 block transfer complete flag cleared. 0: No effect 1: Clear BTCIF5
16	CTCIF5	W	0	All block transmission completion flag of channel 5 is cleared. 0: No effect 1: Clear TCIF5
15	CTEIF4	W	0	Transfer error flag clear for channel 4 0: No effect 1: Clear TEIF4
14	CHBTIF4	W	0	Channel 4 block half transfer flag cleared. 0: No effect 1: Clear HBTIF4
13	CBTCIF4	W	0	Channel 4 block transfer complete flag cleared. 0: No effect 1: Clear BTCIF4
12	CTCIF4	W	0	All block transmission completion flag of channel 4 is cleared. 0: No effect 1: Clear TCIF4
11	CTEIF3	W	0	Transfer error flag clear for channel 3 0: No effect 1: Clear TEIF3
10	CHBTIF3	W	0	Channel 3 block half transfer flag cleared. 0: No effect 1: Clear HBTIF3
9	CBTCIF3	W	0	Channel 3 block transfer complete flag cleared. 0: No effect 1: Clear BTCIF3
8	CTCIF3	W	0	All block transmission completion flag of channel 3 is cleared. 0: No effect 1: Clear TCIF3
7	CTEIF2	W	0	Transfer error flag clear for channel 2 0: No effect 1: Clear TEIF2
6	CHBTIF2	W	0	Channel 2 block half transfer flag cleared. 0: No effect 1: Clear HBTIF2
5	CBTCIF2	W	0	Channel 2 block transfer complete flag cleared. 0: No effect 1: Clear BTCIF2
4	CTCIF2	W	0	All block transmission completion flag of channel 2 is cleared. 0: No effect 1: TCIF2 cleared
3	CTEIF1	W	0	Transfer error flag clear for channel 1 0: No effect 1: Clear TEIF1
2	CHBTIF1	W	0	Channel 1 block half transfer flag cleared. 0: No effect 1: Clear HBTIF1
1	CBTCIF1	W	0	Channel 1 block transfer complete flag cleared. 0: No effect 1: Clear BTCIF1
0	CTCIF1	W	0	All block transmission completion flag of channel 1 is cleared. 0: No effect 1: Clear TCIF1

10.6.3. DMA channel x configuration register (DMA_CCRx) (x=1~7)

Offset address: $0x08+(x-1)*0x14$ (x=1~7)

Reset value: 0x000x 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	M2MARB	BTCSUSP	MNORLD	PNORLD	MEM2MEM	PL[2:0]		
								RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIZE[1:0]		PSIZE[1:0]		MINC[1:0]		PINC[1:0]		TRANSIT	CIRC	DIR	TEIE	HBTIE	BTCE	TCE	EN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23	M2MARB	RW	0	Arbitration mode in channel x memory to memory mode (valid when mem2mem = 1) 0: The bus is not released after data is transmitted once; (Loop mode is not supported) 1: Release the bus after transmitting data once; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
22	BTCSUSP	RW	0	Enter BTC interrupt post-processing mode in channel x loop mode 0: Transmission is not suspended; 1: Suspend transmission; When this bit is configured to 1, after entering the BTC interrupt, the software clears the BTC interrupt and resumes transmission; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
21	MNORLD	RW	0	Memory address circular mode for channel x (valid when CIRC=1) 0: memory address reload enabled; 1: Memory address reload disabled; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
20	PNORLD	RW	0	Peripheral address circular mode of channel x (valid when CIRC=1) 0: Peripheral address reload enabled; 1: Peripheral address reloading is prohibited; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
19	MEM2MEM	RW	0	Channel x memory to memory mode. 0: Interrupt is disabled 1: memory to memory mode enabled; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
18:16	PL[2:0]	RW	x	Channel x priority configuration. 000: Channel priority is 0; 001: Channel priority is 1; 010: Channel priority is 2; 011: Channel priority is 3; 100: Channel priority is 4; 101: Channel priority is 5; 110: Channel priority is 6; Others: Reserved When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
15:14	MSIZE[1:0]	RW	0	Channel x memory data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;

13:12	PSIZE[1:0]	RW	0	Channel x Peripheral Data Width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
11:10	MINC[1:0]	RW	0	Channel x memory address delta mode. 00: No change; 01: Increment; 10: Decrement; 11: Reserved; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
9:8	PINC[1:0]	RW	0	Channel x peripheral address increment mode. 00: No change; 01: Increment; 10: Decrement; 11: Reserved; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
7	TRANST	RW	0	Data transmission mode of channel x (valid when mem2mem = 0) 0: Single transfer; 1: Burst transfer; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
6	CIRC	RW	0	the circular mode 0: Interrupt is disabled 1: Enabled When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
5	DIR	RW	0	Channel x Data Transmission Direction. 0: Read from peripheral; 1: Read from memory; When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;
4	TEIE	RW	0	Channel x Transmission Error Interrupt (TE) enabled. 0: Interrupt is disabled 1: TE interrupt enable;
3	HBTIE	RW	0	Channel x Half Block Transfer Interrupt (HBT) enabled. 0: Interrupt is disabled 1: HBT interrupt enable;
2	BTCIE	RW	0	Channel x Block Transfer Complete Interrupt (BTC) enabled. 0: Interrupt is disabled 1: BTC interrupt enable;
1	TCIE	RW	0	Channel x All block transmission completion interrupt (TC) enabled. 0: Interrupt is disabled 1: TC interrupt enable;
0	EN	RW	0	Channel x enables. 0: Interrupt is disabled 1: Channel x enabled; Cleared by hardware when all block transfers are completed or a transfer error occurs; The software is readable and writable; Note 1: In case of invalid configuration, such as NDT=0 but EN=1, hardware will clear EN to ensure subsequent transfers proceed normally; Note 2: When configuring DMA registers, follow the principle of configuring first and then controlling, i.e., set EN=1 last;

10.6.4. DMA Channel x Data Transfer Number Register (DMA_CNDTRx) (x=1~7)

Offset address: $0x0C+(x-1)*0x14$ (x=1~7)

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	NDT[15:0]	RW	0	Channel x Number of Data Transmission. The number of data transfer is 0 to 65535. This register is written only when the channel is not working (DMA_CCRx.EN = 0). This register becomes read-only after the channel is enabled, indicating the remaining number of transfers. This register value is decremented after each DMA transfer. After the data transfer is completed, the contents of the register either become 0, or when the channel is configured in ciucular mode, the contents of the register will be automatically reloaded to the previously configured value. If this field is zero, no transfer can be served whatever the channel status. When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;

10.6.5. DMA Channel x Peripheral Address Register (DMA_CPARx) (x=1~7)

Offset address: 0x10+(x-1)*0x14 (x=1~7)

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	PA[31:0]	RW	0	Channel x peripheral address. The base address of the channel x peripheral data register as the source or destination of the data transfer. When PSIZE = 2 'b01, the PA [0] bit is ignored. Access is automatically aligned to a half-word address. When PSIZE = 2 'b10, the PA [1: 0] bit is ignored. Access is automatically aligned with the word address. When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;

10.6.6. DMA channel x memory address register (DMA_CMARx) (x=1~7)

Offset address: 0x14+(x-1)*0x14 (x=1~7)

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	MA[31:0]	RW	0	Channel x memory address. Channel x memory address, as the source or destination of data transfer. When MSIZE = 2 'b01, the MA [0] bit is ignored. Access is automatically aligned to a half-word address. When MSIZE = 2 'b10, the MA [1: 0] bit is ignored. Access is automatically aligned with the word address.

				When EN = 0, the software is readable and writable; When EN = 1, the software is only readable;

10.6.7. DMA channel x circular transfer configuration register (DMA_CCCFGRx) (x=1~7)

Offset address: 0x18+(x-1)*0x14 (x=1~7)

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	NDTL[15:0]	RW	0	Number of last block data transfers for channel x in finite circular mode. The number of data transmissions is 0 ~ 65535. This register is written only when the channel is not working (DMA_CCRx.EN = 0). This register is read-only when the channel is enabled, indicating the number of last remaining transfers in the cyclic mode. This register value decrements with each transfer during the last block transfer in circular mode. When EN=0, software can read and write; When EN=1, software can only read;
15:8	Reserved	-	-	Reserved
7:0	NBT[7:0]	RW	0	Number of block circular transfers for channel x. In circular mode, when this register's initial value is 0, the block circular transfer count is infinite; when greater than 0, the count equals this register's value (1~255). This register is written only when the channel is not working (DMA_CCRx.EN = 0). This register becomes read-only after channel enable, indicating remaining transfer blocks. This register value decrements after each block transfer completes. When EN = 0, the software is readable and writable; When EN = 1, the software is only readable; Note: Number of data transmitted in limited cycle = NDT * NBT + NDTL

11. Interrupts and events

11.1. Nested Vectored Interrupt Controller (NVIC)

11.1.1. NVIC main features

- Support 32 maskable external interrupts (not including the sixteen CPU interrupt lines)
- 4 programmable priority levels (2 bits of interrupt priority are used)
- Low-latency exception and interrupt handling
- Power management control
- Implementation of system control registers

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

11.1.2. SysTick calibration value register

The SysTick calibration value is set to 9000, with the SysTick clock configured to 9 MHz (max fHCLK/8), generating a 1ms time base.

11.1.3. Interrupt and exception vectors

Table 11-1 Terminals and Exception Vector Tables

Vector numbering	Prior-ity	Type of priority	Abbreviations	Description	Address
-	-	-	-	Reserved	0x0000_0000
-	-3	fixed	Reset	Reset	0x0000_0004
-	-2	fixed	NMI_Handler	NMI. 1) RCC Clock Security System failure (CSS) 2) SRAM parity error 3) Flash ECC error exceeding 1 bit	0x0000_0008
-	-1	fixed	HardFualt_Handler	All types of errors	0x0000_000C
-	-	-	-	Reserved	0x0000_0010~0x0000_0028
-	3	settable	SVCall	System service called via SWI instruction	0x0000_002C
-	-	-	-	Reserved	0x0000_0030~0x0000_0034
-	5	settable	PendSV	Pendable system service request	0x0000_0038
-	6	settable	SysTick	System tick timer	0x0000_003C
0	7	settable	WWDG	Window watchdog interrupt	0x0000_0040
1	8	settable	PVD	PVD interrupt via EXTI Line16	0x0000_0044
2	9	settable	RTC_TAMP	RTC and TAMP interrupts	0x0000_0048
3	10	settable	Flash	Flash global interrupt	0x0000_004C
4	11	settable	RCC	RCC global interrupt	0x0000_0050
5	12	settable	EXTI0_1	EXTI Line 0 and 1 interrupts	0x0000_0054
6	13	settable	EXTI2_3	EXTI Line 0 and 1 interrupts	0x0000_0058
7	14	settable	EXTI4_15	EXTI Line 4 to 15 interrupts	0x0000_005C
8	15	settable	LCD	LCD global interrupt	0x0000_0060
9	16	settable	DMA_Channel1	DMA channel 1 interrupt	0x0000_0064

10	17	settable	DMA_Channel 2_3	DMA channel 2 and 3 interrupt	0x0000_0068
11	18	settable	DMA_Channel4_7	DMA channel 4 to 7 interrupt	0x0000_006C
12	19	settable	ADC_COMP	ADC and COMP interrupt (COMP combined with EXTI Line 17 and 18)	0x0000_0070
13	20	settable	TIM1_BRK_UP_TRG_COM	TIM1 brake, update, trigger, and commutation interrupt	0x0000_0074
14	21	settable	TIM1_CC	TIM1 capture compare interrupt	0x0000_0078
15	22	settable	TIM2	TIM2 global interrupt	0x0000_007C
16	23	settable	TIM3	TIM3 global interrupt	0x0000_0080
17	24	settable	TIM6_DAC_LPTIM1	LPTIM1, TIM6, and DAC interrupt	0x0000_0084
18	25	settable	TIM7_LPTIM2	LPTIM2 and TIM7 interrupt	0x0000_0088
19	26	settable	PWM	PWM global interrupt	0x0000_008C
20	27	settable	TIM15	TIM15 global interrupt	0x0000_0090
21	28	settable	TIM16	TIM16 global interrupt	0x0000_0094
22	29	settable	TIM17	TIM17 global interrupt	0x0000_0098
23	30	settable	I2C1	I2C1 global interrupt	0x0000_009C
24	31	settable	I2C2	I2C2 global interrupt	0x0000_00A0
25	32	settable	SPI1	SPI1 global interrupt	0x0000_00A4
26	33	settable	SPI2	SPI2 global interrupt	0x0000_00A8
27	34	settable	USART1	USART1 global interrupt	0x0000_00AC
28	35	settable	USART2	USART2 global interrupt	0x0000_00B0
29	36	settable	UART1_LPUART1	UART1 & LPUART1 global interrupt	0x0000_00B4
30	37	settable	UART2_LPUART2	UART2 & LPUART2 global interrupt	0x0000_00B8
31	38	settable	TK	TK global interrupt	0x0000_00BC

1. Gray rows correspond to Cortex®-M0+ interrupts.

11.2. Extended interrupt and event controller (EXTI)

The extended interrupt and event controller (EXTI) manages the CPU and system wake-up through configurable and direct event inputs (Lines). It outputs following signals:

- Interrupt request, generating CPU interrupt input (IRQ)
- Event request, generating CPU event input (RXEV)
- The wake-up request is sent to the power consumption management control module

EXTI wakeup request allows the system to wake from Stop mode. Interrupt and event requests can also be used in Run/Low-power Run modes.

EXTI can manage up to 28 configurable/direct event lines (19 configurable event lines and 9 direct event lines).

11.2.1. EXTI main features

- The system can be woken by GPIO and specified module inputs (PVD/COMP/RTC/TAMP/LPTIM/LPUART/I2C/TK).
- Configurable events (from I/O or peripherals without interrupt pending status bits, generating pulse peripherals)
 - Selectable active trigger edge
 - Interrupt pending status bits
 - Individual interrupt and event generation mask

- SW trigger possibility
- Direct events (peripherals with associated flags and interrupt pending status bits)
 - Fixed rising edge active trigger
 - No interrupt pending status bit in the EXTI
 - Individual interrupt and event generation mask
 - No SW trigger possibility
- I/O port selector

11.2.2. EXTI block diagram

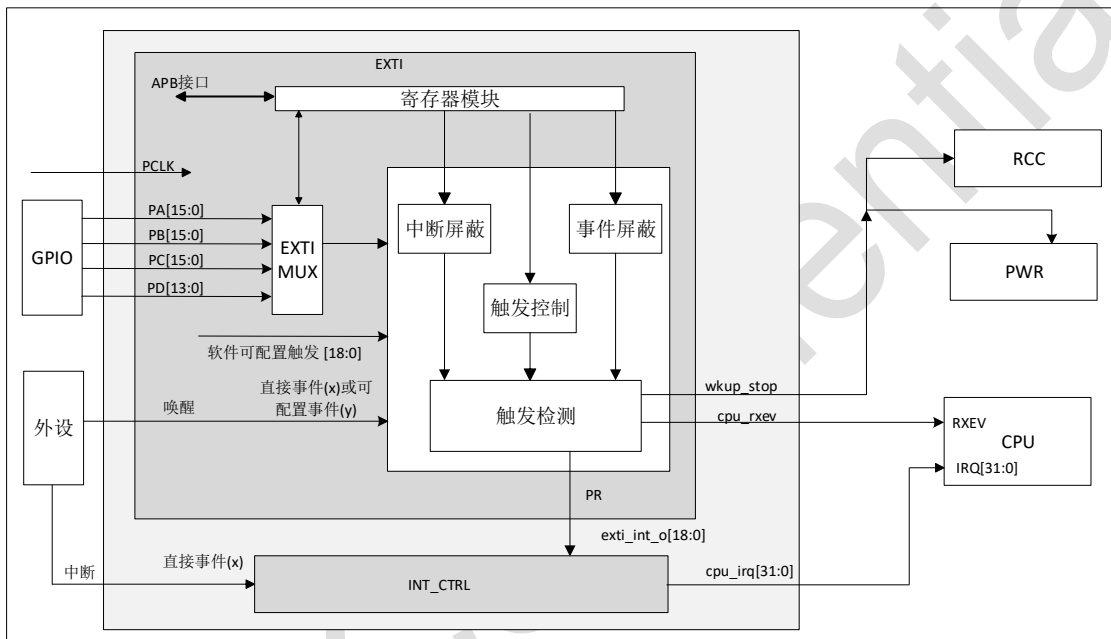


Figure 11-1 EXTI block diagram

Table 11-2 Signal Overview

Signal name	I/O	Description
APB interface	I/O	EXTI register bus interface.
PCLK	I	EXTI system clock
Configurable event (y)	I	Asynchronous wakeup events from peripherals, where no related interrupts or flags exist in the peripheral.
Direct event (x)	I	Synchronous and asynchronous wakeup events from peripherals, with related interrupts and flags in the peripheral.
PA/PB/PC[15:0] PD[13:0]	I	GPIO port
wkup_stop	O	Output to PWR and RCC modules for wakeup handling.
cpu_rxev	O	Output to the CPU's RXEV interface signal.
exti_int_o[18:0]	O	System clock domain signal, configurable for Line-type interrupts.
cpu_irq[31:0]	O	Output to the CPU's IRQ interface signal.

11.2.3. EXTI function description

Peripherals capable of generating wakeup or interrupt event signals in stop mode, connected to the EXTI module.

- Wakeup signals that generate a pulse or have no interrupt status bit inside the peripheral, connected to the configurable Line-type of the EXTI module. The EXTI module generates an interrupt pending bit (which needs to be cleared), and this EXTI interrupt serves as a CPU interrupt signal.
- Interrupt and wakeup signals from peripherals with associated status bits (which need to be cleared in the peripheral), connected to the direct Line-type of the EXTI module.
- All GPIO port inputs to the EXTI MUX module, allowing selected signals to serve as system wakeup signals through configuration.

11.2.4. EXTI configurable event input wakeup

Supports software triggering; software can trigger wakeup functionality by configuring the EXTI_SWIER register.

Configurable registers for rising-edge, falling-edge, or dual-edge triggering; hardware detects configurable event input signals based on configuration and generates corresponding wakeup events or interrupt signals.

Includes interrupt mask register (EXTI_IMR) and event mask register (EXTI_EMR). Events generated for the CPU when enabled. All events for the CPU are ORed and output as a single event input signal RXEV to the CPU.

Configurable type events have a unique interrupt pending request register, shared with the CPU. The pending register is set only when the CPU interrupt mask register (EXTI_IMR) is configured as unmasked. Each configurable type event corresponds to a CPU external interrupt signal (some may be multiplexed to the same CPU external interrupt signal). Interrupts for configurable type events require acknowledgment by the CPU through the EXTI_PR register (write 1 to clear).

Note: When the interrupt pending register (EXTI_PR) remains active (not cleared), the system cannot enter low-power mode.

11.2.5. EXTI direct event input wake-up

Direct type events do not generate interrupts in the EXTI module but produce event signals to wake up the system and CPU subsystem. When the CPU handles the interrupt generated by this type of trigger event, it needs to clear the interrupt status bit of the peripheral module.

11.2.6. EXTI multiplexing

GPIO is connected to the 16 external interrupt/event lines in the following manner:

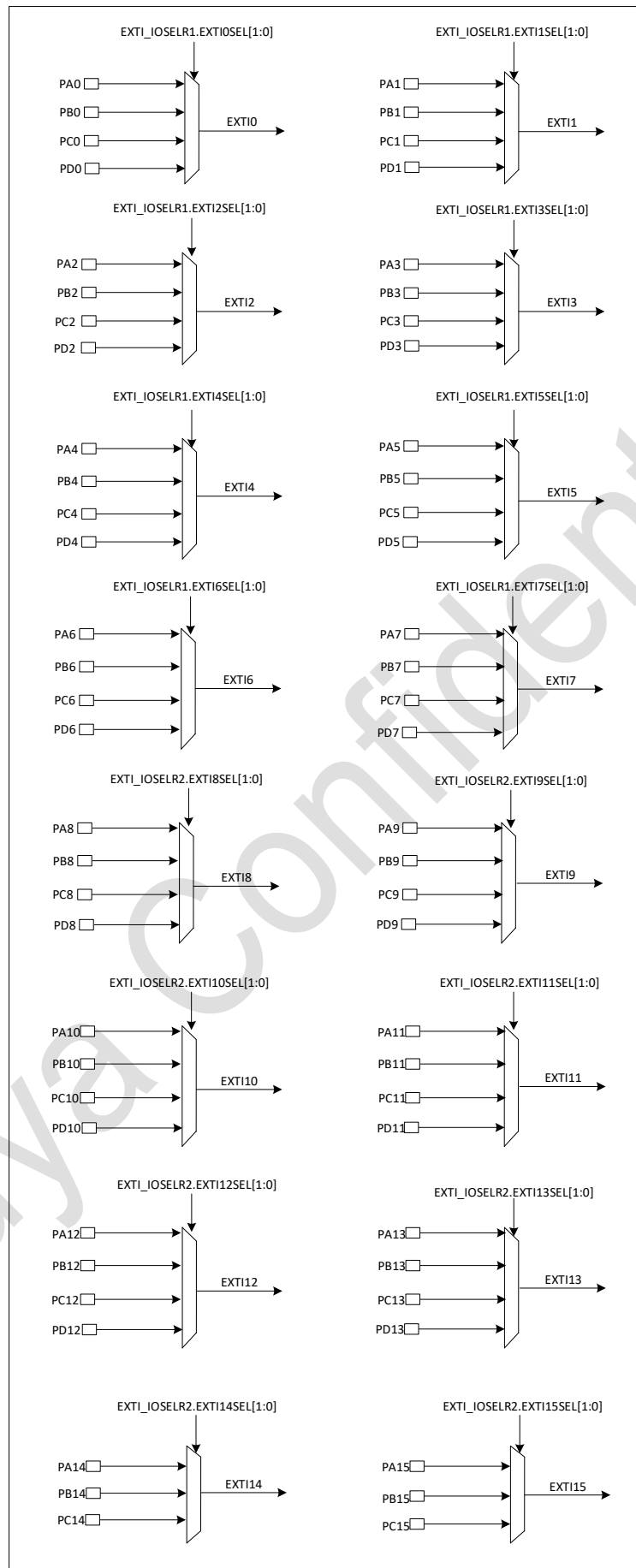


Figure 11-2 EXTI GPIO Multiplexing

11.2.7. EXTI Line

All line connections are shown in the table below:

Table 11-3 EXTI lines connections

EXTI Line	Source	Type
0-15	GPIO	Configuration
16	PVD output	Configuration
17	COMP 1 output	Configuration
18	COMP2 output	Configuration
19	-	-
20	RTC	Direct
21	TAMP	Direct
22	I2C1	Direct
23	I2C2	Direct
24	LPUART1	Direct
25	LPUART2	Direct
26	TK	Direct
27	LPTIM1	Direct
28	LPTIM2	Direct
29	-	-

11.3. EXTI register

11.3.1. EXTI Rising Edge Trigger Selection Register (EXTI_RTSR)

Address offset: 0x00

Reset value: 0x0000 0000

Contains only register control bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RT18	RT17	RT16
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	RT18	RW	0	Configuration type EXTI line18 rising edge trigger configuration. 0: Disabled 1: Enabled
17	RT17	RW	0	Configuration type EXTI line17 rising edge trigger configuration. 0: Disabled 1: Enabled
16	RT16	RW	0	Configuration type EXTI line16 rising edge trigger configuration. 0: Disabled 1: Enabled
15	RT15	RW	0	Configuration type EXTI line15 rising edge trigger configuration. 0: Disabled 1: Enabled
14	RT14	RW	0	Configuration type EXTI line14 rising edge trigger configuration. 0: Disabled 1: Enabled
13	RT13	RW	0	Configuration type EXTI line13 rising edge trigger configuration. 0: Disabled 1: Enabled
12	RT12	RW	0	Configuration type EXTI line12 rising edge trigger configuration. 0: Disabled 1: Enabled

11	RT11	RW	0	Configuration type EXTI line11 rising edge trigger configuration. 0: Disabled 1: Enabled
10	RT10	RW	0	Configuration type EXTI line10 rising edge trigger configuration. 0: Disabled 1: Enabled
9	RT9	RW	0	Configuration type EXTI line9 rising edge trigger configuration. 0: Disabled 1: Enabled
8	RT8	RW	0	Configuration type EXTI line8 rising edge trigger configuration. 0: Disabled 1: Enabled
7	RT7	RW	0	Configuration type EXTI line7 rising edge trigger configuration. 0: Disabled 1: Enabled
6	RT6	RW	0	Configuration type EXTI line6 rising edge trigger configuration. 0: Disabled 1: Enabled
5	RT5	RW	0	Configuration type EXTI line5 rising edge trigger configuration. 0: Disabled 1: Enabled
4	RT4	RW	0	Configuration type EXTI line4 rising edge trigger configuration. 0: Disabled 1: Enabled
3	RT3	RW	0	Configuration type EXTI line3 rising edge trigger configuration. 0: Disabled 1: Enabled
2	RT2	RW	0	Configuration type EXTI line2 rising edge trigger configuration. 0: Disabled 1: Enabled
1	RT1	RW	0	Configuration type EXTI line1 rising edge trigger configuration. 0: Disabled 1: Enabled
0	RT0	RW	0	Configuration type EXTI line0 rising edge trigger configuration. 0: Disabled 1: Enabled

Configurable EXTI Lines are edge-triggered, and no glitches can be generated on these Lines. If a rising edge occurs on the interrupt line during the write operation to the EXTI_RTSR register, the corresponding pending bit is not set.

Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.

11.3.2. EXTI Falling Trigger Selection Register (EXTI_FTSR)

Address offset: 0x04

Reset value: 0x0000 0000

Contains only register control bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FT18	FT17	FT16
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	FT18	RW	0	Configurable EXTI line18 falling edge trigger configuration. 0: Disabled 1: Enabled
17	FT17	RW	0	Configurable EXTI line17 falling edge trigger configuration. 0: Disabled 1: Enabled
16	FT16	RW	0	Configurable EXTI line16 falling edge trigger configuration. 0: Disabled 1: Enabled
15	FT15	RW	0	Configurable EXTI line15 falling edge trigger configuration. 0: Disabled 1: Enabled
14	FT14	RW	0	Configurable EXTI line14 falling edge trigger configuration. 0: Disabled 1: Enabled
13	FT13	RW	0	Configurable EXTI line13 falling edge trigger configuration. 0: Disabled 1: Enabled
12	FT12	RW	0	Configurable EXTI line12 falling edge trigger configuration. 0: Disabled 1: Enabled
11	FT11	RW	0	Configurable EXTI line11 falling edge trigger configuration. 0: Disabled 1: Enabled
10	FT10	RW	0	Configurable EXTI line10 falling edge trigger configuration. 0: Disabled 1: Enabled
9	FT9	RW	0	Configurable EXTI line9 falling edge trigger configuration. 0: Disabled 1: Enabled
8	FT8	RW	0	Configurable EXTI line8 falling edge trigger configuration. 0: Disabled 1: Enabled
7	FT7	RW	0	Configurable EXTI line7 falling edge trigger configuration. 0: Disabled 1: Enabled
6	FT6	RW	0	Configurable EXTI line6 falling edge trigger configuration. 0: Disabled 1: Enabled
5	FT5	RW	0	Configurable EXTI line5 falling edge trigger configuration. 0: Disabled 1: Enabled
4	FT4	RW	0	Configurable EXTI line4 falling edge trigger configuration. 0: Disabled 1: Enabled
3	FT3	RW	0	Configurable EXTI line3 falling edge trigger configuration. 0: Disabled 1: Enabled
2	FT2	RW	0	Configurable EXTI line2 falling edge trigger configuration. 0: Disabled 1: Enabled
1	FT1	RW	0	Configurable EXTI line1 falling edge trigger configuration. 0: Disabled 1: Enabled
0	FT0	RW	0	Configurable EXTI line0 falling edge trigger configuration. 0: Disabled 1: Enabled

Configurable EXTI Lines are edge-triggered, and no glitches can be generated on these Lines. If a falling edge occurs on the interrupt line during the write operation to the EXTI_FTSR register, the corresponding pending bit is not set.

Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.

11.3.3. EXTI Software Interrupt Event Register (EXTI_SWIER)

Address offset: 0x08

Reset value: 0x0000 0000

Contains only register control bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWIE 18	SWIE 17	SWIE 16
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIE 15	SWIE 14	SWIE 13	SWIE 12	SWIE 11	SWIE 10	SWI E9	SWI E8	SWI E7	SWI E6	SWI E5	SWI E4	SWI E3	SWIE 2	SWIE 1	SWIE 0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	SWIE18	RW	0	Configurable EXTI line18 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
17	SWIE17	RW	0	Configurable EXTI line17 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
16	SWIE16	RW	0	Configurable EXTI line16 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
15	SWIE15	RW	0	Configurable EXTI line15 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
14	SWIE14	RW	0	Configurable EXTI line14 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
13	SWIE13	RW	0	Configurable EXTI line13 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).

12	SWIE12	RW	0	Configurable EXTI line12 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
11	SWIE11	RW	0	Configurable EXTI line11 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
10	SWIE10	RW	0	Configurable type EXTI line10 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
9	SWIE9	RW	0	Configurable type EXTI line9 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
8	SWIE8	RW	0	Configurable type EXTI line8 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
7	SWIE7	RW	0	Configurable type EXTI line7 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
6	SWIE6	RW	0	Configurable type EXTI line6 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
5	SWIE5	RW	0	Configurable type EXTI line5 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
4	SWIE4	RW	0	Configurable type EXTI line4 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt

				This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
3	SWIE3	RW	0	Configurable type EXTI line3 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
2	SWIE2	RW	0	Configurable type EXTI line2 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
1	SWIE1	RW	0	Configurable type EXTI line1 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).
0	SWIE0	RW	0	Configurable type EXTI line0 software rising edge trigger configuration. 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared when the corresponding bit of the EXTI_PR register is written to 1 by software. Read returns 0 (after clearing) or the configured value (before clearing).

11.3.4. EXTI pending register (EXTI_PR)

Address offset: 0x0C

Reset value: 0x0000 0000

Contains only register control bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR18	PR17	PR16
													RC_W1	RC_W1	RC_W1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	PR18	RC_W1	0	Configurable type EXTI line18 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
17	PR17	RC_W1	0	Configurable type EXTI line17 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
16	PR16	RC_W1	0	Configurable type EXTI line16 event pending flag. When software or hardware generates rising/falling

				edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
15	PR15	RC_W1	0	Configurable type EXTI line15 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
14	PR14	RC_W1	0	Configurable type EXTI line14 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
13	PR13	RC_W1	0	Configurable type EXTI line13 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
12	PR12	RC_W1	0	Configurable type EXTI line12 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
11	PR11	RC_W1	0	Configurable type EXTI line11 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
10	PR10	RC_W1	0	Configurable type EXTI line10 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
9	PR9	RC_W1	0	Configurable type EXTI line9 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
8	PR8	RC_W1	0	Configurable type EXTI line8 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
7	PR7	RC_W1	0	Configurable type EXTI line7 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;

6	PR6	RC_W1	0	Configurable type EXTI line6 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
5	PR5	RC_W1	0	Configurable type EXTI line5 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
4	PR4	RC_W1	0	Configure the EXTI line4 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
3	PR3	RC_W1	0	Configure the EXTI line3 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
2	PR2	RC_W1	0	Configure the EXTI line2 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
1	PR1	RC_W1	0	Configure the EXTI line1 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
0	PR0	RC_W1	0	Configure the EXTI line0 event pending flag. When software or hardware generates rising/falling edge trigger events, this bit is set. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;

11.3.5. EXTI external interrupt selection register 1 (EXTI_IOSCLR1)

Address offset: 0x10

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	EXTI7SEL [1:0]		Res.	Res.	EXTI6SEL [1:0]		Res.	Res.	EXTI5SEL [1:0]		Res.	Res.	EXTI4SEL [1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	EXTI3SEL [1:0]		Res.	Res.	EXTI2SEL [1:0]		Res.	Res.	EXTI1SEL [1:0]		Res.	Res.	EXTI0SEL [1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:28	EXTI7SEL [1:0]	RW	0	GPIO port selection for EXTI7. 2'b00: PA[7] pin 2'b01: PB[7] pin

				2'b10: PC[7] pin 2'b11: PD[7] pin
27:26	Reserved	-	-	Reserved
25:24	EXTI6SEL [1:0]	RW	0	GPIO port selection for EXTI6. 2'b00: PA[6] pin 2'b01: PB[6] pin 2'b10: PC[6] pin 2'b11: PD[6] pin
23:22	Reserved	-	-	Reserved
21:20	EXTI5SEL [1:0]	RW	0	GPIO port selection for EXTI5. 2'b00: PA[5] pin 2'b01: PB[5] pin 2'b10: PC[5] pin 2'b11: PD[5] pin
19:18	Reserved	-	-	Reserved
17:16	EXTI4SEL [1:0]	RW	0	GPIO port selection for EXTI4. 2'b00: PA[4] pin 2'b01: PB[4] pin 2'b10: PC[4] pin 2'b11: PD[4] pin
15:14	Reserved	-	-	Reserved
13:12	EXTI3SEL [1:0]	RW	0	GPIO port selection for EXTI3. 2'b00: PA[3] pin 2'b01: PB[3] pin 2'b10: PC[3] pin 2'b11: PD[3] pin
11:10	Reserved	-	-	Reserved
9:8	EXTI2SEL [1:0]	RW	0	GPIO port selection for EXTI2. 2'b00: PA[2] pin 2'b01: PB[2] pin 2'b10: PC[2] pin 2'b11: PD[2] pin
7:6	Reserved	-	-	Reserved
5:4	EXTI1SEL [1:0]	RW	0	GPIO port selection for EXTI1. 2'b00: PA[1] pin 2'b01: PB[1] pin 2'b10: PC[1] pin 2'b11: PD[1] pin
3:2	Reserved	-	-	Reserved
1:0	EXTI0SEL [1:0]	RW	0	GPIO port selection for EXTI0. 2'b00: PA[0] pin 2'b01: PB[0] pin 2'b10: PC[0] pin 2'b11: PD[0] pin

11.3.6. EXTI external interrupt selection register 2 (EXTI_IOSCLR2)

Address offset: 0x14

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	EXTI15SEL [1:0]		Res.	Res.	EXTI14SEL [1:0]		Res.	Res.	EXTI13SEL [1:0]		Res.	Res.	EXTI12SEL [1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	EXTI11SEL [1:0]		Res.	Res.	EXTI10SEL [1:0]		Res.	Res.	EXTI9SEL [1:0]		Res.	Res.	EXTI8SEL [1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:28	EXTI15SEL [1:0]	RW	0	GPIO port selection for EXTI15. 2'b00: PA[15] pin 2'b01: PB[15] pin 2'b10: PC[15] pin 2'b11: Reserved
27:26	Reserved	-	-	Reserved
25:24	EXTI14SEL [1:0]	RW	0	GPIO port selection for EXTI14.

				2'b00: PA[14] pin 2'b01: PB[14] pin 2'b10: PC[14] pin 2'b11: Reserved
23:22	Reserved	-	-	Reserved
21:20	EXTI13SEL [1:0]	RW	0	GPIO port selection for EXTI13. 2'b00: PA[13] pin 2'b01: PB[13] pin 2'b10: PC[13] pin 2'b11: PD[13] pin
19:18	Reserved	-	-	Reserved
17:16	EXTI12SEL [1:0]	RW	0	GPIO port selection for EXTI12. 2'b00: PA[12] pin 2'b01: PB[12] pin 2'b10: PC[12] pin 2'b11: PD[12] pin
15:14	Reserved	-	-	Reserved
13:12	EXTI11SEL [1:0]	RW	0	GPIO port selection for EXTI11. 2'b00: PA[11] pin 2'b01: PB[11] pin 2'b10: PC[11] pin 2'b11: PD[11] pin
11:10	Reserved	-	-	Reserved
9:8	EXTI10SEL [1:0]	RW	0	GPIO port selection for EXTI10. 2'b00: PA[10] pin 2'b01: PB[10] pin 2'b10: PC[10] pin 2'b11: PD[10] pin
7:6	Reserved	-	-	Reserved
5:4	EXTI9SEL [1:0]	RW	0	EXTI9 GPIO port selection. 2'b00: PA[9] pin 2'b01: PB[9] pin 2'b10: PC[9] pin 2'b11: PD[9] pin
3:2	Reserved	-	-	Reserved
1:0	EXTI8SEL [1:0]	RW	0	EXTI8 GPIO port selection. 2'b00: PA[8] pin 2'b01: PB[8] pin 2'b10: PC[8] pin 2'b11: PD[8] pin

11.3.7. EXTI Interrupt Mask Register (EXTI_IMR)

Address offset: 0x18

Reset value: 0x1FF0 0000

Note: The interrupt mask bit for direct-type lines defaults to 1, meaning the line is enabled; for configuration-type lines, the mask bit defaults to 0, meaning the line is masked.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	IM28	IM27	IM26	IM25	IM24	IM23	IM22	IM21	IM20	Res.	IM18	IM17	IM16
			RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	Reserved
28	IM28	RW	1	EXTI line28 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
27	IM27	RW	1	EXTI line27 CPU wakeup interrupt mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
26	IM26	RW	1	EXTI line26 CPU wakeup interrupt mask control. 0: Interrupt request is masked

				1: Interrupt request is not masked
25	IM25	RW	1	EXTI line25 CPU wakeup interrupt mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
24	IM24	RW	1	EXTI line24 CPU wakeup interrupt mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
23	IM23	RW	1	EXTI line23 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
22	IM22	RW	1	EXTI line22 CPU wakeup interrupt mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
21	IM21	RW	1	EXTI line21 CPU wakeup interrupt mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
20	IM20	RW	1	CPU wakeup with interrupt mask on EXTI line 20 0: Interrupt request is masked 1: Interrupt request is not masked
19	Reserved	-	-	Reserved
18	IM18	RW	0	Interrupt mask on line 18. 0: Interrupt request is masked 1: Interrupt request is not masked
17	IM17	RW	0	Interrupt mask on line 17. 0: Interrupt request is masked 1: Interrupt request is not masked
16	IM16	RW	0	EXTI line16 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
15	IM15	RW	0	EXTI line15 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
14	IM14	RW	0	EXTI line14 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
13	IM13	RW	0	EXTI line13 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
12	IM12	RW	0	EXTI line12 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
11	IM11	RW	0	EXTI line11 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
10	IM10	RW	0	EXTI line10 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
9	IM9	RW	0	EXTI line9 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
8	IM8	RW	0	EXTI line8 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
7	IM7	RW	0	Interrupt mask on line 7. 0: Interrupt request is masked 1: Interrupt request is not masked
6	IM6	RW	0	Interrupt mask on line 6. 0: Interrupt request is masked 1: Interrupt request is not masked
5	IM5	RW	0	Interrupt mask on line 5. 0: Interrupt request is masked

				1: Interrupt request is not masked
4	IM4	RW	0	Interrupt mask on line 4. 0: Interrupt request is masked 1: Interrupt request is not masked
3	IM3	RW	0	Interrupt mask on line 3. 0: Interrupt request is masked 1: Interrupt request is not masked
2	IM2	RW	0	Interrupt mask on line 2. 0: Interrupt request is masked 1: Interrupt request is not masked
1	IM1	RW	0	Interrupt mask on line 1. 0: Interrupt request is masked 1: Interrupt request is not masked
0	IM0	RW	0	Interrupt mask on line 0. 0: Interrupt request is masked 1: Interrupt request is not masked

11.3.8. EXTI event mask register (EXTI_EMR)

Address offset: 0x1C

Reset value: 0x1FF0 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	EM2 8	EM2 7	EM2 6	EM2 5	EM2 4	EM2 3	EM2 2	EM2 1	EM2 0	Res.	EM1 8	EM1 7	EM1 6
			RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM1 5	EM1 4	EM13	EM1 2	EM1 1	EM1 0	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	Reserved
28	EM28	RW	1	EXTI line28 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
27	EM27	RW	1	EXTI line27 CPU wakeup event mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
26	EM26	RW	1	EXTI line26 CPU wakeup event mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
25	EM25	RW	1	EXTI line25 CPU wakeup event mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
24	EM24	RW	1	EXTI line24 serves as event-triggered CPU wakeup mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
23	EM23	RW	1	EXTI line23 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
22	EM22	RW	1	EXTI line22 serves as event-triggered CPU wakeup mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
21	EM21	RW	1	EXTI line21 serves as event-triggered CPU wakeup mask control. 0: Interrupt request is masked 1: Interrupt request is not masked
20	EM20	RW	1	CPU wakeup with event mask on EXTI line 20 0: Interrupt request is masked 1: Interrupt request is not masked
19	Reserved	-	-	Reserved
18	EM18	RW	0	Interrupt mask on line 18. 0: Interrupt request is masked 1: Interrupt request is not masked
17	EM17	RW	0	Interrupt mask on line 17.

				0: Interrupt request is masked 1: Interrupt request is not masked
16	EM16	RW	0	EXTI line16 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
15	EM15	RW	0	EXTI line15 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
14	EM14	RW	0	EXTI line14 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
13	EM13	RW	0	EXTI line13 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
12	EM12	RW	0	EXTI line12 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
11	EM11	RW	0	EXTI line11 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
10	EM10	RW	0	EXTI line10 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
9	EM9	RW	0	EXTI line9 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
8	EM8	RW	0	EXTI line8 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
7	EM7	RW	0	Interrupt mask on line 7. 0: Interrupt request is masked 1: Interrupt request is not masked
6	EM6	RW	0	Interrupt mask on line 6. 0: Interrupt request is masked 1: Interrupt request is not masked
5	EM5	RW	0	Interrupt mask on line 5. 0: Interrupt request is masked 1: Interrupt request is not masked
4	EM4	RW	0	Interrupt mask on line 4. 0: Interrupt request is masked 1: Interrupt request is not masked
3	EM3	RW	0	Interrupt mask on line 3. 0: Interrupt request is masked 1: Interrupt request is not masked
2	EM2	RW	0	Interrupt mask on line 2. 0: Interrupt request is masked 1: Interrupt request is not masked
1	EM1	RW	0	Interrupt mask on line 1. 0: Interrupt request is masked 1: Interrupt request is not masked
0	EM0	RW	0	Interrupt mask on line 0. 0: Interrupt request is masked 1: Interrupt request is not masked

12. Cyclic redundancy check calculation unit (CRC)

12.1. Introduction to CRC

The CRC unit can calculate CRC codes from 8-bit, 16-bit, or 32-bit data using custom generator polynomials of 7-bit, 8-bit, 16-bit, or 32-bit width.

CRC-based techniques are used to verify data transmission or storage integrity. Within the field of functional safety standards, CRC-based technologies provide a way to verify the integrity of flash memory.

12.2. CRC main features

- Default polynomial: CRC-32 (Ethernet) $0x4C11DB7$

$$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$$
- Supports programmable polynomials, and the polynomial bit width is also programmable (supports 7, 8, 16, 32-bit polynomials)
- Supports 8-bit, 16-bit, and 32-bit data input
- Programmable CRC initial value
- Single I/O 32-bit data and result output share one register
- General-purpose 32-bit register (can be used as temporary storage)
- Calculation time: 4 AHB clocks for 32-bit data; 2 AHB clock cycles for 16-bit data; 1 AHB clock cycle (HCLK) for 8-bit data
- Support input data inversion, output data inversion
- The CRC_DR register can be accessed by AHB bus with 8-bit right-aligned, 16-bit right-aligned, or 32-bit width. Other registers can only be accessed by AHB bus with 32-bit width.

12.3. CRC functional description

12.3.1. CRC block diagram

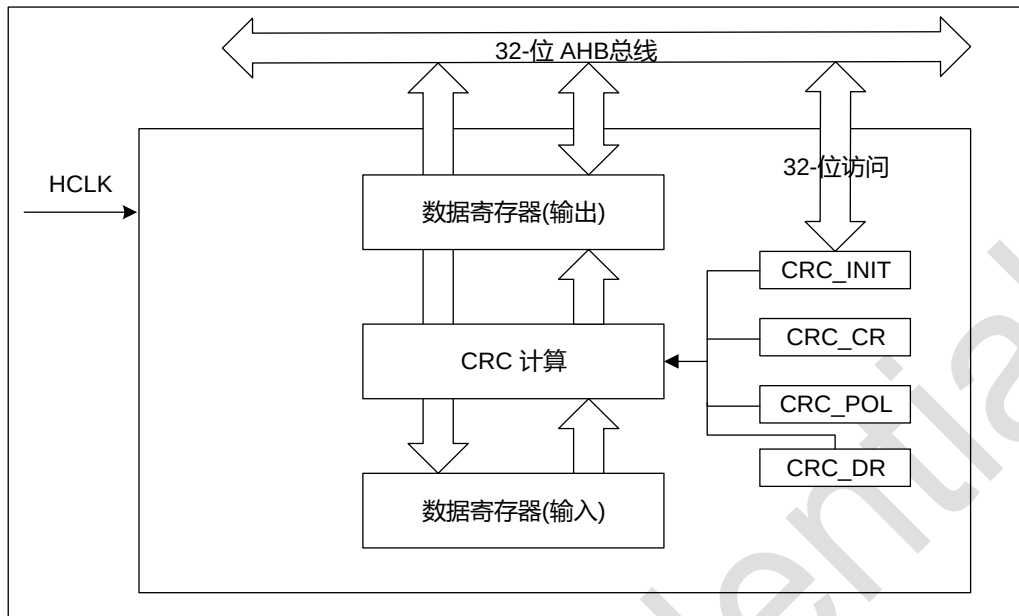


Figure 12-1 CRC block diagram

12.3.2. CRC functional description

The CRC calculation unit has a single 32-bit data register (CRC_DR).

- When writing to this register, it serves as an input register, allowing you to input new data for CRC calculation.
- When reading from this register, it returns the result of the previous CRC calculation.

Each write to the data register combines the previous CRC result with the new calculation (performing CRC on the full 32-bit word or byte-by-byte), depending on the written data format. The CRC_DR register can be accessed by words (32 bits), right-aligned half-words (16 bits), right-aligned bytes (8 bits), while other CRC registers only allow 32 bits.

For data of different widths (determined by HSIZE), the CRC module requires varying computation times as follows:

- Computing 32-bit data requires 4 AHB clock cycles
- Computing 16-bit data requires 2 AHB clock cycles
- It takes 1 AHB clock cycle to calculate 8-bit data

The input cache allows the user to write the second data immediately after the first data is written, without waiting for the previous data to be calculated.

Users can dynamically adjust the data size to reduce the number of accesses to this module when calculating a specific number of bytes of data. For example, to calculate a 5-byte data, one word of data can be written to this module, and then one byte of data can be written to obtain the calculation result.

This module supports input data reversal to accommodate different endian schemes. According to the REV_IN[1:0] bits in the CRC_DR register, input data reversal can be performed in units of 8-bit, 16-bit, or 32-bit.

For example, for input data 0x1A2B3C4D that requires a CRC operation:

- The data after inversion in bytes (8 bits) is 0x58D43CB2
- The data after inversion in units of half words (16 bits) is 0xD458B23C
- The data after inversion in words (32 bits) is 0xB23CD458

The REV_OUT bit in the CRC_CR register can invert the output data. This operation is done bitwise. For example: output data is 0x11223344, after reversal it becomes 0x22CC4488.

Setting the RESET bit in the CRC_CR register loads the CRC calculator's initial value with a programmable value (default is 0xFFFFFFFF).

The CRC initial value can be programmed to the value contained in the CRC_INIT register. When the CRC_INIT register is written, the CRC_DR register is automatically initialized.

The CRC_IDR register may be used to hold temporary values related to the CRC calculation. It is not affected by the RESET bit in the CRC_CR register.

Polynomial programmability

The polynomial coefficients can be fully customized via the CRC_POL register. The polynomial bit width can be configured to 7-bit, 8-bit, 16-bit, or 32-bit based on the POLYSIZE[1:0] bits in the CRC_CR register. Even polynomials are not supported. For example: an even polynomial has the form $x+x^2+\dots+x^n$, while an odd polynomial has the form $1+x+x^2+\dots+x^n$.

If the CRC data bit width is less than 32 bits, the CRC value will be read starting from the least significant bit of CRC_DR.

In order to obtain a reliable CRC calculation result, the value or size of the polynomial cannot be changed during the calculation of the CRC. As a result, if CRC calculation is in progress, the application must either reset the CRC or read the value of CRC_DR before changing the CRC polynomial.

The default polynomial is the CRC-32 (Ethernet) polynomial: 0x4C11DB7.

12.4. CRC registers

12.4.1. CRC Data Register (CRC_DR)

Address offset: 0x00

Reset value: 0xFFFF FFFF

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

Bit	Name	R/W	Reset Value	Function
31:0	DR	RW	32'hFFFFFFFF	Data register. This register is used to write new data to the CRC calculator. It holds the previous CRC calculation result when it is read.

				If the CRC data is less than 32 bits, the lower bits of this register are used to read or write the correct data.

12.4.2. CRC Independent Data Register (CRC_IDR)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	IDR[31:0]	RW	0	General-purpose 32-bit data register These bits can be used as a temporary storage location for bytes. This register is not affected by CRC resets generated by the RESET bit in the CRC_CR register. Note: This register does not participate in CRC calculation and can store any data.

12.4.3. CRC Control Register (CRC_CR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REV_OUT	REV_IN[1:0]	POLYSIZE[1:0]	Res.	Res.	Res.	Res.	RESET
								RW	RW	RW	RW	RW			W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	REV_OUT	RW	0	Invert output data This bit determines whether the bit order of the output data is inverted 0: The bit order of the output data is not inverted 1: Bit order inversion of output data
6:5	REV_IN[1:0]	RW	0	Reverse input data These 2 bits determine whether the bit order of input data is reversed. 00: Bit order not reversed 01: Each byte of input data is reversed 10: Each half-word of input data is reversed 11: Each word of input data is reversed
4:3	POLYSIZE[1:0]	RW	0	CRC polynomial bit width These 2 bits determine the bit width of the CRC polynomial 00: 32-bit CRC polynomial 01: 16-bit CRC polynomial 10: 8-bit CRC polynomial 11: 7-bit CRC polynomial
2:1	Reserved	-	-	Reserved
0	RESET	W	0	This bit is set by software to reset the CRC calculation unit and copy the content of the CRC_INIT register to the data register. This bit can only be set, it is automatically cleared by hardware.

12.4.4. CRC Initial Value Register (CRC_INIT)

Address offset: 0x10

Reset value: 0xFFFF FFFF

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

INIT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	INIT	RW	32'hFFFFFFFF	Programmable CRC initial value This register is used to store the initial CRC value.

12.4.5. CRC Polynomial Register (CRC_POL)

Address offset: 0x14

Reset value: 0x04C1_1DB7

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

Bit	Name	R/W	Reset Value	Function
31:0	POL	RW	32'h04C1_1DB7	Programmable CRC polynomial This register is used to store the coefficients of the CRC polynomial If the width of the CRC polynomial is less than 32 bits, it will be stored in this register in a right-aligned manner.

13. Analog-to-Digital Converter (ADC)

13.1. Introduction to ADC

The chip features one 12-bit SARADC (Successive Approximation Register ADC). The module has a total of up to 30 channels to be measured, including 25 external and 5 internal channels. The external channels include 3 pairs of differential channels.

A/D conversion of the various channels can be performed in single, continuous, or discontinuous mode. The result of the ADC is stored in a left-aligned or right-aligned 16-bit data register.

The analog watchdog allows the application to detect whether the input voltage exceeds user-defined high or low thresholds.

An efficient low-power mode is implemented to allow very low consumption at low frequency.

13.2. ADC main features

- High performance
 - Configurable resolutions of 12-bit, 10-bit, 8-bit, and 6-bit
 - ADC conversion time: 0.3 μ s @ 12-bit (3 MSps, 2.5 T SMP, fADC_CLK = 48 MHz)
 - Supports rule transformation and injection transformation
 - Support self - calibration (initiated by software)
 - Programmable sampling time
 - Programmable data alignment mode (left or right alignment)
 - DMA support for regular channel conversion
 - Configuration supports 16 rule sequence transformations
 - Configuration supports 4 injection sequence transformations
 - Configuration supports 4 injection channel conversion data registers
- Low-power
 - Reduce operating frequency for low power operation while still maintaining proper ADC performance
 - Auto off mode: prevents ADC overrun in applications with low frequency PCLK
- Analog input channels
 - 25 external analog inputs
 - One internal temperature sensor (VTS) channel
 - One internal reference voltage (V_{REFINT}) channel
 - One internal power supply monitor (V_{CC/3}) channel
 - One internal OPA channel
 - One internal DAC channel
 - Three pairs of differential channels
- Conversion operation trigger methods
 - By software
 - Hardware trigger with configurable polarity

- Conversion modes
 - Single (sequence) mode: One trigger performs a single conversion on all selected channels
 - Continuous mode: One trigger continuously converts the selected channel until terminated by software
 - Discontinuous mode: One trigger converts the configured number of channels in the subsequence, multiple triggers convert all channels.
- Interrupt generation
 - Single channel sampling completed
 - Single channel conversion completed
 - End of sequence conversion
 - Analog watchdog
 - Overflow event
 - ADC ready event
- Analog watchdog
 - Configure all channels to support watchdog
 - Configure selected channel to support watchdog
- Oversampling
 - 16-bit data register
 - Oversampling ratio can be adjusted from 2x to 256x
 - Programmable data can be shifted up to 8 bits
- Data preprocessing
 - Gain compensation
 - Offset compensation

13.3. ADC functional description

13.3.1. ADC block diagram

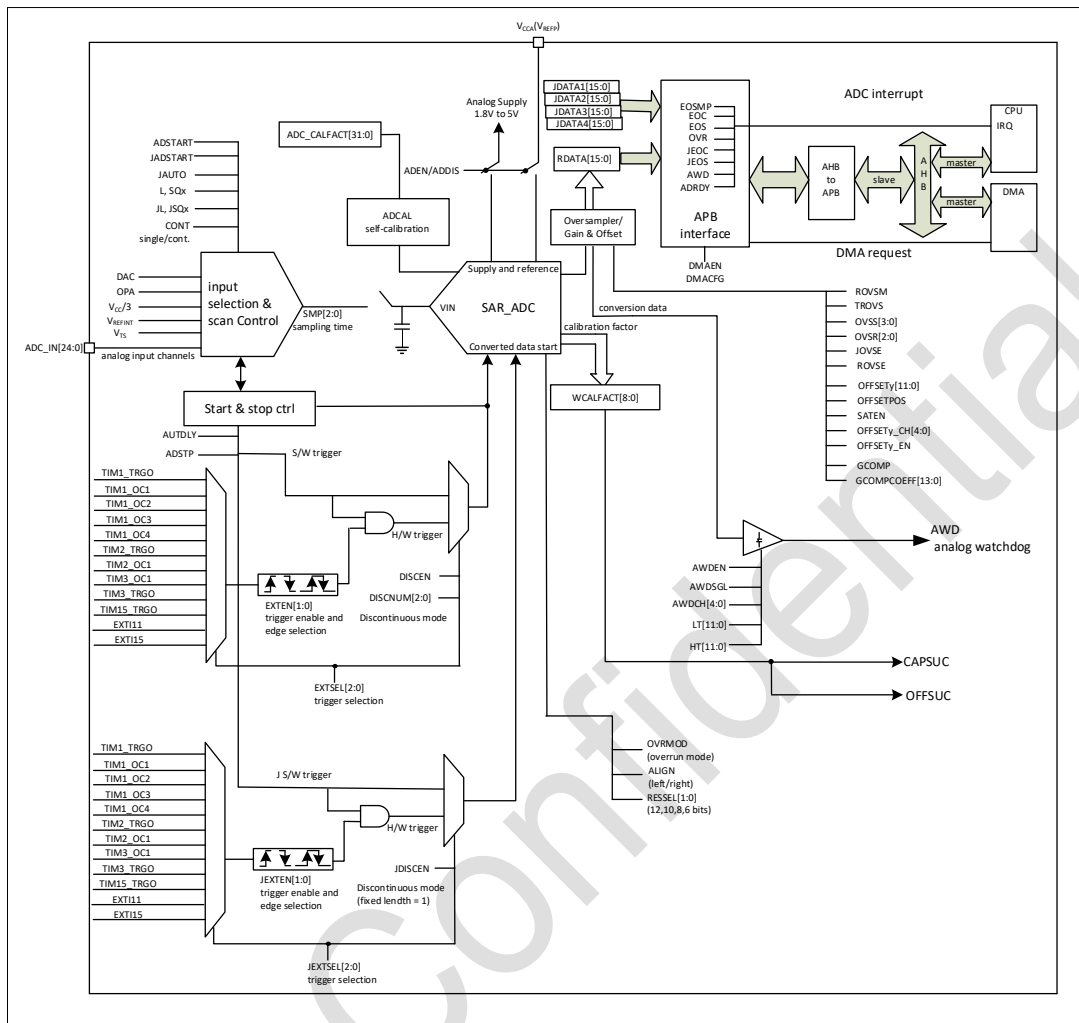


Figure 13-1 ADC block diagram

13.3.2. ADC single-ended and differential channels

ADC channels can be configured as single-ended or differential inputs through the ADC_CCR.DIFF_EN register. When DIFF_EN is 1, some channels are in differential input mode; otherwise, they are in single-ended input mode.

Must be configured when the ADC is disabled (ADEN = 0).

In single-ended mode, the analog voltage of channel i is the difference between $VINP[i]$ and V_{REF-} .

In differential mode, the analog voltage of channel i is the $VINP[i]$ and $VINN[i]$ voltage difference.

In differential mode, the input voltage range is from $-V_{REF+}$ to V_{REF+} , achieving a full scale of $2 \times V_{REF+}$.

When $VINP[i]$ equals V_{REF-} and $VINN[i]$ equals V_{REF+} , the maximum negative differential input voltage ($-V_{REF+}$) corresponds to 0x000 output. When $VINP[i]$ equals V_{REF+} and $VINN[i]$ equals V_{REF-} , the maximum positive differential input voltage (V_{REF+}) corresponds to 0xFFF output. When $VINP[i]$ and $VINN[i]$ are connected together, the zero differential input voltage corresponds to 0x800 output.

The ADC sensitivity in differential mode is twice less than in single-ended mode.

Internal channels only use single-ended mode.

The following figure is a schematic diagram of a single-ended/differential channel:

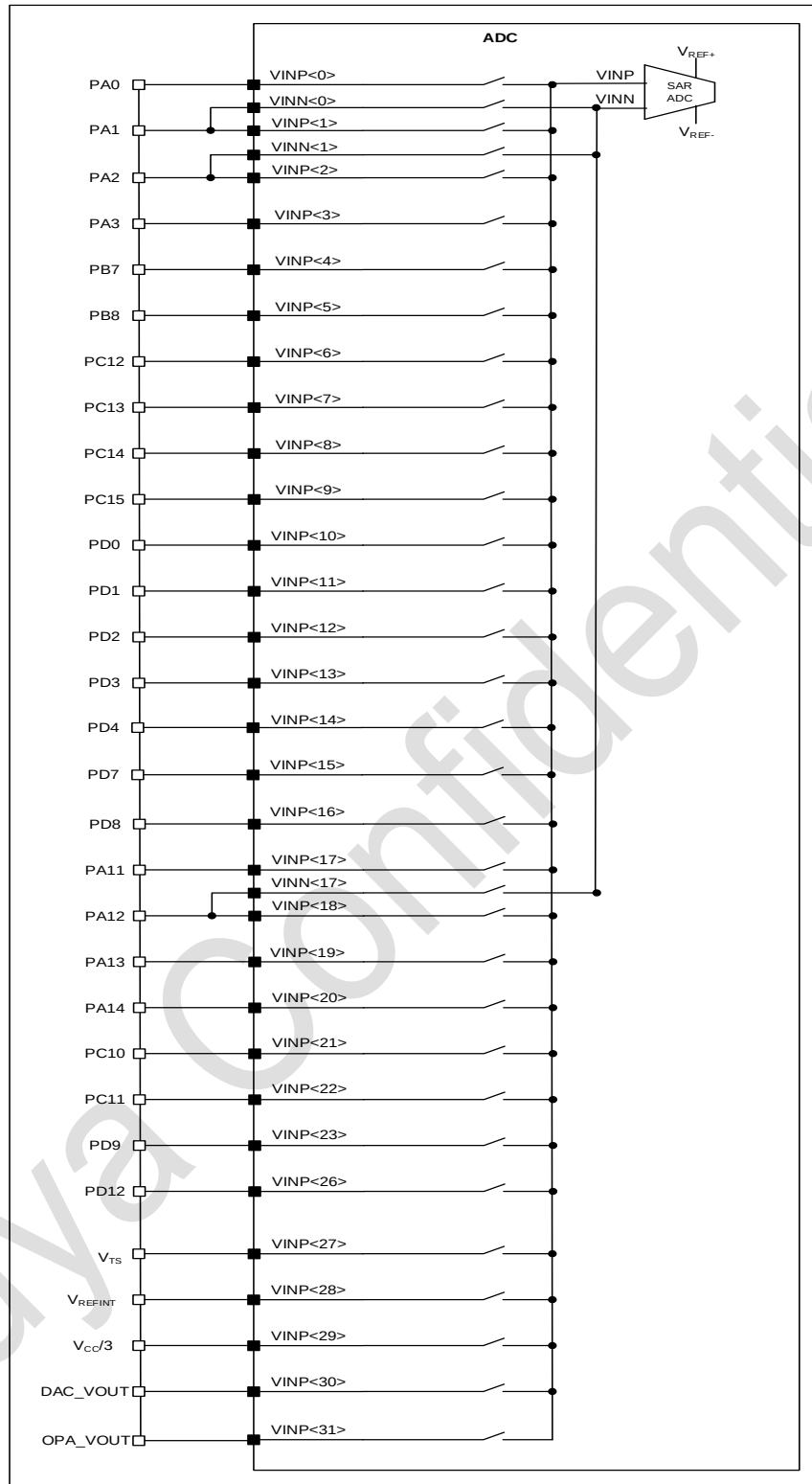


Figure 13-2 ADC channel connection relationship

Supports 3 differential pairs:

- PA0<->PA1, when ADC_CCR.DIFF_EN=1 and channel PA0 is selected, PA1 automatically serves as the INN terminal
- When PA1 <-> PA2, ADC_CCR.DIFF_EN = 1 and the channel selects PA1, PA2 automatically serves as the INN terminal

- PA11<->PA12, when ADC_CCR.DIFF_EN=1 and channel PA11 is selected, PA12 automatically serves as the INN terminal

13.3.3. ADC Calibration (ADCAL)

This ADC features calibration functionality (software-initiated). The ADC calculates calibration coefficients during the calibration process and applies these coefficients until the next power-down. During calibration, the application must not use the ADC and must wait for calibration to complete (ADEN=0).

Calibration is preliminary to any ADC operation. Calibration is used to eliminate errors caused by process variations between chips.

ADC software calibration

Setting ADCAL=1 in software initiates calibration. Calibration can only be started when the ADC is disabled (ADEN=0), and the ADC_CLK during calibration must not exceed 1/4 of the maximum ADC_CLK during normal operation (48 MHz), i.e., 12 MHz.

When calibration is complete, ADCAL is cleared to 0 by hardware. After calibration completes, the calibration factor can be read from the ADC_CALFACT register. The calibration factor remains valid until a system reset occurs.

When the ADC's operating conditions change (V_{CC} change is the primary factor for ADC offset variation, followed by temperature change), recalibration is recommended.

Configure the number of repetitions for the calibration process by setting the CALNUM[2:0] bit field in the ADC_CFGR2 register, and average the results for more accurate calibration.

The internal calibration register can be reset by setting the RSTCAL of ADC_CR, which is set to 0 by a hardware clear. This bit is cleared after the calibration register is initialized (i.e., after RSTCAL is set to 1).

Software procedure to calibrate the ADC:

- Configure the ADC_CCR register to ensure the ADC_CLK meets requirements (must not exceed 1/4 of the maximum ADC_CLK during normal operation (48 MHz))
- Confirm ADC_CR.ADEN=0
- Set ADC_CR.RSTCAL (this step is optional);
- Set the number of calibrations ADC_CFGR2. CALNUM (this step is optional);
- Set ADC_CR.ADCAL=1
- Wait for ADC_SR.EOCAL = 1
- Read the CAPSUC and OFFSUC status bits in the ADC_CALFACT register to confirm whether calibration was successful.

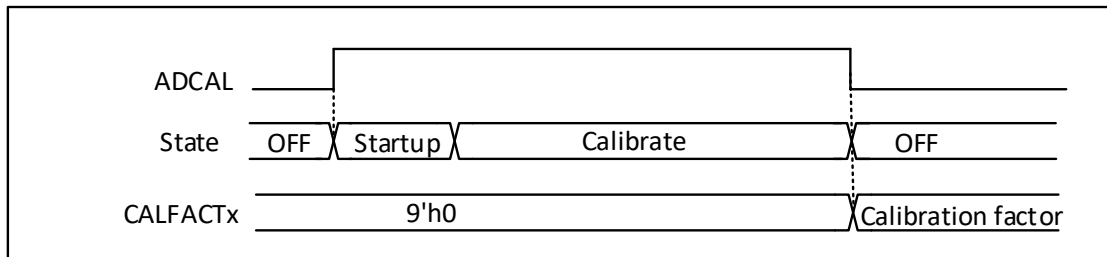


Figure 13-3 ADC Calibration

13.3.4. Software read/write calibration factor

The software can read/write calibration factors by following these steps:

Write calibration factor steps

- Configure the ADC_CCR register, select PCLK as the ADC_CLK clock source
- Make sure that ADEN = 1 and no conversion is in progress, while ADCAL = 0.
- Enable WRVLD, use FACTSEL[4:0] to select which capacitor to write, set WCALFACT[8:0] to write the calibration factor.
- When starting AD conversion, the calibration factor is automatically injected into the analog ADC.

Read calibration factor steps

- Configure the ADC_CCR register, select PCLK as the ADC_CLK clock source
- Make sure that ADEN = 1 and no conversion is in progress, while ADCAL = 0.
- Enable RDVLD, use FACTSEL[4:0] to select which capacitor to read
- Read the RCALFACT[8:0] calibration factor.

13.3.5. ADC switch control (ADEN, ADDIS, ADRDY)

At MCU power-up, the ADC is disabled and put in power-down mode (ADEN=0).

The ADEN bit is used to control the bit to turn the ADC on or off. After the ADC is enabled, the ADC requires a settling time tSTAB before accurate conversion can begin, as shown in the figure below.

There are two control bits used to enable or disable the ADC:

- When ADEN = 1, ADC is enabled. When the ADC is ready for operation, the flag ADRDY will be set.
- When ADDIS = 1, disable the ADC. Once the analog ADC is completely disabled (inactive state), the hardware will automatically clear ADEN and ADDIS.

ADC regular conversion is initiated by setting ADSTART. Depending on EXTEN configuration, conversion can start immediately (software trigger) or wait for hardware trigger.

ADC injected conversion is initiated by setting JADSTART. Depending on JEXTEN configuration, conversion can start immediately (software trigger) or wait for hardware trigger.

13.3.5.1. Steps for software to enable ADC

1. Clear the ADRDY bit in the ADC_ISR register by writing '1'.
2. Set ADEN=1 to enable the ADC.

3. Wait for $ADRDY=1$ ($ADRDY$ will be set after the ADC startup time). If interrupts are enabled by setting the $ADRDYIE$ bit in the ADC_IER register to 1, they can be handled via interrupt.

13.3.5.2. Steps for software to disable ADC

1. Check that both $ADSTART$ and $JADSTART$ are 0 to ensure no ongoing conversion. If necessary, stop any ongoing regular or injected conversion by setting $ADSTP=1$ or $JADSTP=1$ in the ADC_CR register, then wait for $ADSTP=0$ and $JADSTP=0$.
2. Set $ADDIS=1$ to disable the ADC.
3. If required by the application, wait until $ADEN=0$, indicating the analog ADC is actually disabled (once $ADEN=0$, $ADDIS$ will be automatically reset).
4. Clear the $ADRDY$ bit in the ADC_ISR register by writing '1' (optional).

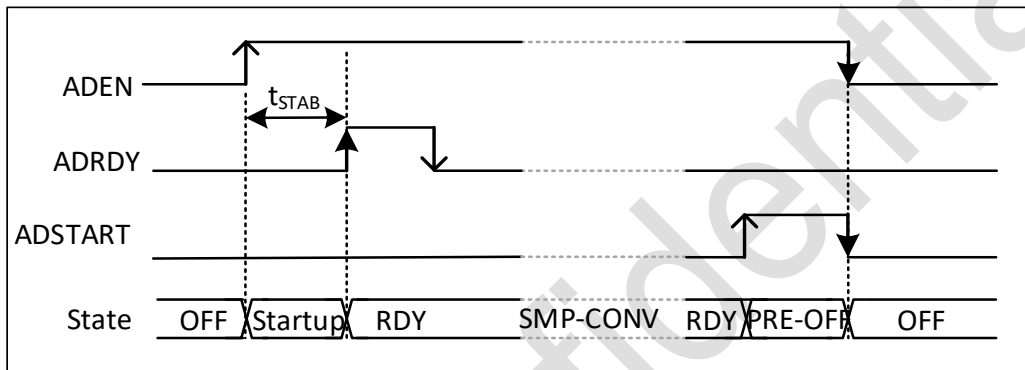


Figure 13-4 Enable/Disable ADC

13.3.6. ADC control bit register operations

ADC register operations follow these principles:

- The software is only allowed to write the $ADSTART$, $JADSTART$, and $ADDIS$ control bits in the ADC_CR register when the ADC is enabled and there is no pending request to disable the ADC (i.e., $ADEN$ is 1 and $ADDIS$ is 0).
- The software is only allowed to write the $ADSTP$ or $JADSTP$ control bits in the ADC_CR register when the ADC is enabled, may be converting, and there is no pending request to disable the ADC (i.e., $ADSTART$ or $JADSTART$ equals 1 and $ADDIS$ is 0).
- $ADDIS=1$ can only be configured by software when $ADEN=1$ and both $ADSTART$ and $JADSTART$ are 0.
- ADC_CFGR , ADC_CFGR2 , ADC_SMPRx , ADC_SQRy , ADC_JSQR , ADC_OFRy , ADC_GCOMP , ADC_TR , ADC_IER register operations comply with:
 - Regular sequence-related registers are configured when $ADEN=1$ and $ADSTART=0$.
 - Injected sequence-related registers are configured when $ADEN=1$ and $JADSTART=0$.

Note: The hardware does not provide protection against these prohibited write accesses. If such writes occur, the ADC behavior may enter an unknown state. To recover from this condition, the ADC must be disabled (clear $ADEN$ to 0 and simultaneously clear all bits in the ADC_CR register).

13.3.7. ADC clock

The ADC has a dual clock-domain architecture, so that the ADC clock (ADC_CLK) is independent from the APB clock (PCLK). ADC_CLK can be generated from two possible clock sources, as shown below:

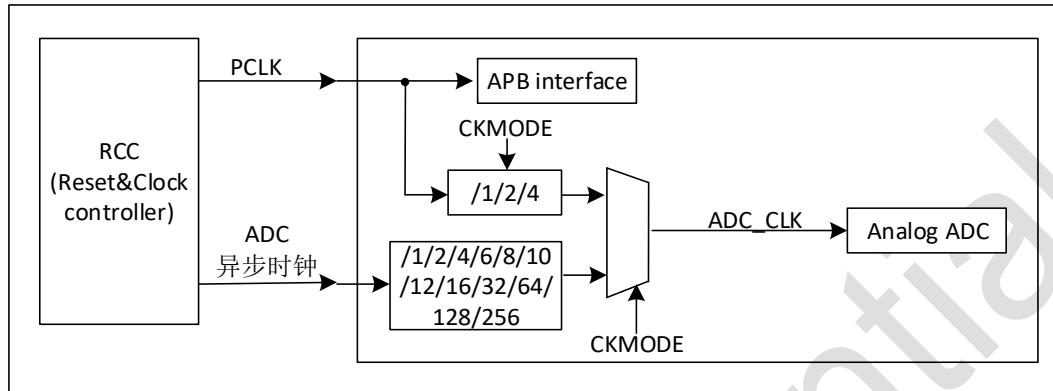


Figure 13-5 ADC clock structure

The input clock for the analog ADC can be selected between two different clock sources:

- The ADC clock can be a specific clock source called the "ADC asynchronous clock," which is independent of and asynchronous to the APB clock. To select this clock scheme, the CKMODE[1:0] bits of the ADC_CCR register must be reset.
- The ADC clock can be provided by dividing the APB clock of the ADC bus interface by a programmable factor (1, 2, or 4, depending on the CKMODE[1:0] bits). To select this clock scheme, the ADC_CCR.CKMODE[1:0] register bits must not be configured as "00".

In option a), when programming the PRESC[3:0] bits in the ADC_CCR register, the resulting ADC clock is finally divided by a prescaler coefficient (1, 2, 4, 6, 8, 10, 12, 16, 32, 64, 128, 256).

The advantage of option a) is that the maximum ADC clock frequency can be achieved regardless of the selected APB clock scheme.

Note: When selecting CKMODE[1:0]=11 (PCLK divided by 1), the user must ensure a 50% duty cycle for the PCLK. This can be achieved by selecting a system clock with a 50% duty cycle and not configuring the APB prescaler in the RCC (both HPRE and PPRE in RCC_CFGR are configured as no division).

Note: Only ADC_CLK frequencies less than or equal to the PCLK frequency are supported.

13.3.8. ADC Channel Selection (SQRx, JSQR)

There are up to 30 multiplexed channels:

- 25 analog inputs from GPIO pins (ADC_IN0...ADC_IN24)
- 5 internal analog inputs (temperature sensor, internal reference voltage, $V_{CC}/3$, OPA output, and DAC output)

It is possible to convert a single channel or to automatically scan a sequence of channels. ADC channel selection can be configured via ADC_SQRx or ADC_JSQR.

ADC divides conversions into two groups: rule conversions and injection conversions. Each group contains a sequence of transformations that can be completed on any channel in any order. For example, the sequence may be converted in the following order: ADC_IN3, ADC_IN8, ADC_IN2, ADC_IN2, ADC_IN0, ADC_IN2, ADC_IN2, ADC_IN15. All channels can be converted by injection or regular channel group.

- A rule transformation group consists of up to 16 transformations. The regular channel of the conversion sequence and its conversion order are selected in the ADC_SQRx register. The sequence length in the rule transformation group must be written L [3: 0].
- An injection conversion group consists of a maximum of 4 conversions. The injection channel and its conversion order are selected in the ADC_JSQR register. The sequence length in the injected conversion group must be written to the JL[1:0] bits.

The ADC_SQRy register must not be modified during possible rule conversion periods. To do this, the rule conversion of the ADC must first be stopped by writing ADSTP=1.

13.3.9. ADC programmable sampling time (SMP)

Before starting a conversion, the ADC needs to establish a direct connection between the voltage source to be measured and the embedded sampling capacitor of the ADC. This sampling time must be enough for the input voltage source to charge the sample and hold capacitor to the input voltage level.

Having a programmable sampling time allows to trim the conversion speed according to the input resistance of the input voltage source.

The number of ADC clock cycles used for ADC sampling of the input voltage can be modified using the SMP[2:0] bits in the ADC_SMPRx register, with each channel independently configurable for different sampling times. If required by the application, the software can change and adapt this sampling time between each conversions.

The total conversion time is calculated as follows:

$$t_{\text{CONV}} = (\text{sample time} + (\text{conversion resolution} + 0.5)) \times \text{ADC clock cycle}$$

For example:

When ADC_CLK = 16 MHz, resolution is 12 bits, and sampling time is 2.5 ADC clock cycles:

$$t_{\text{CONV}} = (2.5 + 12.5) \times \text{ADC clock cycles} = 15 \times \text{ADC clock cycles} = 0.9375 \mu\text{s}$$

The EOSMP flag bit is used to indicate the end of the sampling phase.

Restrictions on sampling time:

- For each channel, the SMP [2: 0] bit must be programmed to the minimum sampling time specified in the ADC characteristics section of the datasheet.

13.3.10. ADC single conversion mode (CONT=0, DISCEN=0)

In Single conversion mode, the ADC performs a single sequence of conversions, converting all the channels once. When CONT = 0 and DISCEN = 0 in the ADC_CFGR register, the ADC is in a single conversion mode.

This mode can be started by the following method (hardware trigger):

- Configure ADC_CFGR.EXTEN and ADC_JSQR.JEXTEN registers to a value other than 2'b00
- Set the ADSTART bit in the ADC_CR register (rule conversion)
- Set the JADSTART bit in the ADC_CR register (injection conversion)
- ADSTART = 1 and hardware triggered event (rule transformation)
- JADSTART = 1 and hardware triggered event (injection transition)

Rule conversion, after each conversion is done:

- The converted data are stored in the 16-bit ADC_DR register.
- The EOC (end of conversion) flag is set
- An interrupt is generated if the EOCIE bit is set.

Rule conversion, after all channel sequence conversion is completed:

- EOS (end of sequence) flag is set (EOC will also be set simultaneously)
- An interrupt is generated if the EOSIE bit is set.

Injection transformations, after each transformation is complete:

- The converted data result is stored in the 16-bit register ADC_JDRx.
- The JEOC (end of conversion) flag is set
- An interrupt is generated if the JEOCIE bit is set.

Injection conversion, after all channel sequence conversion is complete:

- JEOS (end of sequence) flag is set (JEOC will also be set simultaneously)
- An interrupt is generated if the JEOSIE bit is set.

After the conversion is complete, the ADC conversion stops until a new trigger event or ADSTART/JADSTART is reset.

If you convert a single channel, you can configure the sequence length to 1.

13.3.11. ADC continuous conversion mode (CONT=1, DISCEN=0)

This pattern is for rule transformations only.

In continuous conversion mode, when a software or hardware trigger event occurs, the ADC performs a sequence conversion. All channels are converted and the same sequence conversion is automatically restarted.

When CONT = 1 in the register ADC_CFGR, the ADC is selected to be in the continuous conversion mode.

Conversion is started by either:

- Setting the ADSTART bit in the ADC_CR register
- ADSTART=1 and hardware trigger event triggered

Inside the sequence, after each conversion is complete:

- The converted data are stored in the 16-bit ADC_DR register.
- The EOC (end of conversion) flag is set
- An interrupt is generated if the EOCIE bit is set.

After the sequence of conversions is complete:

- EOS (End of Sequence) flag set (EOC will also be set simultaneously)

- An interrupt is generated if the EOSIE bit is set.

(A new sequence restarts immediately and the ADC continuously repeats the conversion sequence.)

Note: To convert a single channel, program a sequence with a length of 1.

ADC cannot be in both discontinuous conversion mode and continuous conversion mode at the same time (i.e., configuring DISCEN=1 and CONT=1 simultaneously is prohibited)

Injected channels do not support continuous conversion mode. The only exception is when injected channels are configured to automatically convert after regular channels in continuous mode (using the JAUTO bit).

13.3.12. ADC discontinuous conversion mode (DISCEN=1 or JDISCEN=1)

13.3.12.1. Regular group mode

This mode is enabled by setting the DISCEN bit in the ADC_CFGR register. The ADC divides a selected sequence into sub-sequences (sequence length 1-8), and the entire sequence conversion is completed by triggering the sub-sequences multiple times externally. DISCNUM specifies the regular channel subsequence length, and ADC_SQRx specifies all conversion channels of a regular sequence, where L [3: 0] bit specifies the sequence length.

For example:

DISCEN=1, DISCNUM=2, channels to be converted: 0, 3, 7, 10, and ADC_SQR0 has SQ1=0, SQ2=3, SQ3=7, SQ4=10:

- First trigger: Channels 0 and 3 are converted, and an EOC event is generated after each conversion
- Second trigger: Channels 7 and 10 are converted, an EOC event is generated after each conversion, and an EOS event is generated after channel 10 conversion is completed
- Third trigger: Channels 0 and 3 are converted, and an EOC event is generated after each conversion
- ...

DISCEN=0, channels to be converted: 0, 3, 7, 10, and ADC_SQR0 has SQ1=0, SQ2=3, SQ3=7, SQ4=10:

- First trigger: The entire sequence is converted, in order: channels 0, 3, 7, and 10.

Each time the conversion is completed, an EOC event is generated, and the conversion to the last channel generates an EOS event in addition to the EOC.

- Any trigger event restarts the full sequence transition.

Note: It is impossible for the ADC to be in both continuous mode and discontinuous conversion mode simultaneously, i.e., configuring DISCEN=1 and CONT=1 at the same time is prohibited.

13.3.12.2. Injected group mode

This mode is enabled by setting the JDISCEN bit in the ADC_CFGR register. An external trigger signal initiates one conversion of the channels described by ADC_JSQR until all conversions in the sequence are completed. The total sequence length is defined by the JL[1:0] bits.

For example:

JDISCEN=1, channels to be converted: 1, 2, 3, and ADC_JSQR has JSQ1=1, JSQ2=3, JSQ3=7:

- First trigger: Channel 1 is converted, and a JEOP event is generated after each conversion
- Second trigger: Convert channel 2, generates a JEOP event upon each conversion completion
- Third trigger: Convert channel 3, generates a JEOP event upon each conversion completion and JEOS event at sequence end
- Fourth trigger: Channel 1 is converted, generates a JEOP event upon each conversion completion
- ...

Notes:

- When all injected channels are converted, the next trigger initiates conversion of the first injected channel
- Auto-injected and discontinuous modes cannot be used simultaneously: When JAUTO is set to 1, DISCEN and JDISCEN bits must remain cleared by software.
- Avoid configuring both regular and injected groups in discontinuous mode simultaneously

13.3.13. ADC injected channel management

13.3.13.1. Triggered injection mode

In triggered injection mode, the ADC_CFGR.JAUTO bit must be 0.

- Conversion of a regular channel group is initiated either by external trigger or by setting the ADSTART bit in ADC_CR register.
- If an external injection trigger occurs or the JADSTART bit in ADC_CR register is set during regular channel group conversion, the current conversion is reset and an injected channel sequence starts (all injected channels are converted once).
- Then, conversion of regular group channels resumes from the interrupted regular conversion.
- If a regular trigger event occurs during injected conversion (multiple triggers treated as one), the injected conversion won't be interrupted; the regular sequence executes after the injected sequence ends. (For cases where injection doesn't interrupt regular conversion: if regular was interrupted before injection, complete the interrupted regular conversion first). Special case: For injected conversion + wait (AUTDLY=1) + software-triggered regular, software must trigger ADSTART to start regular conversion only after JEOS is cleared.

Note: When using triggered injection, ensure trigger intervals are longer than the injection sequence.

Example: If sequence length is 30 ADC clock cycles (two conversions with 2.5 ADC_CLK sampling time), minimum trigger interval must be 31 ADC clock cycles.

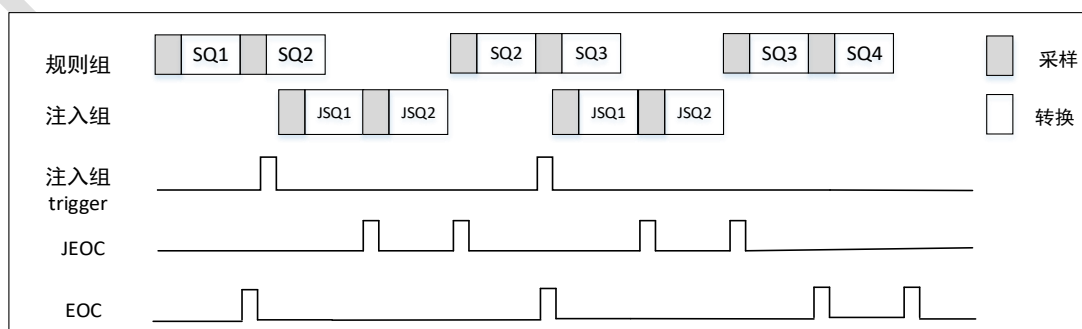


Figure 13-6 ADC triggered injection timing

13.3.13.2. Automatic injection mode

If the JAUTO bit in the ADC_CFGR register is set, the channels in the injection group are automatically converted after the regular channel group. This can be used to convert up to 20 conversion sequence registers programmed in ADC_SQRy and ADC_JSQR (maximum 16 regular channels and 4 injected channels).

In this mode, setting the ADSTART bit in the ADC_CR register to 1 initiates regular conversion, followed by injected conversion (JADSTART must remain cleared). Setting the ADSTP bit terminates the rule conversion and injection conversion (the JADSTP bit cannot be used).

In this mode, external triggers on the injection channel must be disabled.

If the CONT bit is set in addition to the JAUTO bit, continuous conversion of "regular channels + injected channels" occurs.

Note: It is not possible to use both auto-injection and non-continuous modes simultaneously.

When DMA is used in JAUTO mode to export data from the regular conversion sequence, it must be programmed in circular mode (set the CIRC bit in the DMA_CCRx register). If the CIRC bit is reset (single mode), the JAUTO sequence will stop when the DMA transfer complete event occurs.

13.3.14. ADC conversion start (ADSTART, JADSTART)

The software initiates the ADC rule conversion by setting ADSTART = 1.

When ADSTART is set, regular conversion:

- When EXTEN=2'b00 (software trigger), starts immediately
- When EXTEN is not equal to 2'b00, starts upon detecting a valid edge of the selected regular hardware trigger source

The software initiates the ADC injection conversion by setting JADSTART = 1.

After setting JADSTART, injected conversion:

- When JEXTEN[1:0] = 2'b00 (software trigger), starts immediately
- If JEXTEN[1:0] is not equal to 2'b00, starts upon detecting a valid edge of the selected injected hardware trigger source

NOTE: In auto-injection mode (JAUTO = 1), use the ADSTART bit to initiate the rule conversion, followed by the auto-injection conversion (JADSTART must remain clear).

The ADSTART and JADSTART bits are also used to indicate whether an ADC conversion operation is currently in progress. When the ADC is idle, the ADC can be reconfigured with ADSTART=0 and JADSTART=0.

The ADSTART bit can be cleared by hardware:

- Single or continuous conversion mode triggered by software (CONT=0, EXTSEL=2'b00)

At the end of any regular conversion sequence (EOS=1) or at the end of a subsequence conversion when DISCEN = 1

- Non-continuous conversion mode triggered by software (CONT=0, DISCEN=1, EXTSEL=2'b00)
- After the end of the subsequence conversion configured by DISNUM (EOC=1 for the last channel of the subsequence)

- In all cases (CONT=X, EXTSEL=X)

- After software configures ADSTP

Note:

1. In continuous mode (CONT=1), the ADSTART bit cannot be cleared by hardware due to EOS because the sequence automatically restarts conversion.
2. When single conversion mode selects hardware trigger (CONT=0 and EXTSEL≠2'b00), ADSTART will not be cleared by hardware after the EOS flag is set. This avoids the need for software to reset the ADSTART bit and ensures no hardware trigger events are missed.

JADSTART cleared by hardware:

- Software-triggered single conversion mode (JEXTSEL = 2'b00)

- If JDISCEN=1, at the end of the injected conversion sequence (JEOS generation) or at the end of a sub-sequence conversion when JDISCEN = 1

- In all cases (JEXTSEL=x)

- After software configures JADSTP.

Note: When software trigger is selected, ADSTART should not be set if the EOC flag remains high.

13.3.15. ADC timing

The elapsed time between the start of a conversion and the end of conversion is the sum of the configured sampling time plus the successive approximation time depending on data resolution:

$$t_{ADC} = t_{SMPL} + t_{SAR} = [2.5 \text{ |min} + 12.5 \text{ |12bit}] \times t_{ADC\_CLK}$$

$$t_{ADC} = t_{SMPL} + t_{SAR} = 52.083\text{ns |min} + 260.375\text{ns |12bit} = 312.458\text{ns |min (for f}_{ADC\_CLK} = 48$$

MHz)

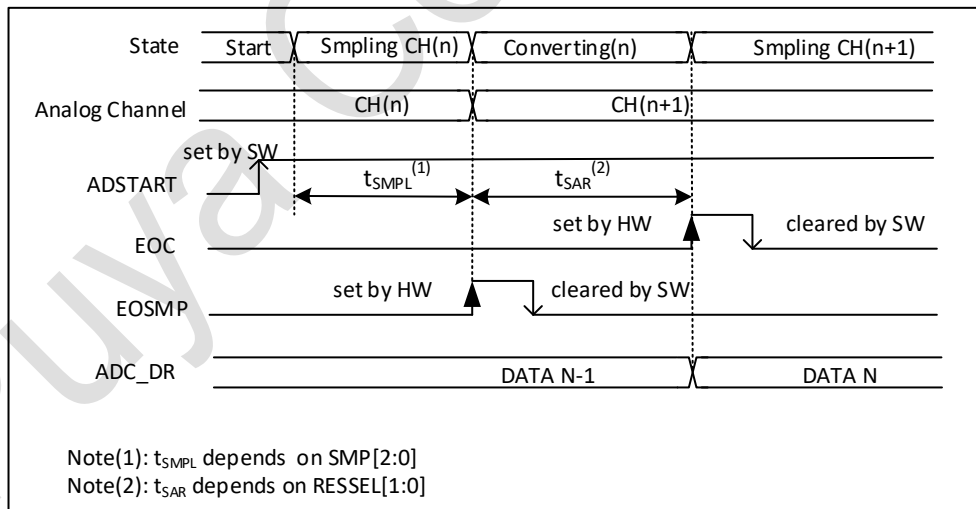


Figure 13-7 Analog-to-digital conversion timing

13.3.16. ADC stops ongoing conversion (ADSTP, JADSTP)

Setting ADSTP=1 in the ADC_CR register stops the current regular conversion, and setting JADSTP=1 in the ADC_CR register stops the current injected conversion.

Stopping the conversion resets the ongoing ADC operation. The ADC can then be reconfigured (e.g., changing channel selection or trigger) to prepare for a new operation.

Note that the injection transformation can be stopped while the rule transformation is still running, and vice versa. For example, this allows reconfiguring the injected conversion sequence and trigger while regular conversions are still running (and vice versa).

When the ADSTP bit is set by software, any ongoing regular conversion is aborted, and partial results are discarded (the ADC_DR register is not updated with the current conversion).

When the JADSTP bit is set by software, any ongoing injected conversion is aborted, and partial results are discarded (the ADC_JDRy register is not updated with the current conversion result).

The scan sequence is also aborted and reset (meaning restarting the ADC will initiate a new sequence).

Once this process completes, the ADSTP/ADSTART bit (for regular conversions) or JADSTP/JADSTART bit (for injected conversions) is cleared by hardware. Software must wait for ADSTART/JADSTART=0 before initiating a new conversion.

Note: In auto-injection mode (JAUTO = 1), setting the ADSTP bit terminates both the rule and injection conversion (JADSTP cannot be used).

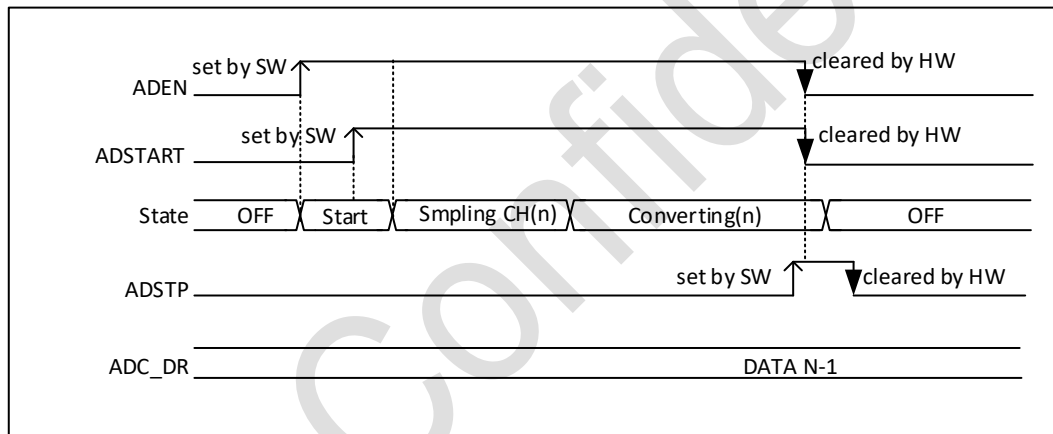


Figure 13-8 ADSTP Stop Rule Conversion

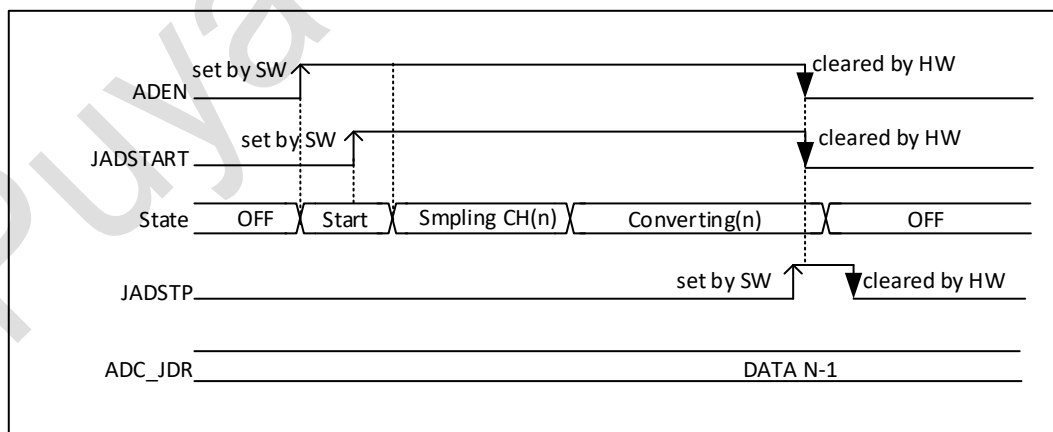


Figure 13-9 JADSTP Stop Injection Conversion

13.3.17. ADC hardware trigger and trigger polarity (EXTSEL, EXTEN, JEXTSEL, JEXTEN)

A conversion or a sequence of conversion can be triggered either by software or by an external event (for example timer and input pin). If the EXTEN[1:0] control bits (for rule conversion) or JEXTEN[1:0] bits (for injected conversion) are not equal to 2'b00, external events can trigger the conversion with the selected polarity.

Once the software sets ADSTART=1, the rule trigger selection becomes valid. Once the software sets JADSTART=1, the injected trigger selection becomes valid.

Any hardware triggers which occur while a conversion is ongoing are ignored. Any rule trigger during the period when ADSTP is high will be ignored. Any injected trigger during the period when JADSTP is high will be ignored.

- If the ADSTART=0 bit is set, any rule hardware trigger will be ignored.
- If the JADSTART=0 bit is set, any injected hardware trigger will be ignored.

The following table provides the correspondence between EXTEN[1:0] and JEXTEN[1:0] values and the polarity of the trigger source.

Table 13-1 EXTEN[1:0] and Trigger Source Polarity Correspondence

EXTEN[1:0]	Source
00	Rule hardware trigger detection disabled, software trigger enabled
01	Rule hardware rising edge trigger
10	Rule hardware falling edge trigger
11	Rule hardware rising and falling edge trigger

Note: The external trigger polarity cannot be changed during rule conversion.

Table 13-2 JEXTEN[1:0] and Trigger Source Polarity Correspondence

JEXTEN[1:0]	Source
00	Injected hardware trigger detection disabled, software trigger enabled
01	Injected hardware rising edge trigger
10	Injected hardware falling edge trigger
11	Injected hardware rising and falling edge trigger

Note: The external trigger polarity cannot be changed during injected conversion.

The EXTSEL and JEXTSEL control bits select which events from up to 16 possible events can trigger the conversion of the rule group and injected group.

Rule group conversion can be interrupted by an injected trigger.

Note: The trigger source selection for rule and injected conversions cannot be changed dynamically.

13.3.18. ADC Programmable Resolution (RES) – Fast Conversion

A faster transition time (tSAR) can be obtained by reducing the transition resolution. The conversion resolution can be configured to 12/10/8/6-bit modes by setting RES[1:0] in the ADC_CFGR register. When applications do not require high-precision data, low conversion resolution can be used to speed up conversion times.

Resolution mode reduces the conversion time of successive approximations, such as:

RES[1:0]	tSAR (ADC Clock Cy- cle)	tSAR(ns) @ fADC = 24 MHz	tSMP (ADC Clock Cy- cle)	tADC(tSMP = 2.5) (ADC Clock Cycle)	tSAR(ns) @ fADC = 24 MHz
12	13.5	562.5 ns	2.5	16	666.7 ns
10	11.5	479 ns	2.5	14	583 ns
8	9.5	395.8 ns	2.5	12	500 ns
6	7.5	312.5 ns	2.5	10	416.7 ns

13.3.19. ADC Conversion/Sampling End (EOC, JEOC, EOSMP)

The ADC notifies the application of each regular conversion end (EOC) event and each injected conversion (JEOC) event.

The ADC sets the EOC flag when new regular conversion data is available in the ADC_DR register. An interrupt can be generated if the EOCIE bit is set. The EOC flag is cleared by software by writing 1 to it or by reading the ADC_DR.

The ADC sets the JEOC flag when new injected conversion data is available in the ADC_JDRy register. An interrupt can be generated if the JEOCIE bit is set. The JEOC flag is cleared by software by writing 1 to it or by reading the corresponding ADC_JDRy register.

The ADC also notifies the end of the sampling phase by setting the status bit EOSMP (only for regular conversions). The EOSMP flag is cleared by software writing 1 to it. An interrupt can be generated if the EOSMPIE bit is set.

13.3.20. ADC Conversion Sequence End (EOS, JEOS)

The ADC notifies the application when the regular sequence is completed (EOS) and the injected sequence is completed (JEOS).

The ADC sets the EOS flag once the last data of the regular conversion sequence is available in the ADC_DR register. An interrupt can be generated if the EOSIE bit is set. The EOS flag is cleared by software by writing 1 to it.

The ADC sets the JEOS flag once the last data of the injected conversion sequence is completed. An interrupt can be generated if the JEOSIE bit is set. The JEOS flag is cleared by software by writing 1 to it.

13.3.21. ADC Timing Examples (Single/Continuous Mode, Hardware/Software Trigger)

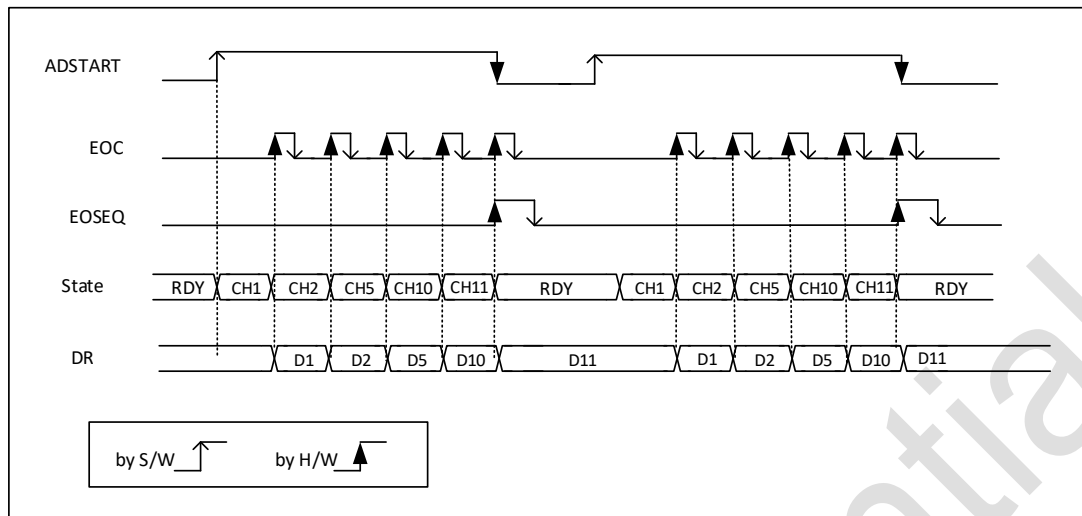


Figure 13-10 Regular Single Mode, Software Trigger

1. EXTEN = 0x0, CONT = 0
2. SQ1~5 are configured as channels 1, 2, 5, 10, 11 respectively, AUTDLY=0

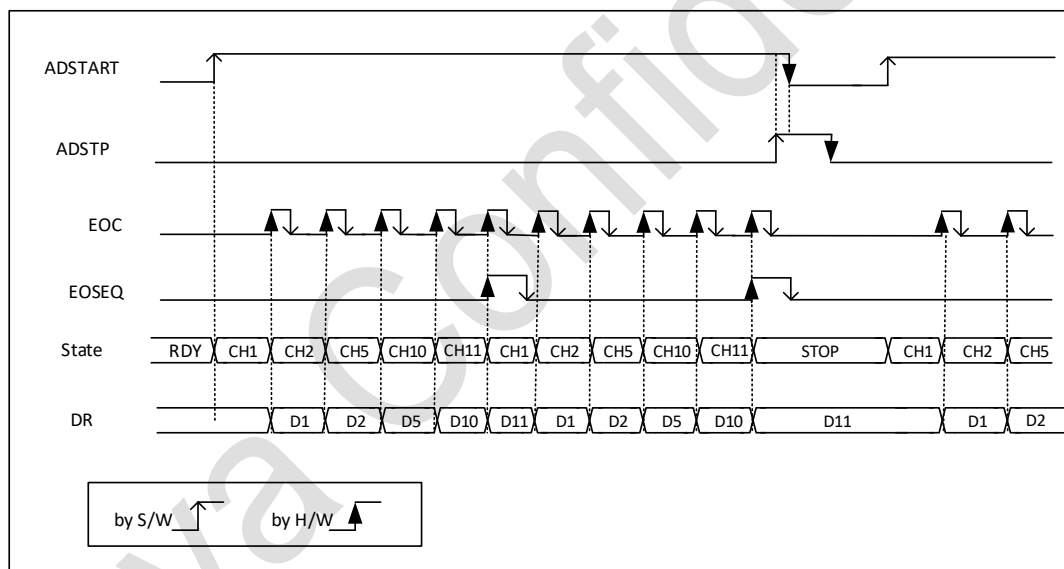


Figure 13-11 Regular Continuous Mode, Software Trigger

1. EXTEN = 0x0, CONT = 1,
2. SQ1~5 are configured as channels 1, 2, 5, 10, and 11 respectively, AUTDLY=0

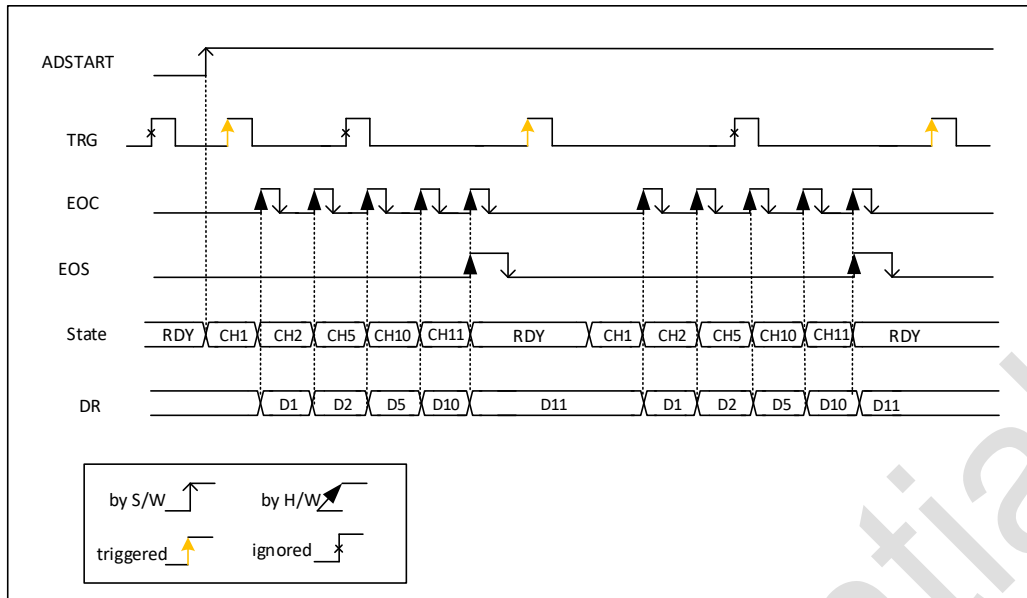


Figure 13-12 Rule single mode, hardware trigger

1. EXTSEL = TRGx, EXTEN = 0x1 (rising edge), CONT = 0
2. SQ1~5 are configured as channels 1, 2, 5, 10, 11 respectively, AUTDLY=0

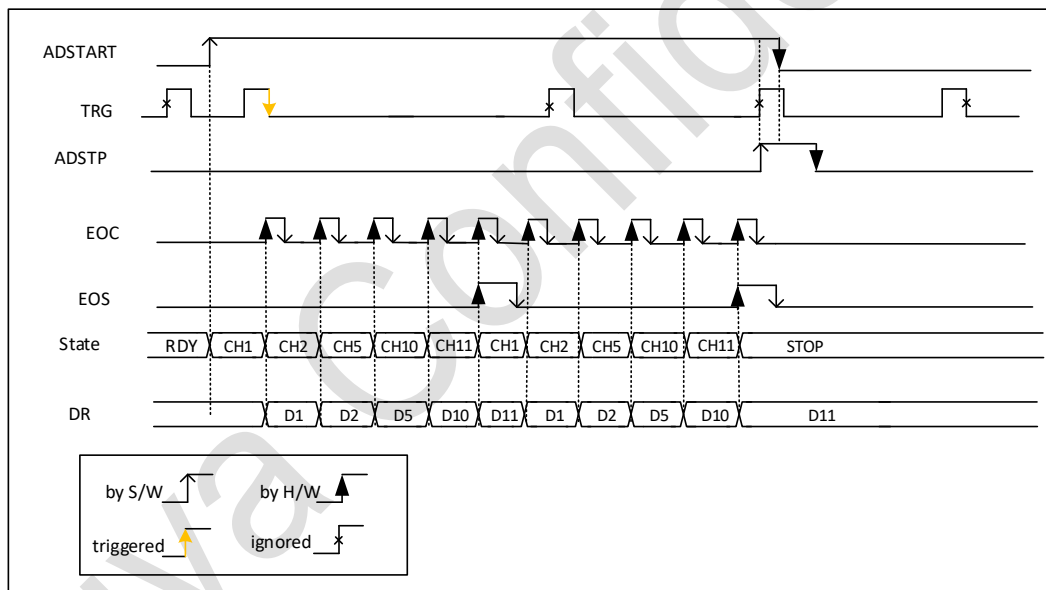


Figure 13-13 Rule continuous conversion, hardware trigger

1. EXTSEL = TRGx, EXTEN = 0x2 (falling edge), CONT = 1
2. SQ1~5 are configured as channels 1, 2, 5, 10, and 11 respectively, AUTDLY=0

13.3.22. ADC data management

13.3.22.1. Data registers and data alignment (ADC_DR, ADC_JDRy, ALIGN)

At the end of each rule conversion (when the EOC event occurs), the converted result data is stored in the 16-bit wide ADC_DR data register.

At the end of each injected conversion (when the JEOP event occurs), the converted result data is stored in the 16-bit wide ADC_JDRy data register.

The data format of ADC_DR and ADC_JDRy is related to the configured data alignment and conversion resolution. The ALIGN bit in the ADC_CFGR register is used to select the alignment of data storage. The data can be right-aligned (ALIGN=0) or left-aligned (ALIGN=1).

The following table shows the relationship between data alignment and resolution (oversampling is invalid ROVSE=0, JOVSE=0):

Table 13-3 Data alignment and resolution

ALIGN	RES	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0x0	0x0				DATA[11:0]											
	0x1	0x0						DATA[9:0]									
	0x2	0x0								DATA[7:0]							
	0x3	0x0										DATA[5:0]					
1	0x0	DATA[11:0]												0x0			
	0x1	DATA[9:0]										0x0					
	0x2	DATA[7:0]								0x0							
	0x3	0x0								DATA[5:0]						0x0	

Special case: When left-aligned, data is aligned in half-word units unless the resolution is set to 6 bits. In this case, the data is aligned in byte units.

Note: Left alignment is not supported in oversampling mode. When the ROVSE and/or JOVSE bits are set, the ALIGN bit value is ignored, and the ADC only provides right-aligned data.

13.3.22.2. Offset (OFFSETy, OFFSETy_CH, OFFSETy_EN)

An offset y (y = 1, 2, 3, 4) can be applied to the channel by setting OFFSETy_EN = 1 in the ADC_OFRy register. The application can select the offset channel via the OFFSETy_CH bit in the ADC_OFRy register. In this case, the conversion result is subtracted by the user-defined offset in OFFSETy, and the result may be negative. Therefore, the read data is signed, with the SEXT bit representing the extended sign value.

Note: Offset correction is not supported in oversampling mode. When the ROVSE and/or JOVSE bits are set, the value of the OFFSETy_EN bit in the ADC_OFRy register is ignored (treated as the reset value).

The following table shows the relationship between the offset calculated value and data resolution:

Table 13-4 Offset Calculated Value VS Resolution

Resolution (RES [1: 0])	Raw conversion result and offset subtraction		Result	Description
	Original conversion result, left-aligned	Offset		
00:12 bit	DATA[11:0]	OFFSET[11:0]	12-bit signed number	-
01:10 bit	{DATA [11: 2], 00}	OFFSET[11:0]	10-bit signed number	The user must configure OFFSET [1: 0] = 00
10:8bit	{DATA [11: 4], 0000}	OFFSET[11:0]	8-bit signed number	The user must configure OFFSET [3: 0] = 0000
11:6bit	{DATA [11: 6], 000000}	OFFSET[11:0]	6-bit signed number	The user must configure OFFSET [5: 0] = 000000

When reading data from ADC_DR (regular channel) or ADC_JDRy (injection channel, y = 1, 2, 3, 4) corresponding to channel "i":

--If one of the offsets for the corresponding channel is enabled (bit OFFSETy_EN=1), the read data is signed.

--If none of the four offsets for this channel is enabled, the read data is unsigned.

The figures below illustrate the alignment methods for signed and unsigned data.

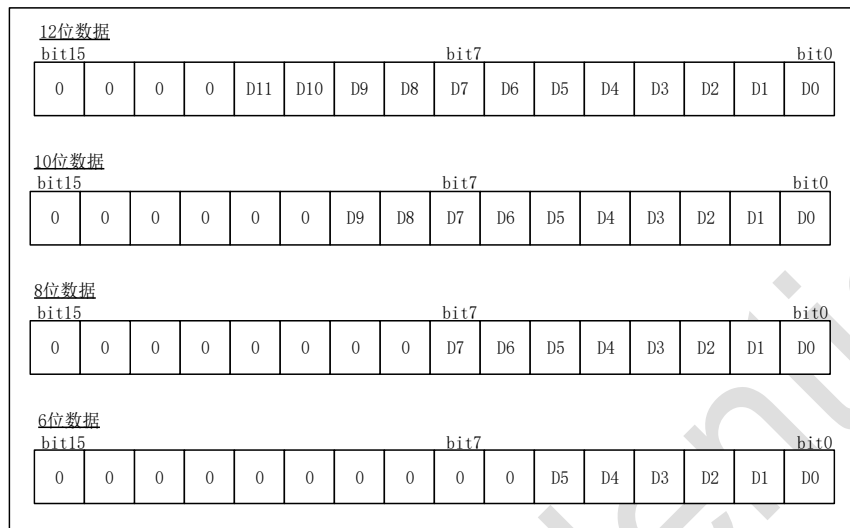


Figure 13-14 Right-aligned (offset disabled, unsigned number)

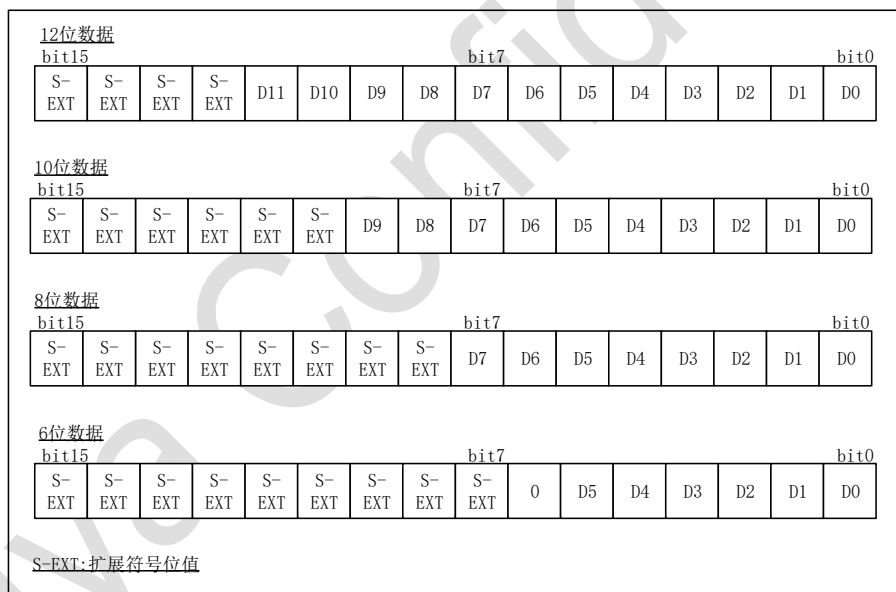


Figure 13-15 Right-aligned (offset enabled, signed number)

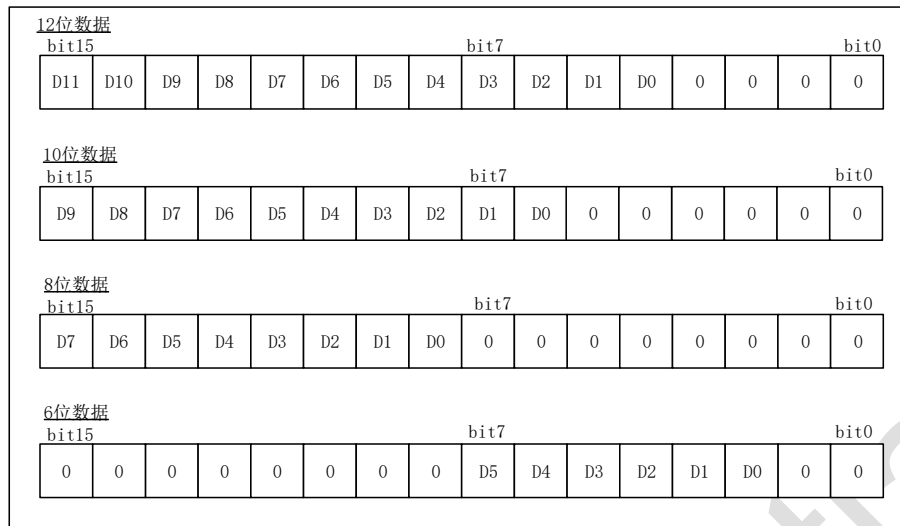


Figure 13-16 Left-aligned (offset disabled, unsigned number)

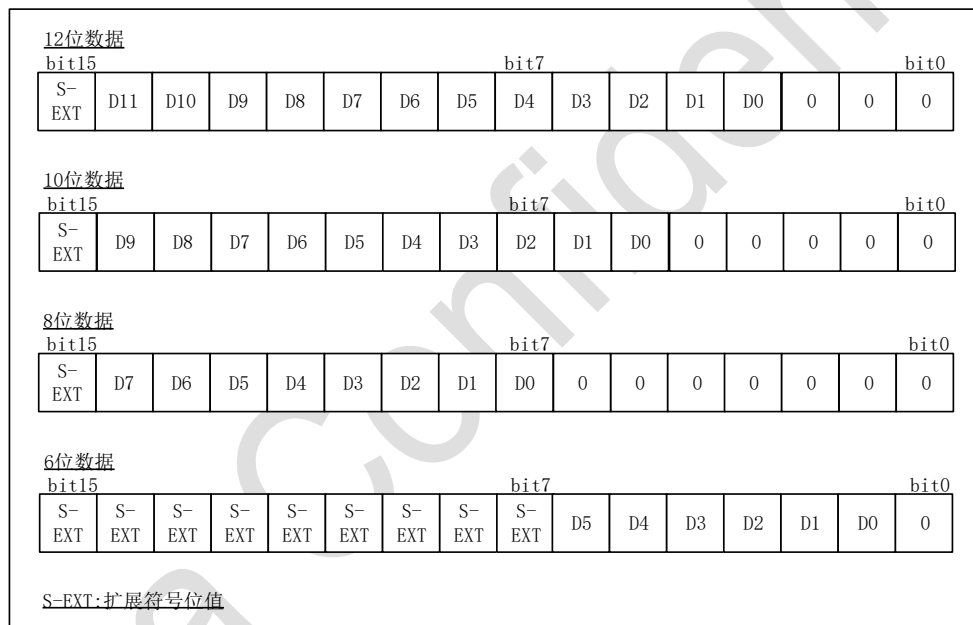


Figure 13-17 Left-aligned (offset enabled, signed number)

13.3.22.3. Gain Compensation (GCOMP, ADC_GCOMP)

When the ADC_CFGR2 register GCOMP bit is set, all converted data is gain compensated. After each conversion, the data will be calculated using the following formula.

$$\text{DATA} = \text{DATA}(\text{adc result}) \times (\text{GCOMP\_COEFF}) / 4096$$

Since GCOMP_COEFF ranges from 0-16383, the actual gain compensation factor ranges from 0-3.999756.

Before storing the result data in the ADC_DATA or ADC_JDRy registers, the LSB-1 value is rounded, and the error is minimized.

Gain compensation is also effective for oversampling. When gain compensation is used in the oversampling mode, the gain calculation is performed after the accumulation and right shift operation to minimize power consumption (the gain calculation is only done once, not every transition).

13.3.22.4. Offset Compensation (SATEN, OFFSETPOS)

When the SATEN bit is set in the ADC_OFRy register during the offset operation, the data is unsigned. All offset data is saturated at 0x000 (at 12-bit resolution). When the OFFSETPOS bit is set, the offset direction is positive and the data is saturated at 0xFF (at 12-bit resolution). In 8-bit resolution, the data is saturated at 0x00 and 0xFF, respectively.

After offset and gain compensation, an analog watchdog comparison is performed on the unsigned number. For proper watchdog operation, offset-compensated data must be in unsigned format (the SATEN bit in the ADC_OFRy register is set to 1).

13.3.22.5. ADC Overload (OVR, OVRMOD)

The ADC Overload Flag (OVR) indicates a buffer overflow event, which occurs when converted data is not read by the CPU or DMA in time, and another conversion result becomes valid.

If EOC is still '1' and a new conversion has completed, the CPU will set the OVR flag in the ADC_ISR register, indicating an ADC overload. An interrupt can be generated if the OVRIE bit is set in the ADC_IER register.

When an overload event occurs, the ADC continues operating and converting unless the software decides to stop and reset the sequence conversion. The software can set ADSTP to 1 in the ADC_CR register to stop ADC conversion. The OVR flag can be cleared by writing 1 in software.

When an overload event occurs, the OVRMOD bit in the ADC_CFGR register can be configured to determine whether the data in the ADC data register is retained or overwritten:

■ OVRMOD=0

- During an overload event, the value in the data register is not overwritten: the previous data is retained, and the new conversion data is discarded. If OVR remains at 1, further conversions can be performed, but the resulting data is discarded.

■ OVRMOD=1

- During an overload event, the data register is overwritten with the latest conversion result, and previously unread data is lost. If OVR remains 1, subsequent conversions are executed, and the ADC_DR register is overwritten with new conversion results, always holding the latest converted value.

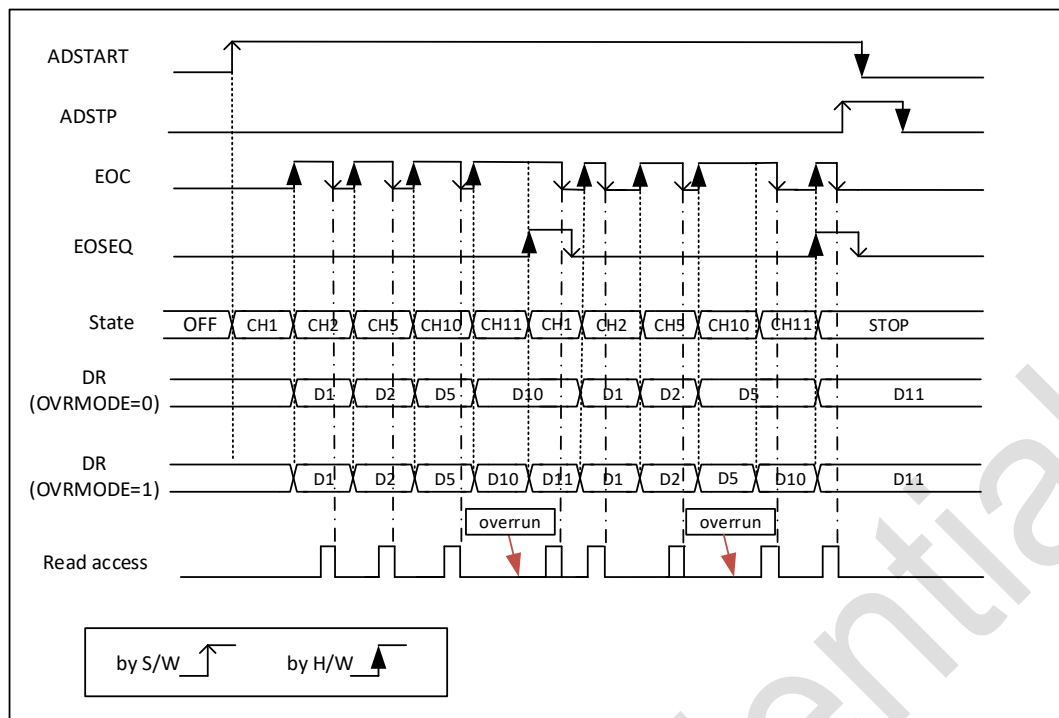


Figure 13-18 Overload

13.3.22.6. Non-DMA mode

If the conversion of the ADC is slow enough, the conversion sequence can be controlled by software. In this case, the software applies the EOC flag and its associated interrupt to process each translation data. At the end of each conversion, at the EOC position bit in the ADC_ISR register, the converted value of the ADC_DR register can be read at this time. The OVRMOD bit in the ADC_CFGR register may be configured to 0 to manage overload events.

13.3.22.7. Non-overload non-DMA mode

There are applications that convert one or more channels without needing to read each conversion result. In this case, the OVRMOD bit must be set to 1, and the software should ignore the OVR flag. When OVRMOD=1, an overload event does not prevent the ADC from continuing conversions, and the data in the ADC_DR register always remains the last converted data.

13.3.22.8. DMA mode

Since the conversion result data for all channels is stored in a single data register, using DMA is more efficient when converting more than one channel. This avoids losing the conversion results stored in the ADC_DR register. When DMA mode is enabled (DMAEN=1 in the ADC_CFGR register), a DMA request is generated at the end of each conversion. This allows the translation data in the ADC_DR register to be transferred to the target address specified by the software.

However, if an overload occurs (OVR=1) because the DMA cannot service the DMA request in time, the ADC stops generating DMA requests, and the corresponding results of new conversions are no longer transmitted by the DMA (transmission resumes when OVR=0). It can also be considered that all data transferred to RAM is valid (because invalid data is never transferred again).

Depending on the configuration of the OVRMOD bit, the data in the ADC_DR register can be selected as either Hold or Override.

The DMA transfer request is blocked until the software clears the OVR bit.

There are two different DMA modes, which depend on the configuration of the DMACFG bits in the ADC_CFGR register:

- **DMA One-Shot Mode (DMACFG=0)**

This mode can be selected when the DMA is programmed to transfer a fixed length of data.

- **DMA Circular Mode (DMACFG=1)**

This mode can be selected when the DMA is programmed in circular mode.

DMA one shot mode (DMACFG=0)

In this mode, the ADC generates a DMA request each time the converted data is valid. Once the DMA has reached the last DMA transfer, the ADC stops generating DMA requests, even if ADC conversions have restarted. (When a DMA_EOT interrupt occurs, the next ADC conversion may have already started)

When the DMA transfer is complete (all transfers configured in the DMA controller have been completed):

- Contents of ADC data register freeze
- Any ongoing conversion is terminated. And the result value is discarded
- No new DMA requests are issued to the DMA controller. This method avoids an ADC overload error if there is still an ADC conversion start
- ADC scan sequence stops and resets
- DMA Stop

DMA continuous mode (DMACFG = 1)

In this mode, even if the DMA reaches the last DMA transfer, the ADC generates a DMA request each time the converted data is valid. This allows the DMA to be configured in a cyclic mode to process a continuous analog input data stream.

13.3.23. **ADC analog watchdog (AWDEN, JAWDEN, AWDSGL, AWDCH, AWD_TR)**

The analog watchdog function is enabled by setting AWDEN in the ADC_CFGR register. It is used to monitor that either one selected channel or all enabled channels.

13.3.23.1. **AWD flag and interrupt**

If the analog voltage conversion by the ADC is below the low threshold LT or above the high threshold HT, the AWD analog watchdog status bit is set. Thresholds are programmed into the ADC_TR.HT and ADC_TR.LT registers with up to 12-bit valid data. An interrupt can be enabled by setting the AWDIE bit in the ADC_IER register. The AWD flag is cleared by software by writing 1 to it. When converting a data with a resolution of less than 12-bit (according to bits RES [1:0]), the LSB of the programmed thresholds must be kept cleared because the internal comparison is always performed on the full 12-bit raw converted data (left aligned).

The table below describes the comparisons performed for all possible resolutions of the analog watchdog.

Table 13-5 Analog watchdog comparison

Resolution bits (RES)	Analog watchdog comparison target		Note
	Raw converted data, left aligned	Thresholds	
00: 12-bit	DATA[11:0]	LT[11:0] and HT[11:0]	
01: 10-bit	DATA[11:2],00	LT[11:0] and HT[11:0]	The user must configure LT[1:0] and HT[1:0] to "00"
10: 8-bit	DATA[11:4],0000	LT[11:0] and HT[11:0]	The user must configure LT[3:0] and HT[3:0] to "0000"
11: 6-bit	DATA[11:6],000000	LT[11:0] and HT[11:0]	The user must configure LT[5:0] and HT[5:0] to "000000"

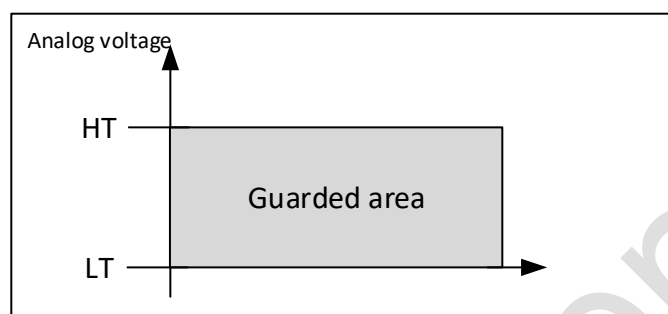


Figure 13-19 Simulated watchdog protection area

Table 13-6 Analog watchdog channel selection

Watchdog monitoring channel	AWDSGL bit	AWDEN bit	JAWDEN bit
None	x	0	0
All injected channels	0	0	1
All regular channels	0	1	0
All injected and regular channels	0	1	1
Single injection channel	1	0	1
Single regular channel	1	1	0
Single injection or regular channel	1	1	1

13.3.23.2. ADC_AWD_OUT output signal generation

The analog watchdog is associated with an internal hardware signal, ADC_AWD_OUT. This signal is directly connected to the ETR (external trigger) input of certain on-chip timers.

When the associated analog watchdog is enabled, ADC_AWD_OUT activates:

- ADC_AWD_OUT is set when a protected conversion exceeds the programmed threshold.
- ADC_AWD_OUT is reset when the next protected conversion completes and the value is within the programmed threshold range (but if subsequent protected conversions still exceed the threshold, the signal remains at 1).
- ADC_AWD_OUT is also reset when the ADC is disabled (by setting ADDIS=1). Note that stopping regular or injected conversions (by setting ADSTP=1 or JADSTP=1) does not affect ADC_AWD_OUT generation.

Note: The AWD flag is set by hardware and cleared by software; the AWD flag does not affect ADC_AWD_OUT generation (e.g., if the software does not clear the flag, ADC_AWD_OUT may toggle even if the AWD flag remains at 1).

The ADC_AWD_OUT signal is generated by the PCLK domain.

AWD comparison is performed at the end of each ADC conversion. The same applies to regular and injected ADC_AWD_OUT.

The following timing diagram shows the case when all channels are selected:

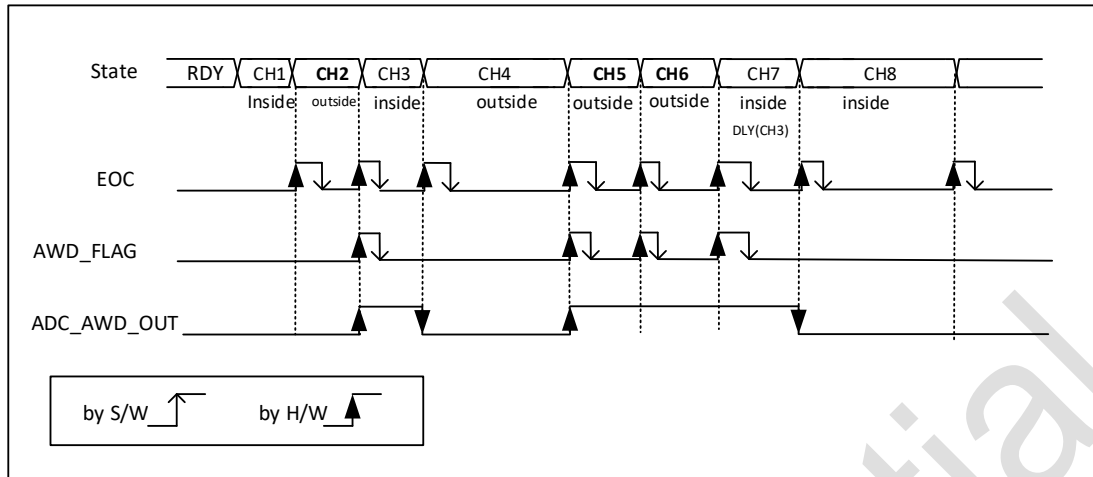


Figure 13-20 Select all channels to simulate watchdog

The following timing diagram shows the case when the AWD_FLAG is not cleared by software:

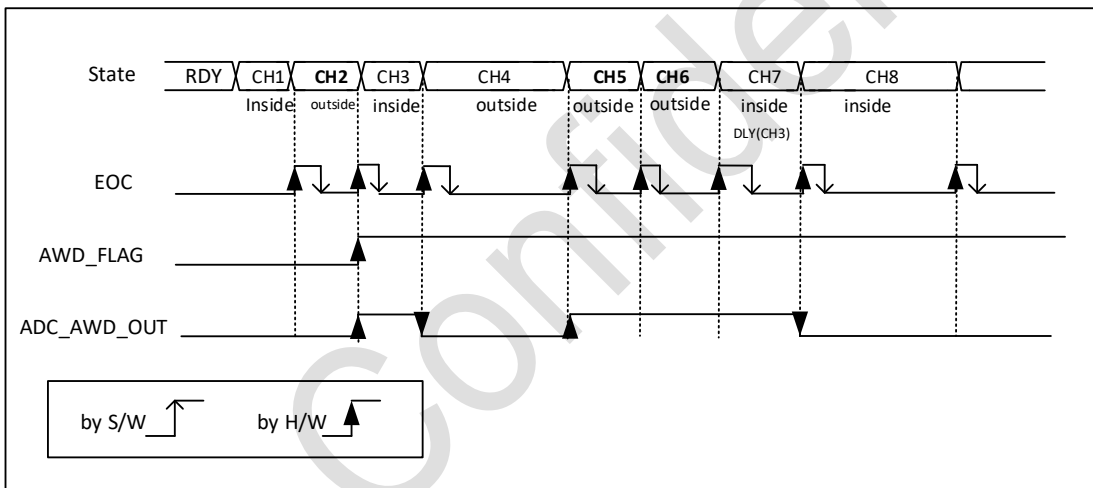


Figure 13-21 Software does not clear AWDx

13.3.23.3. Analog watchdog with gain and offset compensation

When gain and offset compensation are enabled, the analog watchdog compares the threshold after data compensation.

Note: When offset compensation is enabled (OFFSETy_EN in ADC_OFrY register is set to 1), data overflow or underflow may cause incorrect watchdog results. When saturation is enabled (SATEN set to 1 in ADC_OFrY), the watchdog provides correct results. However, this makes it impossible to use signed data formats.

13.3.24. ADC Dynamic Low-Power Feature

13.3.24.1. Auto-Delayed Conversion

Auto-delayed conversion is useful for simplifying software and optimizing application performance, as programs running at low frequencies may encounter ADC overflow risks.

When AUTDLY is set to 1 in the ADC_CFGR register, a new conversion can only start once all previous data has been processed:

- For regular conversions: once the ADC_DR register is read or the EOC bit is cleared;
- For injected conversions: when the JEOS bit is cleared;

Note: After an injected conversion ends, if JEOS is not cleared and a regular trigger occurs, the regular trigger is ignored.

This way, the ADC speed is automatically adjusted to match the system's data reading speed.

A delay is inserted after each regular conversion (regardless of DISCEN=0 or 1) and each injected conversion sequence (regardless of JDISCEN=0 or 1).

Note: No delay is inserted between injected conversion sequences, but it is inserted after the last sequence.

During conversion, hardware trigger events occurring during this delay (for the same group of conversions) are ignored.

Note: For software triggers, conversions can still be restarted during this delay by setting ADSTART or JADSTART (subject to the aforementioned principles: EOC cleared or data read for regular conversions, JEOS cleared for injected conversions; if conditions are not met, hardware ensures ADSTART and JADSTART cannot be set): software must read data before starting a new conversion to avoid data overwrite loss.

No delay is inserted between conversions of different groups (injected conversions start immediately after regular conversions, and vice versa):

- If an injected trigger occurs during the auto-delay of a regular conversion, the injected conversion starts immediately.
- Once the injected sequence completes, the ADC waits for the previous regular conversion's delay (if not ended) before starting a new regular conversion.

In auto-injected mode (JAUTO=1), the behavior differs slightly—new regular conversions can only start once the auto-delay of the previous injected conversion sequence ends (when JEOS is cleared). This is to ensure that the software can read all data for a given sequence before starting a new sequence.

To stop conversion in continuous auto-injection mode and auto-delay mode (JAUTO=1, CONT=1, AUTDLY=1), follow these steps:

1. Wait until JEOS=1 (no further conversion restart)
2. Clear JEOS
3. Set ADSTP = 1
4. Read the regular data.

In AUTDLY mode, if a hardware trigger event for regular conversion occurs during an already ongoing regular sequence or during the delay following the last regular conversion of the sequence, the event is ignored. However, if it occurs after this delay, it is considered pending, even if it occurs during the injection sequence immediately following the delay. The conversion then starts after the delay of the injection sequence ends.

In AUTDLY mode, if a hardware trigger event for injection occurs during an already ongoing injection sequence or during the delay following the last injection conversion of the sequence, the event is ignored.

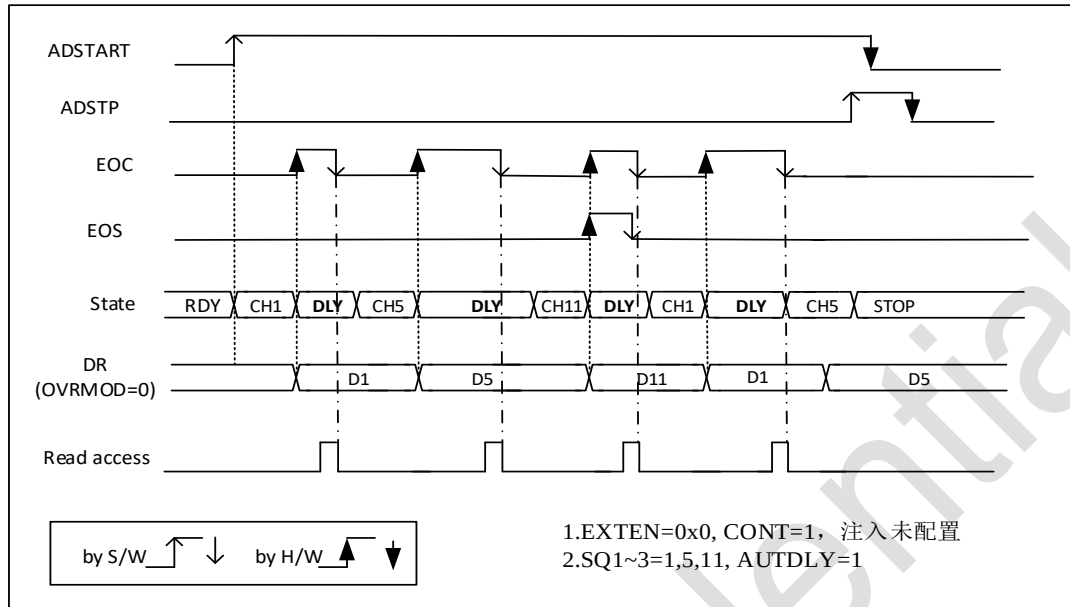


Figure 13-22 AUTDLY mode, regular conversion in continuous mode, software-triggered

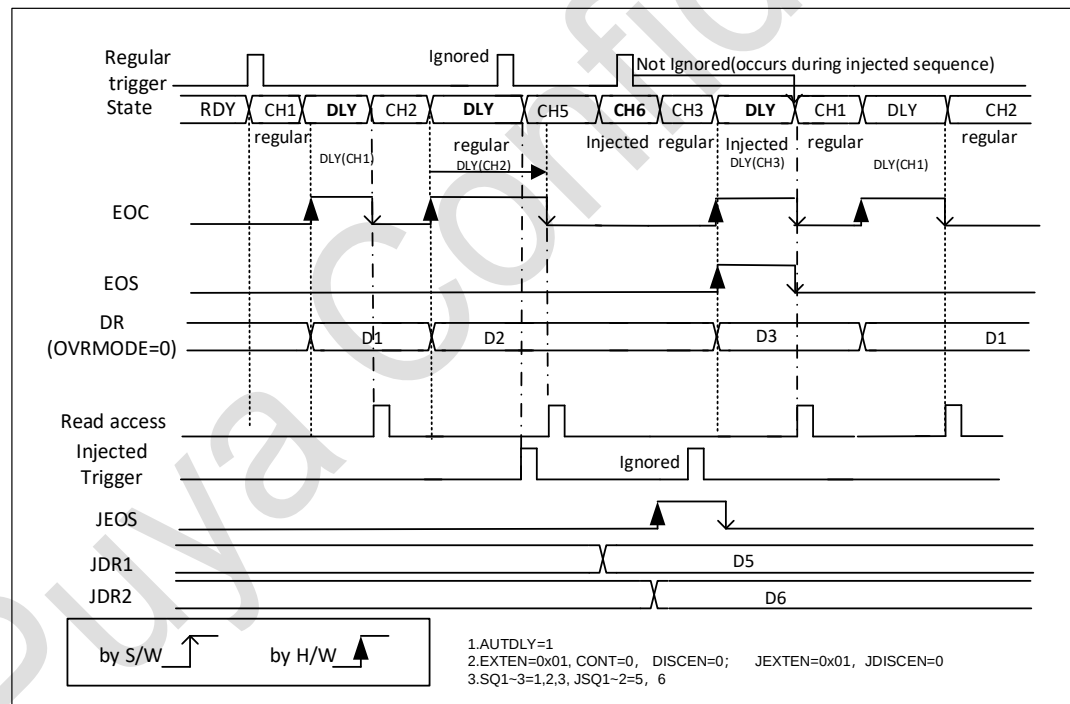


Figure 13-23 AUTDLY mode, regular conversion interrupted by injection, hardware-triggered

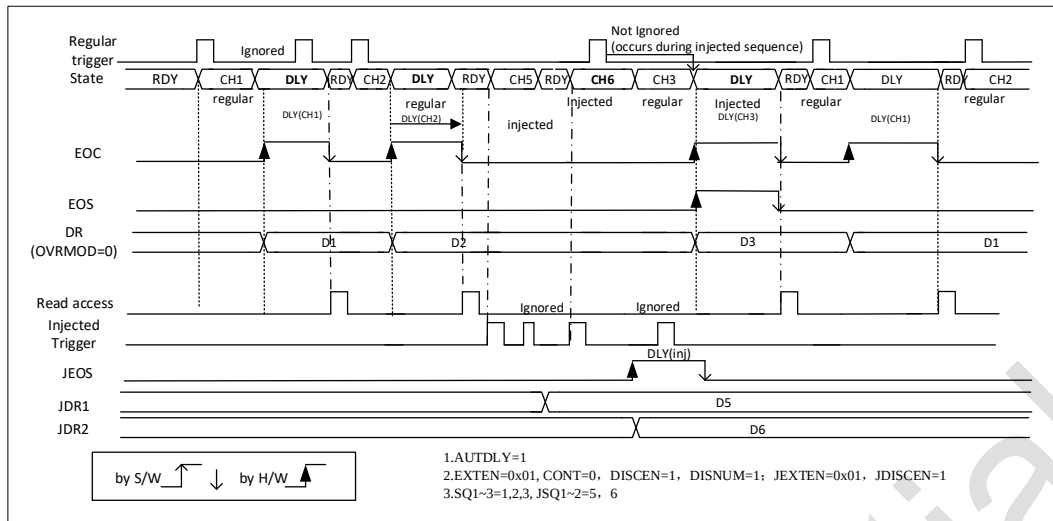


Figure 13-24 AUTODLY mode, regular conversion interrupted by injection, hardware-triggered, non-continuous conversion

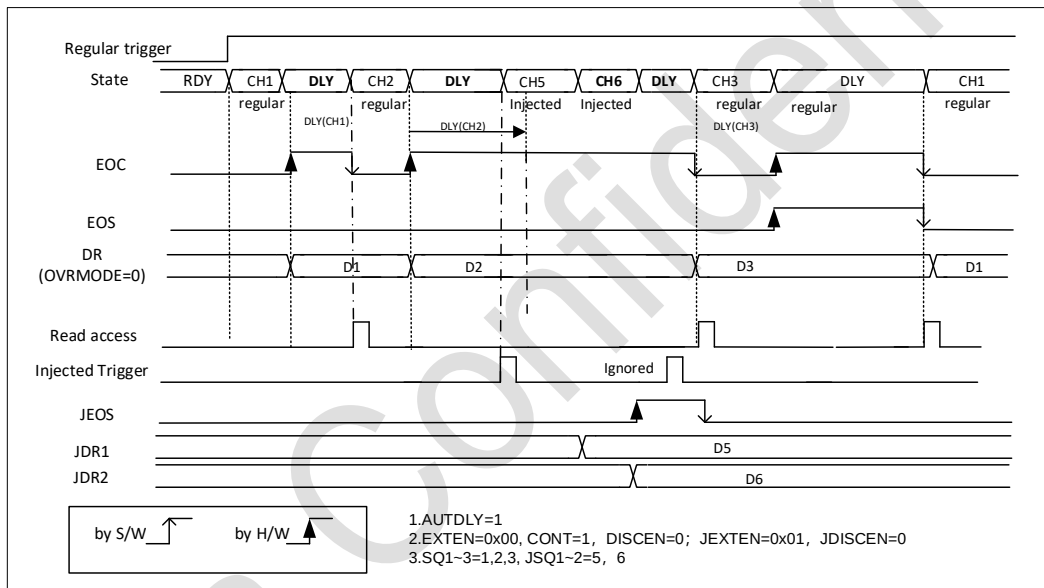


Figure 13-25 AUTODLY mode, regular conversion interrupted by injection, regular software-triggered, injection hardware-triggered, continuous conversion

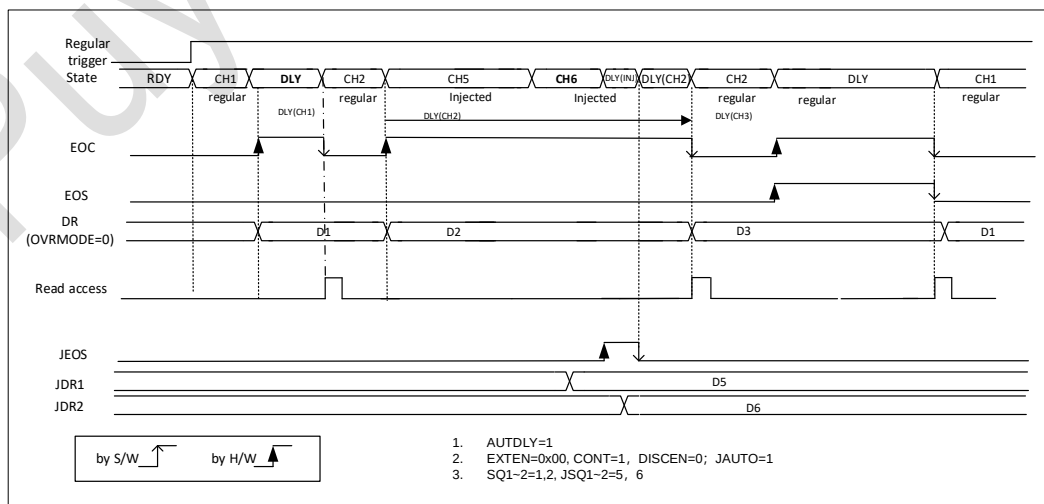


Figure 13-26 AUTODLY mode, auto-injection mode

13.3.25. ADC oversampling

The oversampling unit performs data preprocessing to reduce CPU load. It can handle multiple conversions and average them into a single data (increasing the data width up to 16 bits).

The oversampler provides the following formula, where N and M can be adjusted:

$$Result = \frac{1}{M} \times \sum_{n=0}^{n=N-1} Conversion(t_n)$$

The oversampler allows the following functions to be performed by hardware: averaging, reducing data rate, improving signal-to-noise ratio, basic filtering.

The oversampling ratio N is defined by the OVFS[2:0] bits in the ADC_CFGR2 register and can be adjusted from 2x to 256x. The division coefficient M consists of a right bit shift of up to 8 bits and is defined using the OVSS [3: 0] bit in the ADC_CFGR2 register.

The summing unit can produce a result of up to 20 bits (256 x 12-bit result), which is first shifted to the right. It is then truncated to 16 least significant bits before final transfer to the ADC_DR data register, rounding the least significant bits left by the shift to the nearest value.

Note: If the shifted intermediate result exceeds 16 bits, the result is truncated as is without saturation.

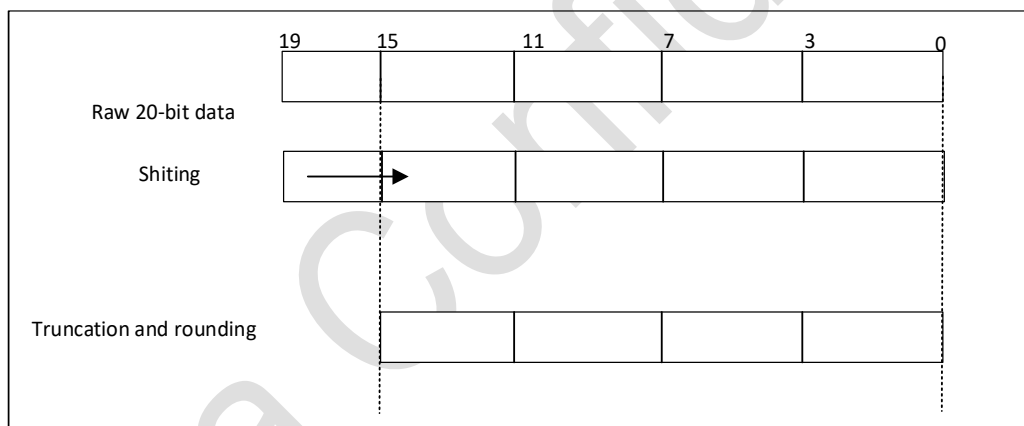


Figure 13-27 20bit truncated to 16bit

The figure below gives a numerical example of processing, from the raw 20-bit accumulated data to the final 16-bit result.

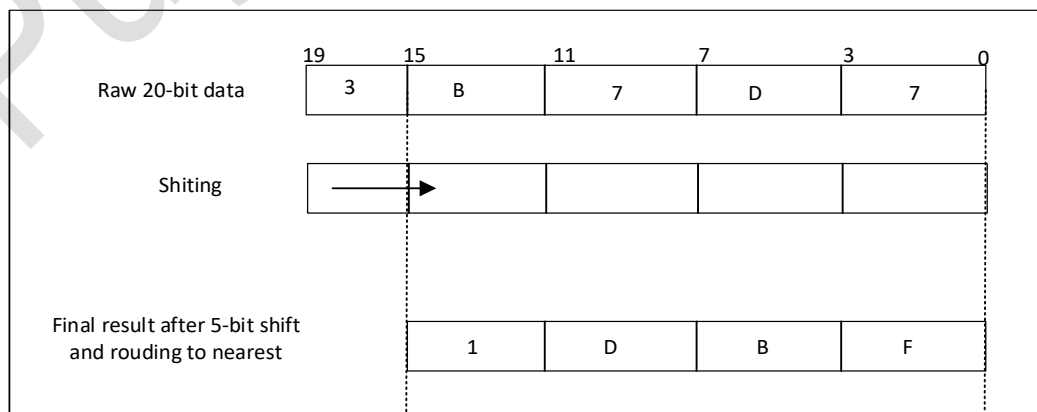


Figure 13-28 Numerical example with 5-bit shift and rounding

The table below gives the data formats for various N and M combinations where the raw converted data is 0xFFFF.

Table 13-7 Maximum Output Results vs. N and M (gray cells indicate truncation)

Over-sampling rate	Maximum raw data	No shift OVSS=00 00	1-bit shift OVSS=00 01	2-bit shift OVSS=00 10	3-bit shift OVSS=00 11	4-bit shift OVSS=01 00	5-bit shift OVSS=01 01	6-bit shift OVSS=01 10	7-bit shift OVSS=01 11	8-bit shift OVSS=10 00
2x	0x1FFE	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040	0x020
4x	0x3FFC	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040
8x	0x7FF8	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080
16x	0xFFFF	0x FFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100
32x	0x1FFE0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200
64x	0x3FFC0	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400
128x	0x7FF80	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800
256x	0xFFFF0	0xFF00	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF

There is no change in the transition timing in oversampling mode: the sampling times remain equal throughout the oversampling sequence. Every N transitions provide a new data, delayed by $N \times T_{CONV} = N \times (t_{SMPL} + t_{SAR})$ equivalent. The flags are set as follows:

- Set end of sampling phase flag (EOSMP) after each sampling phase
- End of Transition (EOC) occurs every N transitions when oversampling results are available
- The end of sequence (EOS) occurs after the completion of the oversampled data sequence (i.e. after the $N \times$ total sequence length conversion)

13.3.25.1. Supported ADC operating modes when oversampling (single ADC mode)

In oversampling mode, most ADC operating modes remain the same:

- Single mode/continuous mode conversion
- Initiate ADC conversion via software or external trigger
- Stop ADC during transition (abort)
- Read data via CPU/DMA
- Low Power Mode (AUTDLY)
- Programmable resolution: In this case, the reduced converted values (according to RES[1:0] bits in ADC_CFGR register) are accumulated, truncated, rounded, and shifted in the same way as for 12-bit conversion.

Note: Data alignment is not available when using oversampled data. The ALIGN bit in ADC_CFGR is ignored and the data is always provided right-aligned. Offset correction is not supported in oversampling mode. When the ROVSE and/or JOVSE bits are set, the value of the OFFSETy_EN bit in the ADC_OFRy register is ignored (considered a reset).

13.3.25.2. Analog watchdog

The analog watchdog function is maintained (AWDSGL, AWDEN, and JAWDEN bits), but with the following differences:

Ignore the RES [1: 0] bit and always use the full 12-bit values HT [11: 0] and LT [11: 0] for comparison. Comparison is performed on the most significant 12 bits of the 16-bit oversampling result ADC_DR [15: 4]

Note: Caution must be taken when using high shift values, which reduces the comparison range. For example, if the oversampling result is shifted by 4 bits, resulting in 12 bits of data right-aligned, a valid analog watchdog comparison can only be performed on 8 bits. The comparison is made between ADC_DR[11:4] and HT[0:7]/LT[0:7], and HT[11:8]/LT[11:8] must be reset.

13.3.25.3. Trigger Mode

The averager can also be used for basic filtering. While not a very powerful filter (slow roll-off and limited stopband attenuation), it can be used as a notch filter to suppress constant parasitic frequencies (usually from mains or switched mode power supplies). For this purpose, a specific discontinuous mode can be enabled using the TROVS bit in ADC_CFGR2 to allow user-defined oversampling frequency, independent of the conversion time itself.

The figure below shows how to respond to trigger-initiated conversion in discontinuous mode. If the TROVS bit is set, the contents of the DISCEN bit are ignored and treated as 1.

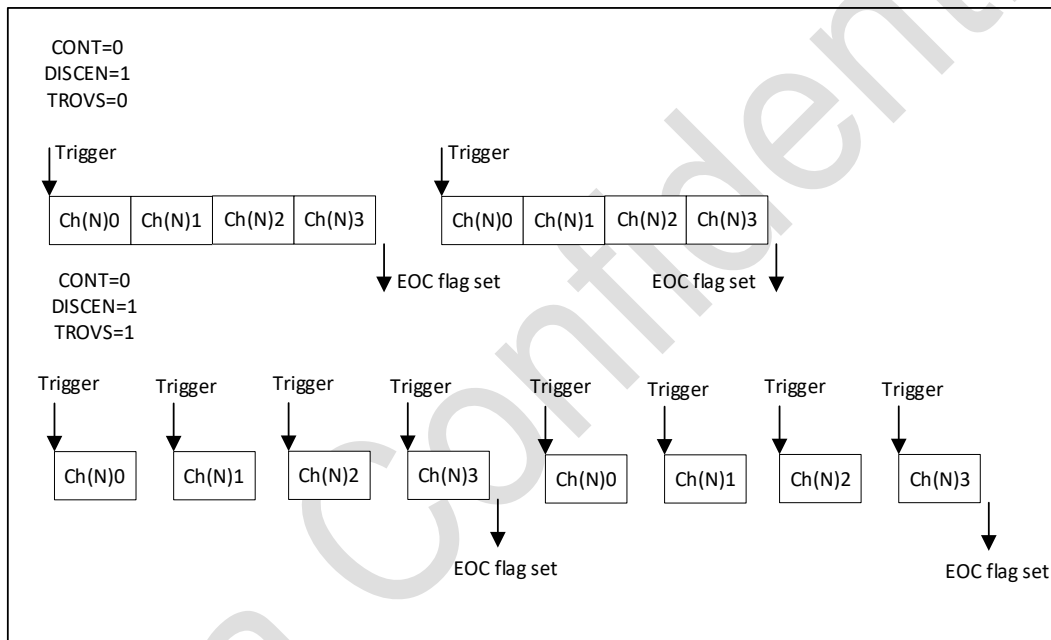


Figure 13-29 Triggered regular oversampling mode (TROVS=1)

13.3.25.4. Injection and regular sequence management during oversampling

In oversampling mode, there may be differences between injected sequence and regular sequence behavior. If both sequences must be used simultaneously, oversampling can be enabled for them, but with some limitations (this is related to the single accumulation unit).

13.3.25.5. Oversample regular channels only

The regular oversampling pattern bit ROVSM defines how a regular oversampling sequence is restored if the sequence is interrupted by the injected transition:

- In continuous mode, accumulation restarts from the last valid data (prior to the abort request issued due to injection trigger). This ensures oversampling will be completed regardless of injection frequency (assuming at least one regular conversion can be completed between triggers);
- In recovery mode, accumulation restarts from 0 (discarding previous conversion results). This mode guarantees all data used for oversampling undergoes back-to-back conversions within a single time slot. Note that the injection trigger period must exceed the oversampling duration. If

this condition is not met, oversampling cannot be completed and the rule sequence will be disabled.

The following figure shows an example of 4x oversampling ratio.

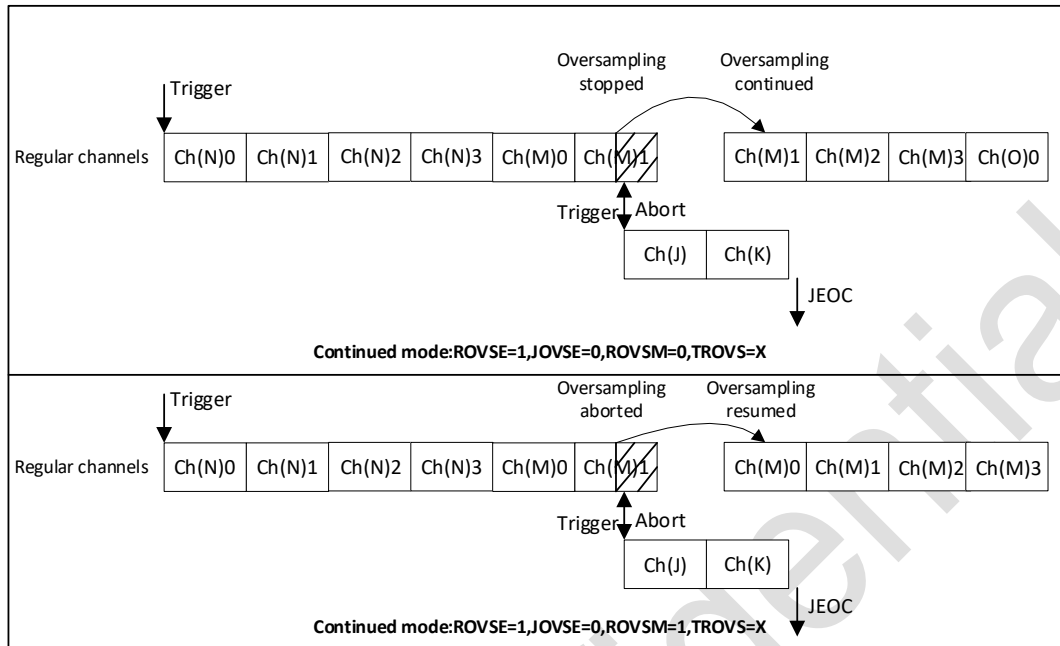


Figure 13-30 Rule oversampling mode (4x oversampling ratio)

13.3.25.6. Oversample only the injection channel

The injection oversampling mode bit JOVSE is dedicated to enabling oversampling for conversions in the injection sequence.

13.3.25.7. Regular channel and injection channel oversampling

The ROVSE and JOVSE bits can be set at the same time. In this case, the regular oversampling mode is forced to recovery mode (ignoring the ROVSM bit), as shown in the following figure

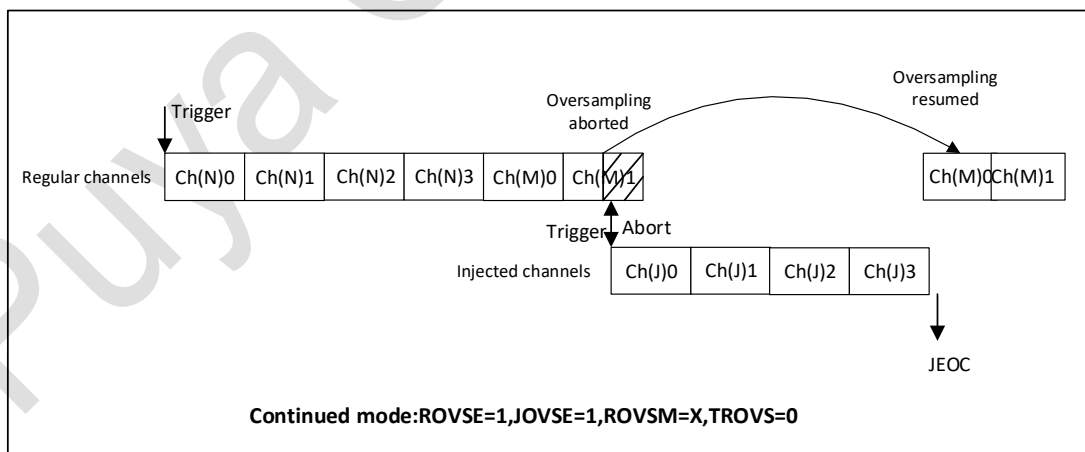


Figure 13-31 Oversampling mode using both regular and injected conversions

13.3.25.8. Regular channel oversampling interrupted by injection trigger

Injected conversions can be executed in triggered rule mode. In this case, the injection mode oversampling mode must be disabled, and the ROVSM bit (forced recovery mode) must be ignored. The JOVSE bit must be reset (software needs to configure it to 0). If configured to 1, the hardware will clear JOVSE in this case). The behavior is shown in the image below.

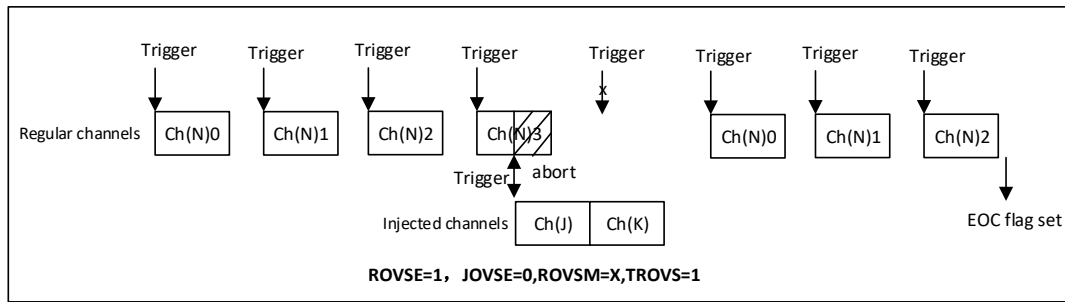


Figure 13-32 Triggered rule oversampling interrupted by injection

13.3.25.9. Automatic injection mode

Automatically injected sequences can be oversampled and all conversion results stored in registers to save DMA resources. This mode is only available when both regular and injection oversampling are enabled: JAUTO = 1, ROVSE = 1, and JOVSE = 1, other combinations are not supported. The ROVSM bit is ignored in auto-injection mode. The following diagram shows the sequence of transformations.

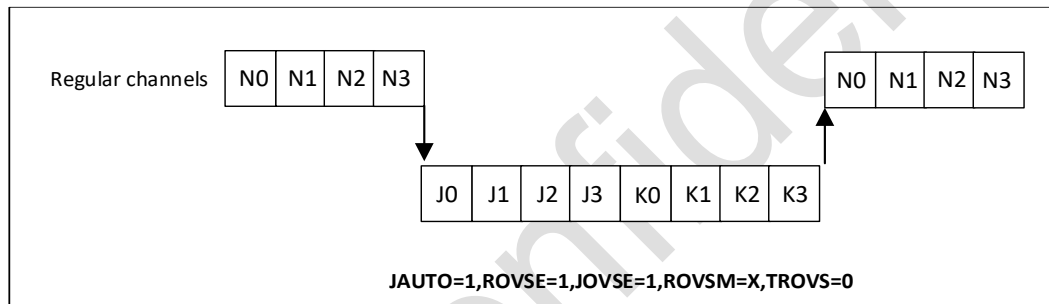


Figure 13-33 Oversampling in Auto-Injection Mode

The trigger mode can also be enabled using the TROVS bit. In this case, the ADC must be configured as follows: JAUTO = 1, DISCEN = 0, JDISCEN = 0, ROVSE = 1, JOVSE = 1, and TROVSE = 1. (This configuration prohibits combination with CONT (enabling CONT will cause malfunction)).

13.3.25.10. Mode Combination Summary

Table 13-8 Oversampler Mode Section

Regular over-sampling ROVSE	Injection Over-sampling JOVSE	Oversampling Mode ROVSM; 0=Continuous 1=Recovery	Trigger Rule Mode TROVS	Description
1	0	0	0	Regular continuous mode
1	0	0	1	-
1	0	1	0	Rule recovery model
1	0	1	1	Trigger Rule Recovery Mode
1	1	0	X	-
1	1	1	0	Injection and rule recovery models
1	1	1	1	-
0	1	X	X	Injection oversampling

13.3.26. Temperature sensor

The temperature sensor can be used to measure the junction temperature (T_J) of the device.

The temperature sensor is internally connected to an ADC input channel, which is used to convert the sensor's output voltage into a digital value. The sampling time for the temperature sensor must be greater than the minimum stabilization time specified in the datasheet. When not in use, the sensor can be put in power down mode.

The temperature sensor's output voltage changes linearly with temperature, but due to manufacturing variations, the offset of the temperature curve may differ between chips. To improve the accuracy of the temperature sensor, calibration values are individually measured for each part during production test and stored in the system memory area.

The internal voltage reference (V_{REFINT}) provides a stable voltage output to the ADC and comparators.

Note: The ADC_CCR.TSEN=1 bit must be set to activate the internal temperature sensor channel, and the ADC_CCR.VREFINT_EN=1 bit must be set to activate the internal reference voltage.

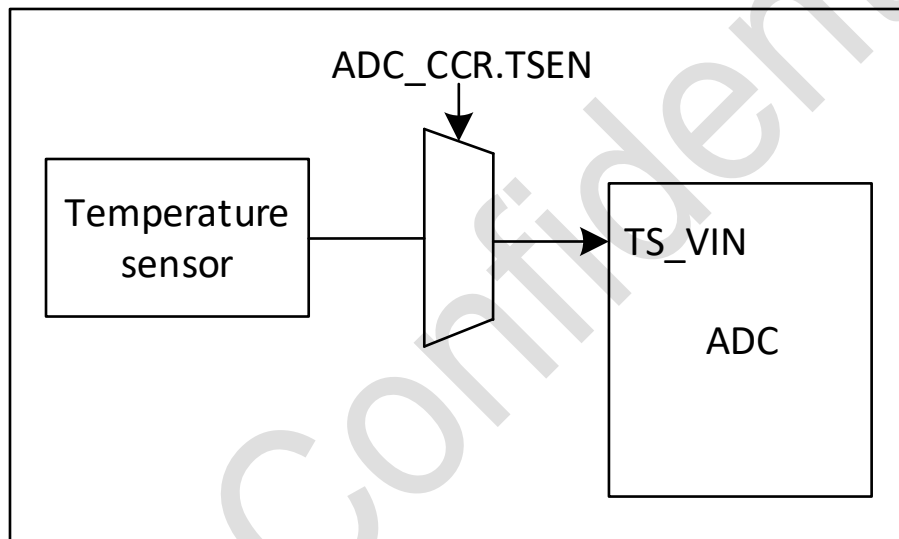


Figure 13-34 Temperature Sensor Block Diagram

13.3.26.1. Reading the temperature

To use the temperature sensor, perform the following operations:

- Select the ADC_VINP[27] input channel
- Select a sampling time greater than the minimum sampling time specified in the datasheet
- Set the TSEN bit in the ADC_CCR register to wake up the temperature sensor from power-down mode
- Set ADEN to enable the ADC, then set ADSTART (or use external trigger) to start ADC conversion
- Read the VTS conversion data from the ADC_DR register

Calculate the actual temperature using the following formula:

$$Temperature(in\ ^\circ C) = \frac{105^\circ C - 30^\circ C}{TS_{CAL2} - TS_{CAL1}} \times (TS_{DATA} - TS_{CAL1}) + 30^\circ C$$

TSCAL2 represents the calibration value of the temperature sensor at 105°C (read from Flash address 0x1FFF 1E24)

TSCAL1 represents the calibration value of the temperature sensor at 30°C (read from Flash address 0x1FFF 1E20)

TS_DATA is the actual temperature sensor output value converted by ADC

Note: There is a startup time required for the sensor to correctly output VTS after waking from power-down mode, and the ADC also has a startup time after power-on. To reduce this delay, set both the ADEN and TSEN bits simultaneously.

13.3.27. Monitoring the internal reference voltage

The internal reference voltage can be monitored to obtain a reference value for evaluating the ADC V_{REF+} voltage.

The internal reference voltage is internally connected to input channel ADC_VINP[28].

The sampling time for this channel must be greater than the settling time specified in the product datasheet.

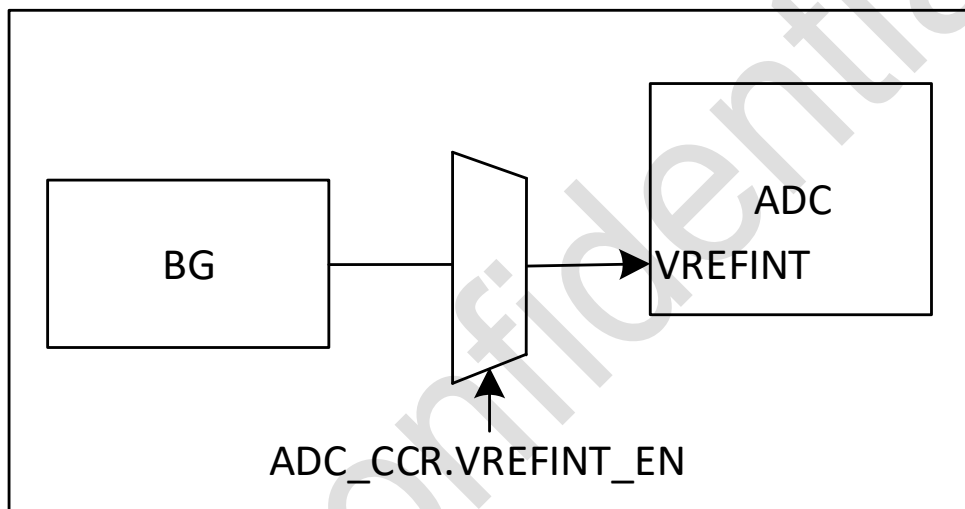


Figure 13-35 Temperature Sensor Block Diagram

Note: The ADC_CCR.VREFINT_EN=1 bit must be set to activate the internal reference voltage.

The following formula can give the voltage value of the true V_{CC} :

$$V_{REFINT} = 1.2V = \frac{ADC\_DATAx}{4095} \times V_{CC}$$

V_{REFINT} is fixed at 1.2 V;

ADC_DATA is the conversion data of the V_{REFINT} channel.

Convert the relative value of channel voltage measured by an ADC into absolute voltage

value

The ADC gives a digital value based on the ratio of the voltage on the analog power input and conversion channel. Most applications need to convert this ratio into a voltage value. For the case where the V_{CC} is known and the ADC conversion value is right-aligned, this absolute voltage value can be obtained by the following formula:

$$V_{CHANNEL} = \frac{ADC\_DATAx}{4095} \times V_{CC}$$

$V_{CHANNEL}$ is the channel voltage;

ADC_DATA is the conversion data in ADC_DR;

4095 is represented as 12 bits.

13.3.28. Monitoring the power supply

Because the V_{CC} voltage may be inaccurate, the V_{CC} pin needs to be internally connected to the bridge distributor to ensure the correct operation of the ADC.

The bridge auto-enable connects $V_{CC}/3$ to the ADC_VINP[29] input channel.

13.4. ADC interrupts

An interrupt can be generated by any of the following events:

- ADC ready (ADRDY flag)
- End of any rule transition (EOC flag)
- End of rule sequence conversion (EOS flag)
- End of any one injection transition (JEOC flag)
- End of injection sequence conversion (JEOS flag)
- Analog watchdog detection occurred (AWD flag)
- End of sampling phase (EOSMP flag)
- Data overrun occurred (OVR flag)
- Calibration completed (EOCAL flag)

A separate interrupt enable bit is used to flexibly set the ADC interrupt.

Table 13-9 ADC interrupts

Interrupt event	Event flag	Enable control bit
ADC Ready	ADRDY	ADRDYIE
End of rule conversion	EOC	EOCIE
End of rule sequence conversion	EOS	EOSIE
End of injection conversion	JEOC	JEOCIE
End of injection sequence conversion	JEOS	JEOSIE
Analog watchdog status bit is set	AWD	AWDIE
End of sampling phase	EOSMP	EOSMPIE
Overrun	OVR	OVRIE
Calibration Complete	EOCAL	EOCALIE

13.5. ADC registers

13.5.1. ADC Interrupt Status Register (ADC_ISR)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	EOCAL	Res.	Res.	Res.	AWD	JEOS	JEOC	OVR	EOS	EOC	EOSMP	ADRDY
				RC_W1				RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	EOCAL	RC_W1	0	Calibration Completion Mark Set by hardware when calibration completes, cleared by software writing 1. 0: Calibration not started or not completed (or flag event has been acknowledged and cleared by software) 1: Calibration completed
10:8	Reserved	-	-	Reserved
7	AWD	RC_W1	0	Analog watchdog flag

				Set by hardware when the converted voltage is below the value programmed in ADC_TR.LT register or above ADC_TR.HT register. This bit is cleared by writing 1 0: No analog watchdog event occurred (or flag event has been acknowledged and cleared by software) 1: Analog watchdog event occurred
6	JEOS	RC_W1	0	Injected channel sequence end flag This bit is set to 1 by hardware when all channels in the injected sequence have completed conversion. Cleared by software writing 1. 0: Injected conversion sequence not completed (or flag event has been acknowledged and cleared by software) 1: Injected conversion sequence completed
5	JEOC	RC_W1	0	End of injected channel conversion flag This bit is set by hardware when new conversion data is available in the ADC_JDRy register after each injected channel conversion. Cleared by software writing 1 or by reading the corresponding ADC_JDRy register 0: Injected channel conversion not complete (or flag event has been acknowledged and cleared by software) 1: Injection channel conversion completed
4	OVR	RC_W1	0	Regular channel overload This bit is set by hardware when an overrun occurs. When the EOC flag is set, it indicates a new conversion has completed. It is cleared by software writing 1 to it. 0: No overrun occurred (or flag event has been acknowledged and cleared by software) 1: Overrun has occurred
3	EOS	RC_W1	0	End of regular channel sequence flag This bit is set by hardware when the selected regular sequence conversion ends. This bit is cleared by writing 1 0: Regular conversion sequence not complete (or flag event has been acknowledged and cleared by software) 1: Regular conversion sequence complete
2	EOC	RC_W1	0	End of regular channel conversion flag The hardware sets this bit when a new data result can be read from the ADC_DR register after each transition of each regular channel. It is cleared by software writing 1 to it or by reading the ADC_DR register. 0: Regular channel conversion not complete (or flag event has been acknowledged and cleared by software) 1: Regular channel conversion complete
1	EOSMP	RC_W1	0	End of channel sampling flag This bit is set by hardware at the end of the sampling phase for each regular conversion, and cleared by software writing 1 0: Sampling phase not ended (or flag event has been acknowledged and cleared by software) 1: End of sampling phase reached
0	ADRDY	RC_W1	0	ADC Ready This bit is set by hardware when the ADC is enabled (ADEN=1 or ADCAL=1) and the ADC reaches the ready state to accept conversion requests. This bit can be cleared by writing 1 in software. 0: ADC is not ready to start conversion (or the flag event has been acknowledged and cleared by software) 1: ADC is ready to start conversion

13.5.2. ADC Interrupt Enable Register (ADC_IER)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	EOCALI E	Res.	Res.	Res.	AWDI E	JEOSI E	JEOCI E	OVRI E	EOSI E	EOCI E	EOSMPI E	ADRDYI E
				RW				RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	EOCALIE	RW	0	Calibration Complete Interrupt Enable 0: Calibration completion interrupt disabled 1: Calibration completion interrupt enabled
10:8	Reserved	-	-	Reserved
7	AWDIE	RW	0	Analog watchdog interrupt enable 0: Analog watchdog interrupt disabled 1: Enable analog watchdog interrupt Software can write this bit when ADSTART=0 and JADSTART=0 (ensuring no conversion is in progress).
6	JEOSIE	RW	0	Injected sequence end of conversion interrupt enable 0: JEOS interrupt disabled 1: Enable JEOS interrupt Software can write this bit when JADSTART=0 (ensuring no conversion is in progress).
5	JEOCIE	RW	0	Interrupt enable for injected channels 0: JEOC interrupt disabled 1: Enable JEOC interrupt Software can write this bit when JADSTART=0 (ensuring no conversion is in progress).
4	OVRIE	RW	0	Regular channel overrun interrupt enable bit 0: Disable ADC overrun interrupt 1: Enable ADC overrun interrupt Software can write this bit when ADSTART=0 (ensuring no conversion is in progress).
3	EOSIE	RW	0	Regular channel sequence end interrupt enable bit 0: Disable sequence end interrupt 1: Enable sequence end interrupt Software can write this bit when ADSTART=0 (ensuring no conversion is in progress).
2	EOCIE	RW	0	Regular channel conversion end interrupt enable bit 0: Disable conversion end interrupt 1: Enable conversion end interrupt Software can write this bit when ADSTART=0 (ensuring no conversion is in progress).
1	EOSMPIE	RW	0	Regular channel sampling flag end interrupt enable bit 0: Sampling flag end interrupt disabled 1: EOSMP interrupt enabled. Software can write this bit when ADSTART=0 (ensuring no conversion is in progress).
0	ADRDYIE	RW	0	ADC ready interrupt enable bit 0: Disable ADC ready interrupt 1: Enable ADC ready interrupt Software can write this bit when ADSTART=0 and JADSTART=0 (ensuring no conversion is in progress).

13.5.3. ADC control register (ADC_CR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAL	Res	Res	Res	RSTCAL	Res	Res	Res	Res	Res	Res.	Res.	Res.	Res.	Res.	Res.
RS				RS											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	JADSTP	ADSTP	JADSTART	ADSTART	ADDIS	ADEEN
										RS	RS	RS	RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31	ADCAL	RS	0	ADC calibration starts. Software sets to start ADC calibration, hardware automatically clears it upon completion (success or failure). 0: Calibration complete

				1: Write 1 to calibrate the ADC. Read at 1 means that a calibration is in progress. Notes: 1. Software can write 1 but not 0, hardware clears it. 2. ADCAL=1 can only be written when ADEN=0. 3. The calibration coefficients can only be updated by software writing to ADC_CALFACT when ADEN=1, ADSTART=0, and JADSTART=0 (ADC is enabled, and no conversion is currently in progress).
30:28	Reserved	-	-	Reserved
27	RSTCAL	RS	1'b0	Reset calibration This bit is set to 1 by software write and cleared to 0 by hardware write. This bit is cleared after the calibration register is initialized (i.e., after RSTCAL is set to 1). 0: Calibration register initialized 1: Initialize calibration register Note: If RSTCAL is set when conversion is ongoing, additional cycles are required to clear the calibration registers.
26:6	Reserved	-	-	Reserved
5	JADSTP	RS	0	The ADC stops injecting the conversion command. Software sets to stop and discard ongoing injection transitions (JADSTP commands). It is cleared by hardware when the conversion is effectively discarded and the ADC is ready to accept a new start conversion command. 0: No ADC stop injection conversion command in progress 1: Write 1 stops the ADC injection conversion, and read 1 indicates that a JADSTP command is in progress. Software writing that the bit is 0 is an invalid operation. Note: JADSTP can only be written as 1 when JADSTART=1 and ADDIS=0.
4	ADSTP	RS	0	The ADC stops the rule conversion command. This bit is set by software to stop and discard an ongoing conversion (ADSTP Command). It is cleared by hardware when the conversion is effectively discarded and the ADC is ready to accept a new start conversion command. 0: No ADC stop conversion command ongoing 1: Write 1 stops ADC rule conversion, and read 1 indicates that an ADSTP command is in progress. Software writing that the bit is 0 is an invalid operation. Notes: 1. ADSTP can only be written as 1 when ADSTART=1 and ADDIS=0. 2. In auto-injected mode (JAUTO=1), ADSTP can simultaneously terminate both regular and auto-injected conversions.
3	JADSTART	RS	1'b0	ADC starts injected conversion This bit is set by software and is used to initiate ADC conversion of the injection channel. The conversion is determined to start immediately by software or by a hardware trigger event based on the configuration of JEXTEN[1:0]. 0: No ADC injected conversion is currently in progress 1: Writing 1 starts ADC injected conversion; reading as 1 indicates the ADC is operating, possibly in conversion. If this bit is cleared by hardware: – In single conversion mode (CONT=0, DISCEN=0), when software-triggered (JEXTEN=00): at the end of sequence conversion (when JEOS flag is set) In all other cases: after the execution of the JADSTP command, at the same time as the JADSTP bit is cleared by hardware. Notes: 1. Software can only configure JADSTART=1 when ADEN=1 and ADDIS=0. Writing 0 to this bit by software has no effect. 2. In auto-injection mode (JAUTO=1), setting ADSTART can initiate both regular and auto-injected conversions.
2	ADSTART	RS	0	ADC start regular conversion

				This bit is set by software to start ADC regular conversion. Depending on the EXTEN [1:0] configuration bits, a conversion either starts immediately (software trigger configuration) or once a hardware trigger event occurs (hardware trigger configuration). 0: No ADC regular conversion is currently in progress 1: Write 1 to start ADC rule conversion. Read 1 indicates that the ADC is operating and may be converting. If this bit is cleared by hardware: In single conversion mode (CONT=0, DISCEN=0), when software trigger is selected (EXTEN=00): at the assertion of the end of Conversion Sequence (EOS) flag. – In discontinuous conversion mode (CONT=0, DISCEN=1), when software-triggered (EXTEN=00): at the end of conversion (when EOC flag is set) In all other cases: after the execution of the ADSTP command, at the same time as the ADSTP bit is cleared by hardware. Notes: - Software can only configure ADSTART=1 when ADEN=1 and ADDIS=0. Writing 0 to this bit by software has no effect. - In auto-injection mode (JAUTO=1), setting ADSTART can initiate both regular and auto-injected conversions.
1	ADDIS	RS	0	ADEN disabled. Software set disables ADC. This bit is cleared by hardware when ADC is disabled (ADEN cleared by hardware). Software writing that the bit is 0 is an invalid operation. 0: No ADDIS command is currently being executed 1: Write 1 to disable the ADC. Read 1 means that an ADDIS command is in progress. Note: ADDIS can only be set to 1 when ADEN=1, ADSTART=0, and JADSTART=0 (indicating no ongoing conversion).
0	ADEN	RS	0	ADC enable command Software sets this bit to enable ADC. After ADRDY flag is set to 1, ADC is ready for conversion operation. Software writing that the bit is 0 is an invalid operation. 0: ADC disabled (OFF state) 1: ADC is enabled Note: ADEN can only be configured to 1 when all bits in ADC_CR register are 0.

Note: The interval between software clearing ADEN and re-enabling both ADEN and ADSTART should be 4 ADC_CLK cycles.

13.5.4. ADC configuration register (ADC_CFGR)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AWDCH[5:0]						JAUTO	JAWDEN	AWDEN	AWDSGL	Res.	JDISCEN	DISNUM[2:0]			DISCEN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALIGN	AUTDLY	CONT	OVRRMOD	EXTEN[1:0]		Res.	EXTSEL[3:0]				RES[1:0]		Res.	DMAFG	DMAEN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	AWDCH[5:0]	RW	0000	Analog watchdog channel selection. These bits are set and cleared by software. They select the input channel to be guarded by the analog watchdog. 000000: External Channel 0 000001: External Channel 1 ...

				010111: External Channel 23 011000: Reserved 011001: Reserved 011010: External Channel 26 011011: Internal VTS 011100: Internal V_{REFINT} 011101: Internal $V_{CC}/3$ 011110: Internal DAC Output 011111: Internal OPA Output Others: Reserved Note: This bit can only be configured when ADSTART=0 and JADSTART=0.
25	JAUTO	RW	1'b0	Automatic injected group conversion This bit is set and cleared by software to enable/disable automatic conversion of the injected channel group after the regular channel group conversion ends. 0: Disable automatic conversion of the injected channel group 1: Enable automatic conversion of the injected channel group Note: This bit can only be configured when ADSTART=0 and JADSTART=0.
24	JAWDEN	RW	1'b0	Injected Channel Analog Watchdog Enable Enable the analog watchdog on the injected channels. This bit is set and cleared by software. 0: Analog watchdog disabled on injected channels 1: Enable the analog watchdog on the injected channels Note: Software is allowed to write this bit only when JADSTART=0 (to ensure no ongoing conversion).
23	AWDEN	RW	0	Regular Channel Analog Watchdog Enable Enable analog watchdog on regular channels. This bit is set and cleared by software. 0: Analog watchdog disabled 1: Analog watchdog enabled Note: Software can write this bit only when ADSTART=0 (to ensure no ongoing conversion).
22	AWDSGL	RW	0	Enable the watchdog on a single channel or on all channels Software can enable or disable the analog watchdog on the channel set by AWDCH[5:0] or on all channels by setting or clearing this bit. 0: Analog watchdog enabled on all channels 1: Enable analog watchdog on a single channel (AWDCH[5:0] configures which channel). Note: Software can write this bit only when ADSTART=0 and JADSTART=0 (to ensure no ongoing conversion).
21	Reserved	-	-	Reserved
20	JDISCEN	RW	1'b0	Injected channel discontinuous mode enable. Software can enable or disable discontinuous mode on the injected channel group by setting or clearing this bit. 0: Disable discontinuous mode on the injected channel group. 1: Enable discontinuous mode on the injected channel group. Note: Software is allowed to write this bit only when JADSTART=0 (to ensure no ongoing conversion).
19:17	DISCNUM [2: 0]	RW	3'b0	Number of discontinuous mode conversion channels. These bits are written by software to define the number of regular channels to be converted in discontinuous mode after receiving an external trigger. 000: 1 channel 001: 2 channel ... 111: 8 channels Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
16	DISCEN	RW	0	Discontinuous mode enable.

				Software enables/disables discontinuous mode by setting or clearing this bit. 0: Discontinuous mode disabled 1: Discontinuous mode enabled It is not possible to enable both discontinuous mode and continuous mode simultaneously, i.e., setting DISCEN=1 and CONT=1 is prohibited. Note: Software can write this bit only when ADSTART=0 (to ensure no ongoing conversion).
15	ALIGN	RW	0	Data alignment. Software selects right or left alignment by setting or clearing this bit. 0: Right alignment 1: Left alignment Note: Software can write this bit only when ADSTART=0 and JADSTART=0 (to ensure no ongoing conversion).
14	AUTDLY	RW	0	Delayed conversion mode. The software enables/disables the wait conversion mode by setting and clearing this bit. 0: Disable wait conversion mode 1: Enable wait conversion mode Note: Software is allowed to write these bits only when ADSTART=0 and JADSTART=0 (to ensure no ongoing conversion).
13	CONT	RW	0	Single / continuous conversion mode This bit is set and cleared by software. If it is set, conversion takes place continuously until it is cleared. It is not possible to enable both discontinuous mode and continuous mode simultaneously, i.e., setting DISCEN=1 and CONT=1 is prohibited. Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
12	OVRMOD	RW	0	Overrun management mode The software configures the data overload management method by setting and clearing this bit. 0: When an overload occurs, the ADC_DR register retains the original data 1: When an overload occurs, the ADC_DR register is overwritten by the last conversion result Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
11:10	EXTEN[1:0]	RW	00	External trigger enable and polarity selection The software selects the trigger polarity and enables the trigger by setting and clearing this bit. 00: Hardware trigger detection disabled (conversions can be started by software) 01: Hardware trigger detection on the rising edge 10: Hardware trigger detection on the falling edge 11: Hardware trigger detection on both the rising and falling edges Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
9	Reserved	-	-	Reserved
8:5	EXTSEL[3:0]	RW	000	External trigger selection. This bit selects the external event to trigger the regular conversion start. 0000: TRG0(TIM1_TRGO) 0001: TRG1(TIM1_OC1) 0010: TRG2(TIM1_OC2) 0011: TRG3(TIM1_OC3) 0100: TRG4(TIM1_OC4) 0101: TRG5(TIM2_TRGO) 0110: TRG6(TIM2_OC1)

				0111: TRG7(TIM3_OC1) 1000: TRG8(TIM3_TRGO) 1001: TRG9(TIM15_TRGO) 1010: Reserved 1011: Reserved 1100: TRG12(EXTI11) 1101: TRG13(EXTI15) 1110: Reserved 1111: Reserved Note: When EXTI11 or EXTI15 is selected as the trigger, the trigger polarity follows the definition in the EXTI module. That is, as long as the EXTI polarity definition is met, setting EXTEN to rising or falling edge can trigger the conversion.
4:3	RES[1:0]	RW	00	Data resolution These bits are written by software to select the resolution of the conversion. 00: 12-bit 01: 10-bit 10: 8-bit 11: 6-bit Notes: 1. Software is allowed to write these bits only when ADEN=0 2. Software is allowed to write these bits only when ADSTART=0 and JADSTART=0 (to ensure no ongoing conversion).
2	Reserved	-	-	Reserved
1	DMACFG	RW		DMA access mode configuration The software selects between two DMA operation modes by setting and clearing this bit, which takes effect when DMAEN=1. 0: DMA single mode selection 1: DMA cycle mode selection Note: Software is allowed to write these bits only when ADSTART=0 and JADSTART=0 (to ensure no ongoing conversion).
0	DMAEN	RW	0	DMA access enable. The software enables/disables DMA requests by setting and clearing this bit. Use the DMA controller to manage automatic conversion data. 0: Disable DMA 1: Enable DMA Note: Software is allowed to write these bits only when ADSTART=0 and JADSTART=0 (to ensure no ongoing conversion).

13.5.5. ADC configuration register 2 (ADC_CFGR2)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	CALNUM[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GCOMP
	RW	RW	RW												RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	ROVSM	TROVS	OVSS[3:0]			OVSR[2:0]			OVSE	ROVSE	
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30:28	CALNUM[2:0]	RW	3'b0	Average number of calibrations 000: 1 calibration and average 001: 2 calibrations and averaging 010: 4 calibrations and averaging

				011: 8 calibrations and averaging 100: 16 calibrations and averaging 101: 32 calibrations and averaging Others: Reserved
27:17	Reserved	-	-	Reserved
16	GCOMP	RW	0	Gain compensation mode The software enables/disables the gain compensation mode by setting and clearing this bit. 0: Disable gain compensation 1: Enable gain compensation and apply it to all channels Note: The software is only allowed to write this bit when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress)
15:11	Reserved	-	-	Reserved
10	ROVSM	RW	1'b0	Regular oversampling mode The software selects the regular oversampling mode by setting and clearing this bit. 0: Continuous mode: When a triggered injection conversion occurs, oversampling is temporarily paused and resumes after the injection sequence (the oversampling buffer is preserved during the injection sequence). 1: Resume mode: When a triggered injection conversion occurs, the current oversampling is aborted, and oversampling restarts from the beginning after the injection sequence (the oversampling buffer is cleared when the injection sequence starts conversion). Note: Software is allowed to write this bit only when ADSTART=0 (this ensures no conversion is in progress).
9	TROVS	RW	1'b0	Trigger rule oversampling Software sets and clears this bit to enable/disable triggered oversampling. 0: All oversampling conversions for a channel are completed continuously after triggering. 1: Each oversampling conversion for a channel requires a new trigger. Note: Software is allowed to write this bit only when ADSTART=0 (this ensures no conversion is in progress).
8:5	OVSS[3:0]	RW	4'b0	Oversampling shift control. This bit field is configured by software to define the number of right shifts applied to the raw oversampling result. 0000: No shift 0001: Shift by 1 bit 0010: shift by 2 bits 0011: shift by 3 bits 0100: Shift 4 bits 0101: Shift by 5 bits 0110: Shift 6 bits 0111: Shift by 7 bits 1000: shift 8 bits Other: Reserved Note: Software is allowed to write these bits only when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress).
4:2	OVSR[2:0]	RW	3'b0	Oversampling rate This bit field is configured by software to define the oversampling ratio. 000: 2x 001: 4x 010: 8x 011: 16x 100: 32 x 101: 64x 110: 128 x 111: 256 x Note: The software allows writing these bits only when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress).
1	OVSE	RW	1'b0	Injection Oversampling Enable The software enables/disables injected oversampling by setting and clearing this bit.

				0: Disable injection oversampling 1: Enable injected oversampling Note: The software allows writing this bit only when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress)
0	ROVSE	RW	1'b0	Regular Oversampling Enable The software enables/disables regular oversampling by setting and clearing this bit. 0: Disable regular oversampling 1: Enable regular oversampling Note: The software allows writing this bit only when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress)

13.5.6. ADC Sample Time Register 1 (ADC_SMPR1)

Offset address: 0x14.

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SMP9 [2:0]			SMP8 [2:0]			SMP7 [2:0]			SMP6 [2:0]			SMP5 [2:1]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP5[0]	SMP4 [2:0]			SMP3 [2:0]			SMP2 [2:0]			SMP1 [2:0]			SMP0 [2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:0	SMPx[2:0]	RW	3'b000	Sampling time selection The software can configure this bit to select the sampling time for channel x. 000: 2.5 ADC_CLK 001: 6.5 ADC_CLK 010: 12.5 ADC_CLK 011: 24.5 ADC_CLK 100: 47.5 ADC_CLK 101: 92.5 ADC_CLK 110: 247.5 ADC_CLK 111: 640.5 ADC_CLK Software can only write these bits when ADSTART=0 and JADSTART=0 (ensuring no ongoing conversion). Note: When ADC_CLK frequency = PCLK, sampling time cannot be configured as 000 (2.5 ADC_CLK).

13.5.7. ADC Sample Time Register 2 (ADC_SMPR2)

Offset address: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SMP19 [2:0]			SMP18 [2:0]			SMP17 [2:0]			SMP16 [2:0]			SMP15 [2:1]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP15[0]	SMP14 [2:0]			SMP13 [2:0]			SMP12 [2:0]			SMP11 [2:0]			SMP10 [2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:0	SMPx[2:0]	RW	3'b000	Refer to the description of ADC_SMPR1

13.5.8. ADC Sample Time Register 3 (ADC_SMPR3)

Offset address: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SMP29 [2:0]			SMP28 [2:0]			SMP27 [2:0]			SMP26 [2:0]			SMP25 [2:1]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP25[0]	SMP24 [2:0]			SMP23 [2:0]			SMP22 [2:0]			SMP21 [2:0]			SMP20 [2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:0	SMPx[2:0]	RW	3'b000	Refer to the description of ADC_SMPR1

13.5.9. ADC Sample Time Register 4 (ADC_SMPR4)

Offset address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMP31 [2:0]			SMP30 [2:0]		
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5:0	SMPx[2:0]	RW	3'b000	Refer to the description of ADC_SMPR1

13.5.10. ADC watchdog threshold register (ADC_TR)

Offset address: 0x24

Reset value: 0x0FFF 0000

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

Bit	Name	R/W	Reset value	Function
31:28	Reserved	-	-	Reserved
27:16	HT[11:0]	RW	12'hFFF	Analog watchdog higher threshold These bits are written by software to define the higher threshold for the analog watchdog. Software can only write these bits when ADSTART=0 and JADSTART=0 (ensuring no ongoing conversion).
15:12	Reserved	-	-	Reserved
11:0	LT[11:0]	RW	12'h0	Analog watchdog lower threshold These bits are written by software to define the lower threshold for the analog watchdog. Software can only write these bits when ADSTART=0 and JADSTART=0 (ensuring no ongoing conversion).

13.5.11. ADC channel selection register 1 (ADC_SQR1)

Offset address: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SQ4 [5:0]						SQ3 [5:0]						SQ2 [5:4]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ2 [3:0]				SQ1 [5:0]						Res.	Res.	L [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:24	SQ4 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 4th conversion in regular sequence, these bits define the channel number for the 4th conversion in the conversion sequence. Refer to SQ1 definition for corresponding channels.
23:18	SQ3 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 3rd conversion in regular sequence, these bits define the channel number for the 3rd conversion in the conversion sequence. Refer to SQ1 definition for corresponding channels.
17:12	SQ2 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 2nd conversion in regular sequence, these bits define the channel number for the 2nd conversion in the conversion sequence. Refer to SQ1 definition for corresponding channels.
11:6	SQ1 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 1st conversion in regular sequence, these bits define the channel number for the 1st conversion in the conversion sequence. Corresponding channels are as follows: 000000: External channel 0 (PA0) 000001: External channel 1 (PA1) 000010: External channel 2 (PA2) 000011: External channel 3 (PA3) 000100: External Channel 4 (PB7) 000101: External Channel 5 (PB8) 000110: External Channel 6 (PC12) 000111: External Channel 7 (PC13) 001000: External Channel 8 (PC14) 001001: External Channel 9 (PC15) 001010: External Channel 10 (PD0) 001011: External Channel 11 (PD1) 001100: External Channel 12 (PD2) 001101: External Channel 13 (PD3) 001110: External Channel 14 (PD4) 001111: External Channel 15 (PD7) 010000: External Channel 16 (PD8) 010001: External Channel 17 (PA11) 010010: External Channel 18 (PA12) 010011: External Channel 19 (PA13) 010100: External Channel 20 (PA14) 010101: External Channel 21 (PC10) 010110: External Channel 22 (PC11) 010111: External Channel 23 (PD9) 011000: Reserved 011001: Reserved 011010: External Channel 26 (PD12) 011011: Internal TS 011100: Internal V_{REFINT} 011101: Internal $V_{CC}/3$ 011110: Internal DAC Output 011111: Internal OPA output Other: Reserved Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
5:4	Reserved	-	-	Reserved
3:0	L [3: 0]	RW	4'b0	Regular channel sequence length These bits are defined by software. These bits define the number of channels in the regular channel conversion sequence. 0000: 1 conversion 0001: 2 conversions 1111: 16 conversions

				Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).

13.5.12. ADC channel selection register 2 (ADC_SQR2)

Offset address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SQ9 [5:0]						SQ8 [5:0]						SQ7 [5:4]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ7 [3:0]				SQ6 [5:0]						SQ5 [5:0]					
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:24	SQ9 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 9th conversion in regular sequence, these bits define the channel number for the 9th conversion in the sequence. Refer to SQ1 definition for corresponding channels.
23:18	SQ8 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 8th conversion in regular sequence, these bits define the channel number for the 8th conversion in the sequence. Refer to SQ1 definition for corresponding channels.
17:12	SQ7 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 7th conversion in regular sequence, these bits define the channel number for the 7th conversion in the sequence. Refer to SQ1 definition for corresponding channels.
11:6	SQ6 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 6th conversion in regular sequence, these bits define the channel number for the 6th conversion in the sequence. Refer to SQ1 definition for corresponding channels.
5:0	SQ5 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 5th conversion in regular sequence, these bits define the channel number for the 5th conversion in the sequence. Refer to SQ1 definition for the corresponding channel.

13.5.13. ADC channel selection register 3 (ADC_SQR3)

Offset Address: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SQ14 [5:0]						SQ13 [5:0]						SQ12 [5:4]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ12 [3:0]				SQ11 [5:0]						SQ10 [5:0]					
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:24	SQ14 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 14th conversion in the regular sequence, these bits define the channel number for the 14th conversion in the sequence. Refer to SQ1 definition for corresponding channels.

23:18	SQ13 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 13th conversion in the regular sequence, these bits define the channel number for the 13th conversion in the sequence. Refer to SQ1 definition for corresponding channels.
17:12	SQ12 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 12th conversion in the regular sequence, these bits define the channel number for the 12th conversion in the sequence. Refer to SQ1 definition for corresponding channels.
11:6	SQ11 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 11th conversion in the regular sequence, these bits define the channel number for the 11th conversion in the sequence. Refer to SQ1 definition for corresponding channels.
5:0	SQ10 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 10th conversion in the regular sequence, these bits define the channel number for the 10th conversion in the sequence. Refer to SQ1 definition for corresponding channels.

13.5.14. ADC channel selection register 4 (ADC_SQR4)

Offset Address: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	SQ16 [5:0]						SQ15 [5:0]					
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:6	SQ16 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. 16th conversion in the regular sequence, these bits define the channel number for the 16th conversion in the sequence. Refer to SQ1 definition for corresponding channels.
5:0	SQ15 [5:0]	RW	6'b0	Channel selection. These bits are defined by software. The 15th conversion in the regular sequence, these bits define the channel number for the 15th conversion in the regular conversion sequence. Refer to SQ1 definition for corresponding channels.

13.5.15. ADC data register (ADC_DR)

Offset address: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	RDATA[15:0]	R	16'h0	Regular channel conversion data. These bits are read-only. The conversion result of the last converted channel is placed in this register. The data are left- or right-aligned.

13.5.16. ADC injected channel selection register (ADC_JSQR)

Offset address: 0x4C

Reset value : 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
JSQ4 [5:0]						JSQ3 [5:0]						JSQ2 [5:2]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JSQ2 [1:0]		JSQ1 [5:0]						JEXTEN[1:0]		JEXTSEL[3:0]			JL[1:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	JSQ4 [5:0]	RW	6'b0	4th transformation of injection sequence These bits are defined by software. The 4th conversion in the injected sequence, these bits define the channel number for the 4th conversion in the injected conversion sequence. Refer to ADC_SQR1.SQ1 definition for corresponding channels. Note: Software is allowed to write these bits only when JADSTART=0 (to ensure no ongoing conversion).
25:20	JSQ3 [5:0]	RW	6'b0	3rd transformation of injection sequence These bits are defined by software. The 3rd conversion in the injected sequence, these bits define the channel number for the 3rd conversion in the injected conversion sequence. Refer to ADC_SQR1.SQ1 definition for corresponding channels. Note: Software is allowed to write these bits only when JADSTART=0 (to ensure no ongoing conversion).
19:14	JSQ2 [5:0]	RW	6'b0	2nd transformation of injection sequence These bits are defined by software. The 2nd conversion in the injected sequence, these bits define the channel number for the 2nd conversion in the injected conversion sequence. Refer to ADC_SQR1.SQ1 definition for corresponding channels. Note: Software is allowed to write these bits only when JADSTART=0 (to ensure no ongoing conversion).
13:8	JSQ1 [5:0]	RW	6'b0	1st transformation of injection sequence These bits are defined by software. The 1st conversion in the injected sequence, these bits define the channel number for the 1st conversion in the injected conversion sequence. Refer to ADC_SQR1.SQ1 definition for corresponding channel. Note: Software is allowed to write these bits only when JADSTART=0 (to ensure no ongoing conversion).
7:6	JEXTEN[1:0]	RW	2'b0	Injected channel trigger enable and polarity selection These bits are set and cleared by software to select external trigger polarity and enable trigger for injected group. 00: Disable hardware trigger detection (conversion can be initiated by software) 01: Hardware trigger detection of rising edge 10: Software triggered detection of falling edge 11: Both rising and falling edges have hardware trigger detection Note: Software is allowed to write these bits only when JADSTART=0 (to ensure no ongoing conversion).
5:2	JEXTSEL[3:0]	RW	5'b0	External trigger event selection for injected channel group. This register selects the external event that triggers the start of injected conversion. 0000: TRG0(TIM1_TRGO) 0001: TRG1(TIM1_OC1) 0010: TRG2(TIM1_OC2) 0011: TRG3(TIM1_OC3) 0100: TRG4(TIM1_OC4) 0101: TRG5 (TIM2_TRGO) 0110: TRG6(TIM2_OC1) 0111: TRG7(TIM3_OC1) 1000: TRG8(TIM3_TRGO)

				1001: TRG9(TIM15_TRGO) 1010: Reserved 1011: Reserved 1100: TRG12(EXTI11) 1101: TRG13(EXTI15) 1110: Reserved 1111: Reserved Note: When EXTI11 or EXTI15 is selected as the trigger, the trigger polarity follows the definition in the EXTI module. That is, as long as the EXTI polarity definition is met, setting EXTEN to rising or falling edge can trigger the conversion. Note: Software is allowed to write these bits only when JADSTART=0 (to ensure no ongoing conversion).
1:0	JL[1:0]	RW	2'b0	Injected sequence length These bits are written in software to define the total number of transitions in the injection channel transition sequence. 00: 1 conversion 01: 2 conversions 10: 3 conversions 11: 4 conversions Note: Software is allowed to write these bits only when JADSTART=0 (to ensure no ongoing conversion).

13.5.17. ADC offset register (ADC_OFRy)

Offset address: 0x60+0x04(y-1), (y=1 to 4)

Reset value: 0x0000 0000

Note: SATEN and OFFSETPOS are only available in ADC_OFR1, and are shared by ADC_OFR1~4.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OFFSETy_EN	OFFSETy_CH[5:0]						SATEN	OFFSETPOS	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	OFFSETy [11: 0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	OFFSETy_EN	RW	1'b0	OFFSETy Enable This bit is written by software to enable or disable the offset value OFFSETy. Note: Software is only allowed to write this bit when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress).
30:25	OFFSETy_CH[5:0]	RW	6'b0	Channel selection for data offset y These bits are written by software to define the channel to which the OFFSETy offset will be applied. Some channels are not physically connected and therefore cannot be selected for the data offset y. Note: Software is only allowed to write this bit when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress).
24	SATEN	RW	1'b0	Saturation enabling This bit is set and cleared by the software to enable the offset to saturate at 0x000 and 0xFFFF. 0: No saturation control, the offset result can be signed 1: Saturation enabled, offset result unsigned and saturated at 0x000 and 0xFFFF Note: Software is only allowed to write this bit when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress).
23	OFFSETPOS	RW	1'b0	Positive offset This bit is set and cleared by the software to enable positive offset. 0: negative offset 1: positive offset Note: Software is only allowed to write this bit when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress).
22:12	Reserved	-	-	Reserved
11:0	OFFSETy [11: 0]	RW	12'b0	The data offset y of the OFFSETy_CH [4: 0] channel.

				These bits are written by software to define the offset y that is subtracted from the raw converted data at the time of converting a channel (which can be regular or injected). The channel for applying data offset y must be configured in the OFFSETy_CH bit. The conversion result can be read from the ADC_DR (regular conversion) or the ADC_JDRy register (injection conversion). If multiple offsets (OFFSETy) point to the same channel, only the offset with the lowest x value is considered for subtraction. For example: If OFFSET1_CH=4 and OFFSET2_CH=4, then OFFSET1[11:0] is subtracted when converting channel 4. Note: Software is only allowed to write this bit when ADSTART=0 and JADSTART=0 (this ensures no conversion is in progress).

13.5.18. ADC injected channel data register (ADC_JDRy)

Offset address: 0x80+0x04(y-1), (y=1 to 4)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JDATA[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	JDATA[15:0]	R	16'h0	Injection transformation data These bits are read-only. The conversion result of the injection channel y is placed in this register. The data are left- or right-aligned.

13.5.19. ADC calibration factor register (ADC_CALFACT)

Offset address: 0xB4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCALFACT[8:0]									Res.	CAPSUC	OFFSUC	Res.	Res.	Res.	Res.
R	R	R	R	R	R	R	R	R		RC_W1	RC_W1				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDVLD	WRVLD	FACTSEL [4: 0]						WCALFACT [8: 0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:23	RCALFACT[8:0]	R	9'h00	Read calibration result register (represented in two's complement). When RDVLD is active, the ADC calibration result is read. The software can only read it if ADCAL is 0. When RDVLD is valid, the overflow flag bit is read. The software can only read it if ADCAL is 0.
22	Reserved	-	-	Reserved
21	CAPSUC	RC_W1	1'h0	Capacitance calibration status bit. Indicates whether the ADC capacitance calibration is successful. Hardware sets the bit, software writes 1 to clear. 0: Capacitance calibration failed 1: Capacitance calibration succeeded
20	OFFSUC	RC_W1	1'h0	Offset calibration status bit. Indicates whether ADC offset calibration succeeded. Hardware sets the bit, software writes 1 to clear. 0: Offset calibration failed 1: Offset calibration succeeded
19:16	Reserved	-	-	Reserved
15	RDVLD	RW	1'h0	Overflow flag bit read enabled. 0: Read overflow flag disabled 1: Enable read overflow flag Also calibration factor read enable. 0: Read calibration factor disabled

				1: Enable read calibration factor
14	WRVLD	RW	1'h0	Calibration factor write enabled. 0: Write calibration factor disabled 1: Enable write calibration factor
13:9	FACTSEL [4: 0]	RW	5'h00	Read/write calibration value selection bit When RDVLD/WRVLD is active, select which capacitor the calibration result to be read/the calibration factor to be written to be loaded. {cov_error, cal_error} is set by hardware and cleared by software. You need to enable RDVLD when reading {cov_error, cal_error}. Enable WDVLD when writing {cov_error, cal_error}. Clear cov_error by writing WCALFACT[1], clear cal_error by writing WCALFACT[0]. 5'd1: dos 5'd2: dcpm5 5'd3: dcpm4 5'd4: dcpm3 5'd5: dcpm2 5'd6: dcpm1 5'd7: dcpn4 5'd8: dcpn3 5'd9: dcpn2 5'd10: dcpn1 5'd11: dcpk3 5'd12: dcm5 5'd13: dcm4 5'd14: dcm3 5'd15: dcm2 5'd16: dcm1 5'd17: dcnm0 5'd18: {cov_error, cal_error}: Cov_error: When AD conversion is performed, the calibration result is compensated to overflow during AD conversion. The hardware sets this bit, and the software writes 1 to clear it. cal_error: During calibration, if an overflow occurs in the calibration process, hardware sets this bit. Software clears it by writing 1.
8:0	WCALFACT [8: 0]	RW	9'h00	Write the calibration factor register (complement representation). When WRVLD is valid, the WCALFACT value is loaded into the ADC. The software can only write when ADCAL is 0.

13.5.20. ADC gain compensation register (ADC_GCOMP)

Offset address: 0xC0

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13:0	GCOMP_COEFF [13: 0]	RW	14'h0	Gain compensation coefficient These bits are configured by software to program the gain compensation coefficient. 00 100 0 000 0 000 0: Gain factor of 0.5 ... 01 000 0 000 0 000 0: Gain factor of 1 10 000 0 000 0 000 0: Gain factor of 2 11 000 0 000 0 000 0: Gain factor of 3 ...

				This coefficient is divided by 4096 to give a gain factor ranging from 0 to 3.999756. Note: This gain compensation is only valid when the GCOMP bit in the ADC_CFGR2 register is set to 1.

13.5.21. ADC common configuration register (ADC_CCR)

Offset address: 0x308

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	DIFF_en	VREFSEL	PWR MODE [2: 0]			Res	TSEN	VREFINT_EN	PRESC[3:0]				CKMODE[1:0]	
.	.	RW	RW	RW	RW	RW	.	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res.	Res.	Res	Res	Res	Res	Res.	Res.	Res	Res	Res	Res	Res.	Res.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29	DIFF_en	RW	0	Differential input enable 0: single-ended input 1: differential input
28	VREFSEL	RW	0	ADC reference voltage selection. 0: Select V _{REFP} (connect to V _{CC} when V _{REFP} is not present) 1: Select V _{REFBUF} (VREFBUF_CR.VREFBUF_EN must be enabled)
27:25	PWR MODE [2: 0]	RW	3'b0	Power Consumption Mode Configuration of Comparators in ADC 000: typical 6μA 001: typical 7μA 010: typical 8μA 011: typical 9μA 100: typical 2μA 101: typical 3μA 110: typical 4μA 111: typical 5μA Note: Writing to these bits via software is only allowed when the ADC is disabled (ADCAL=0, ADSTART=0, JADSTART=0, ADSTP=0, JADSTP=0, ADDIS=0, and ADEN=0).
24	Reserved	-	-	Reserved
23	TSEN	RW	0	Temperature sensor enable bit. This bit can be set or cleared by software to enable/disable the temperature sensor. 0: Temperature sensor channel disabled 1: Temperature sensor channel enabled Note: Writing to this bit via software is only allowed when the ADC is disabled (ADCAL=0, ADSTART=0, JADSTART=0, ADSTP=0, JADSTP=0, ADDIS=0, and ADEN=0).
22	VREFINT_EN	RW	0	V _{REFINT} reference enable bit. This bit can be set or cleared by software to enable/disable the V _{REFINT} reference. 0: Disabled 1: Enabled Note: Writing to this bit via software is only allowed when the ADC is disabled (ADCAL=0, ADSTART=0, JADSTART=0, ADSTP=0, JADSTP=0, ADDIS=0, and ADEN=0).
21:18	PRESC[3:0]	RW	0	ADC prescalation factor, for asynchronous clock mode Configured by software to select the ADC clock frequency. 0000: Input ADC clock undivided 0001: Input ADC clock 2 divided 0010: Input ADC clock 4 divided 0011: Input ADC clock 6 divided 0100: Input ADC clock 8 divided 0101: Input ADC clock 10 divided 0110: Input ADC clock 12 divided 0111: Input ADC clock 16 divided 1000: Input ADC clock 32 divided 1001: Input ADC clock 64 divided 1010: Input ADC clock 128 divided 1011: Input ADC clock

				256 divided Other values: Reserved Note: Writing to these bits via software is only allowed when the ADC is disabled (ADCAL=0, ADSTART=0, JADSTART=0, ADSTP=0, JADSTP=0, ADDIS=0, and ADEN=0).
17:16	CKMODE[1:0]	RW	0	ADC clock mode Configured by software to define the method of providing clock to the analog ADC: 00: ADC_CLK (asynchronous clock mode), source configured by RCC 01: PCLK/2 (synchronous clock mode) 10: PCLK/4 (synchronous clock mode) 11: PCLK (synchronous clock mode). When this configuration is enabled, the clock duty cycle of PCLK must be 50%. In all synchronous clock modes, there is no jitter in the delay from a timer trigger to the start of a conversion. Note: Writing to these bits via software is only allowed when the ADC is disabled (ADCAL=0, ADSTART=0, JADSTART=0, ADSTP=0, JADSTP=0, ADDIS=0, and ADEN=0).
15:0	Reserved	-	-	Reserved

14. Digital-to-analog converter (DAC)

14.1. Introduction

The digital-to-analog converter module (DAC) is a 12-bit digital input, voltage-output digital-to-analog converter. The DAC can be configured in 8-bit or 12-bit mode, or can be used in conjunction with a DMA controller. When the DAC is operating in 12-bit mode, the data can be left justified or right justified. The DAC can use the reference voltage V_{REFP} (refer to the datasheet for availability) via pin input for more accurate conversion results.

14.2. DAC main features

- 1 DAC converter: Each converter corresponds to 1 output channel.
- Left or right data alignment in 12-bit mode
- Synchronized update capability
- Noise-wave generation
- Triangular-wave generation
- Channels support DMA functionality.
- Support DMA underflow error detection
- Conversion on external trigger
- Input reference voltages V_{REFP} (V_{CCA}) and V_{REFBUF} .

14.3. DAC functional description

14.3.1. DAC block diagram

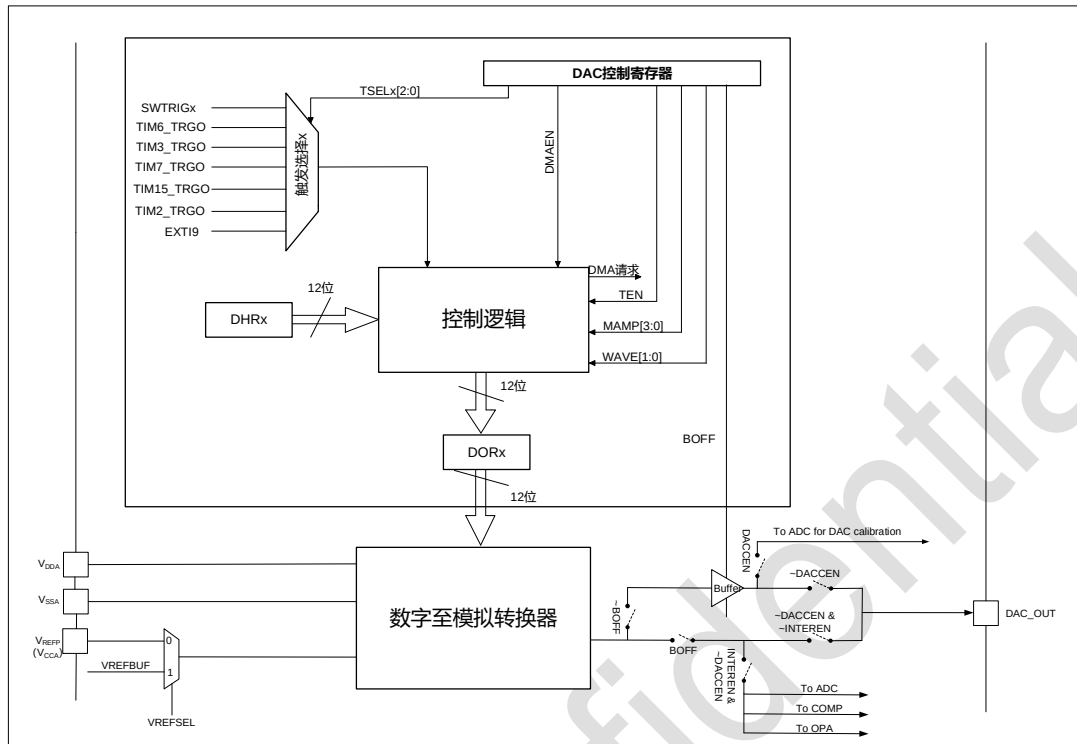


Figure 14-1 DAC block diagram

14.3.2. DAC channel enable.

Set the EN bit in the DAC_CR1 register to '1' to power on the DAC. After a startup time t_{WAKEUP} (refer to the datasheet for the value), the DAC is enabled.

Note: The EN bit only enables the analog part of the DAC. Even if this bit is set to '0', the digital part (e.g., registers) of the DAC remains operational.

14.3.3. DAC output buffer enable.

The DAC integrates an output buffer, which can reduce output impedance and directly drive external loads without requiring an external op-amp. The output buffer can be enabled or disabled by setting the BOFF bit in the DAC_CR1 register.

14.3.4. DAC calibration

14.3.4.1. DAC external calibration (via ADC).

The DAC can be calibrated via ADC. External reference voltage and internal reference voltage can store calibration values separately. Taking the selection of internal reference voltage V_{REFBUF} ($VREFSEL=1$) as an example, the steps are as follows:

1. Enable the ADC, select the internal channel mode for ADC, and choose the DAC input for the internal channel (the ADC must be calibrated first).
2. Enable V_{REFBUF} and select the V_{REFBUF} voltage value as required.
3. Configure the EN and DACCEN bits of the DAC_CR1 register to 1, BOFF=0; set DAC_OS_TRIMCR_VREFBUF in the DAC_CALR register to 6'h11.

4. Fix DAC_DOR.DACDOR[11:0]=0x7FF (2047) for the DAC, adjust the value of DAC_OS_TRIMCR_VREFBUF until the ADC output value is 2047 ± 1 , at which point the calibration is complete.

Note: The trim range is $\pm 15\text{mV}$. Increasing DAC_OS_TRIMCR_VREFBUF by two bits will increase the ADC output by 1LSB @3.3V 25°C.

14.3.4.2. DAC Internal Calibration

The DAC can be calibrated internally. External reference voltage and internal reference voltage can store calibration values separately. Taking the selection of internal reference voltage V_{REFBUF} (VREFSEL=1) as an example, the steps are as follows:

1. Configure EN=1, BUF_CAL=1, BOFF=0 in the DAC_CR1 register; set DAC_OS_TRIMCR_VREFBUF[5:0] in the DAC_CALR register to 6'h11.
2. Fix DAC_DOR.DACDOR[11:0]=12'h7FF (2047) for the DAC, adjust the value of DAC_OS_TRIMCR_VREFBUF[5:0], and observe the value of the DAC_DOUT register. When the register value changes from 0 to 1, record the current value of DAC_OS_TRIMCR_VREFBUF[5:0].

14.3.5. DAC data format

Depending on the selected configuration mode, the data has to be written in the specified register as described below:

- DAC data alignment has three modes:
 - 8-bit data right-aligned: Users must write data to bits DAC_DHR8R[7:0] (actually stored in bits DACDHR[11:4]).
 - 12-bit data left-aligned: Users must write data to bits DAC_DHR12L[15:4] (actually stored in bits DACDHR[11:0]).
 - 12-bit data right-aligned: Users must write data to bits DAC_DHR12R[11:0] (actually stored in bits DACDHR[11:0]).

Based on the operation of the DAC_DHRyyyx register, after the corresponding shift, the written data is transferred to the DACDHR register (DACDHR is the internal data storage register). Subsequently, the content of the DACDHR register is either automatically transferred to the DACDOR register or transferred to the DACDOR register via software trigger or external event trigger.

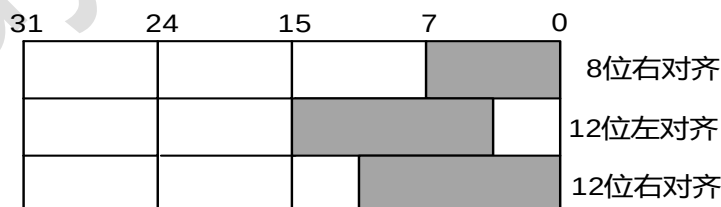


Figure 14-2 Data registers in single DAC channel mode

14.3.6. DAC conversion

Data cannot be written directly to the DAC_DOR register. Any data output to the DAC must be written to the DACDHR register (data is actually written to the DAC_DHR8R, DAC_DHR12L, or DAC_DHR12R registers).

If hardware trigger is not selected (TEN bit in the DAC_CR1 register is '0'), the data stored in the DACDHR register will be automatically transferred to the DACDOR register after one APB clock cycle. If hardware trigger is selected (TEN bit in register DAC_CR1 set to '1'), data transfer completes 3 APB clock cycles after the trigger occurs.

Once data is loaded from the DACDHR register into the DACDOR register, the output becomes valid after the settling time $t_{SETTLING}$ (refer to datasheet for value), which varies depending on power supply voltage and analog output load.

Note: When the DAC is disabled, the DACDOR value is not output to analog.

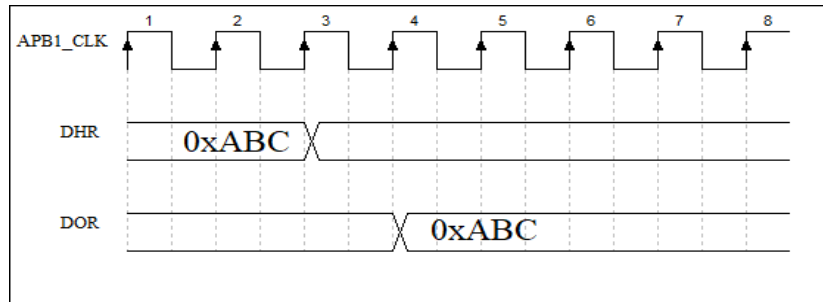


Figure 14-3 Timing diagram for conversion when TEN = 0

14.3.7. DAC output voltage

The digital input is linearly converted to an analog voltage output through the DAC, ranging from 0 to V_{REFP} .

The analog output voltages on each DAC channel pin are determined by the following equation:

$$\text{DAC output} = V_{REFP} \times (\text{DACDOR} / 4096).$$

14.3.8. DAC Trigger Selection

If the TEN bit is set to 1, DAC conversion can be triggered by an external event (timer counter, external interrupt line). The configuration control bits TSEL[2:0] can select one of 8 trigger events to initiate DAC conversion, as shown in the table below (this product supports 7 valid triggers).

Table 14-1 DAC Trigger Selection

Source	Type	TSEL[2:0]
Timer 6 TRGO event	Internal signal from on-chip timers	000
Timer 3 TRGO event		001
Timer 7 TRGO event		010
Timer 15 TRGO event		011
Timer 2 TRGO event		100
Reserved		101
EXTI Line9	External pin	110
SWTRIG (software trigger)	Software control bit	111

Each time the DAC interface detects a rising edge output from the selected timer TRGO or external interrupt line 9, the most recent data stored in the DACDHR register is transferred to the DACDOR register. After 3 APB clock cycles, the DACDOR register is updated with the new value.

If software trigger is selected, conversion begins once the DAC_CR2.SWTRIG bit is set to '1'. After data is transferred from the DACDHR register to the DACDOR register, the SWTRIG bit is automatically cleared to '0' by hardware.

Note:

1. Do not change TSEL[2:0] bits when EN is '1'.
2. If software trigger is selected, data transfer from the DACDHR register to the DACDOR register requires only 1 APB clock cycle.
3. The trigger frequency must not exceed 1 MHz.

14.3.9. DMA functionality**14.3.9.1. DMA request**

The DAC has DMA capability.

If the DMAEN bit is set to '1', a DMA request is generated 3 clock cycles after an external trigger (not software trigger) occurs, and then the data in the DACDHR register is transferred to the DACDOR register.

14.3.9.2. DMA underrun

The DAC's DMA request has no buffer queue. Therefore, if a second external trigger arrives before the response to the first external trigger (first request) is received, no new DMA request will be issued, and the DMA underrun flag DMAUDR in the DAC_SR register will be set to '1' to report the error condition. The DMA data transfer is then disabled and no more DMA requests are processed. The DAC channelx continues to convert old data.

The software shall clear the DMAUDR flag by writing '1', disable the enable bit of the used DMA channel, and reinitialize the DMA and DAC channels to properly restart DMA transfer. Meanwhile, the software should adjust the DAC trigger conversion frequency or reduce the DMA workload to prevent DMA underflow from occurring again. Finally, DAC conversion can be resumed by enabling the DMA channel and configuring the external trigger.

For each DAC channel, if the corresponding DMAUDRIE bit in the DAC_CR1 register is enabled, an interrupt will also be generated.

14.3.10. DAC Noise Generation

A linear feedback shift register (LFSR) can be used to generate pseudo-noise with varying amplitudes. Set the WAVE[1:0] bits to '01' to enable the DAC noise generation function. The preload value of the LFSR is 0xAAA. This register is updated three APB cycles after each trigger event, following a specific calculation algorithm.

By setting the MAMP[3:0] bits in the DAC_CR1 register, part or all of the LFSR data can be masked. The resulting LFSR value is added to the DACDHR value. If overflow occurs, the maximum value retained by the register is written to DACDOR; otherwise, the calculated value is written to DACDOR. If the register LFSR value is 0x000, a '1' (anti-lock mechanism) is injected.

Setting the WAVE[1:0] bits to '0' resets the LFSR waveform generation algorithm.

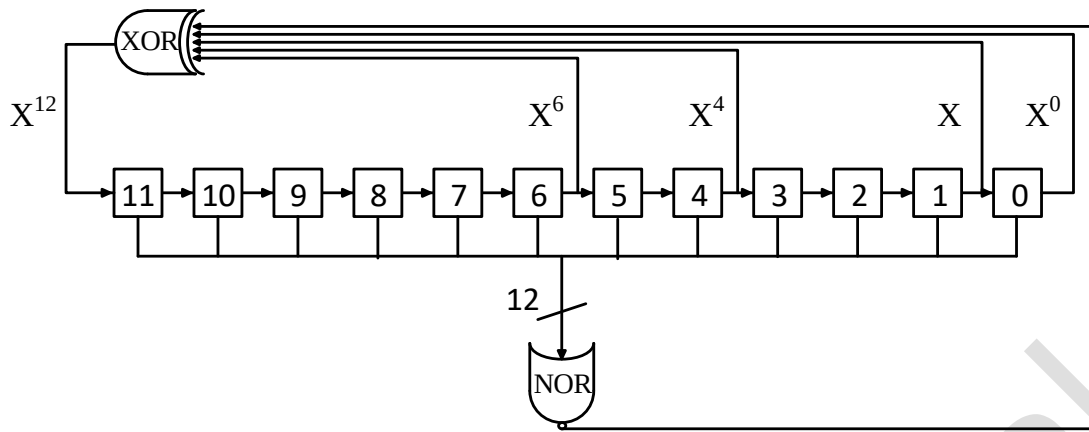


Figure 14-4 DAC LFSR Algorithm

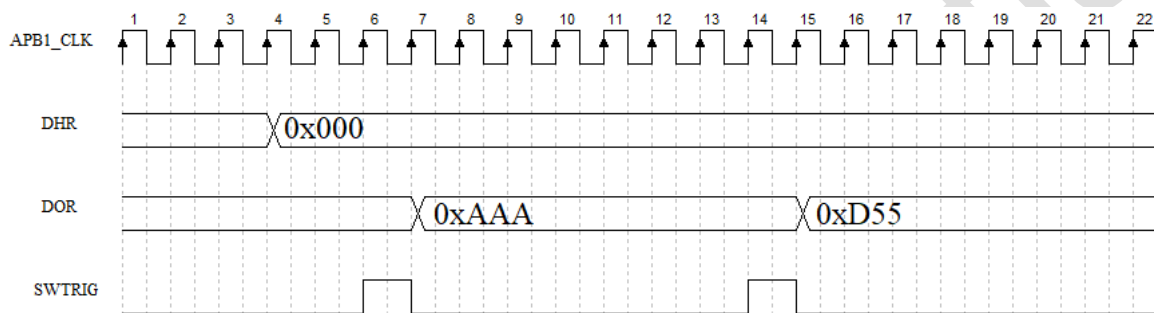


Figure 14-5 DAC Conversion with LFSR Waveform Generation (Software Trigger Enabled)

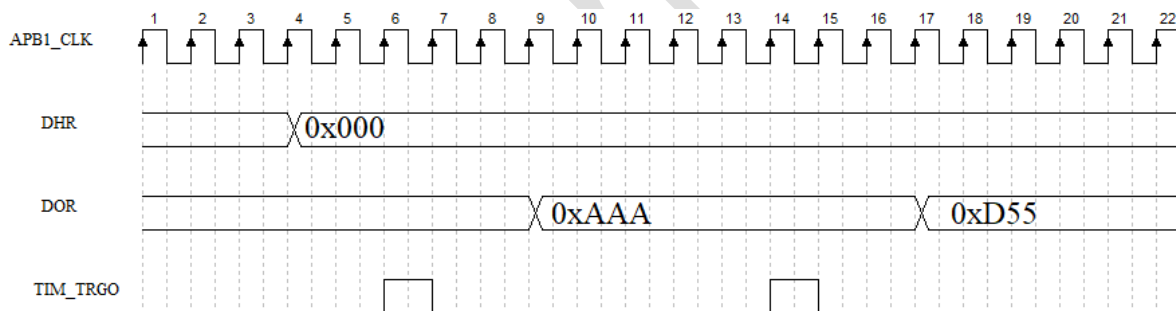


Figure 14-6 DAC Conversion with LFSR Waveform Generation (Hardware Trigger Enabled)

Note: 1. To generate noise, the DAC trigger must be enabled by setting the TEN bit in the DAC_CR1 register to '1';

2. If the DHR value changes, the LFSR register value will reset to 0xAAA;

3. If TEN transitions from '1' to '0', the LFSR register value will also reset to 0xAAA.

14.3.11. DAC Triangle Wave Generation

A small amplitude triangular wave can be added to DC or slowly changing signals. Set the WAVE[1:0] bits to '1x' to enable the DAC triangle wave generation function. Set the MAMP[3:0] bits in the DAC_CR1 register to select the amplitude of the triangle wave. An internal triangle counter is incremented three APB clock cycles after each trigger event. Once the configured amplitude is reached, the counter is decremented down to 0, then incremented again and soon. When the sum of the set base value and the maximum MAMP amplitude exceeds the maximum value of the DACDOR

register, the counter starts decrementing once DACDOR reaches its maximum value. After decrementing to the base value, it begins incrementing again, repeating this cycle indefinitely.

Setting WAVE[1:0] bits to '0' can reset the triangular wave generation.

Note: When the base value is the maximum value of register DACDOR, the output of DACDOR will be its maximum value regardless of MAMP.

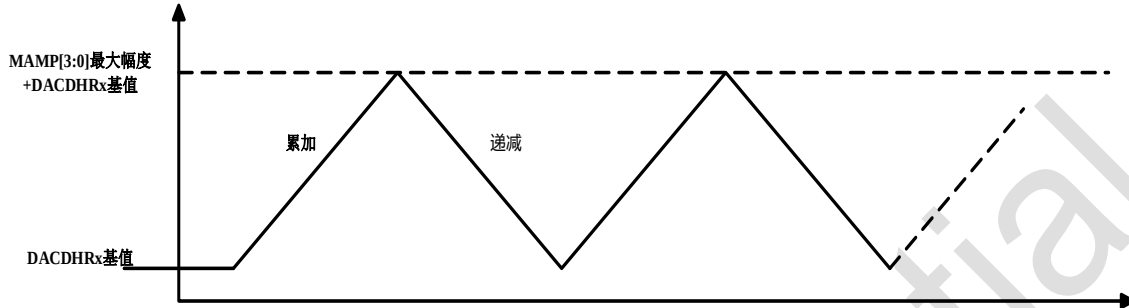


Figure 14-7 DAC Triangular Wave Generation

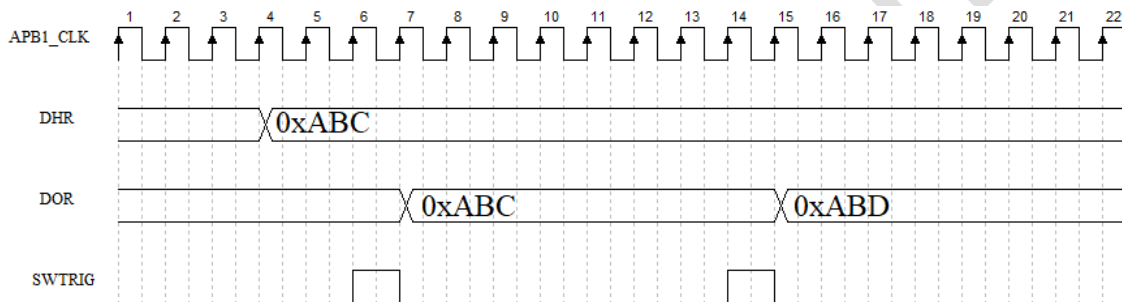


Figure 14-8 DAC conversion with triangular generation (software trigger enabled)

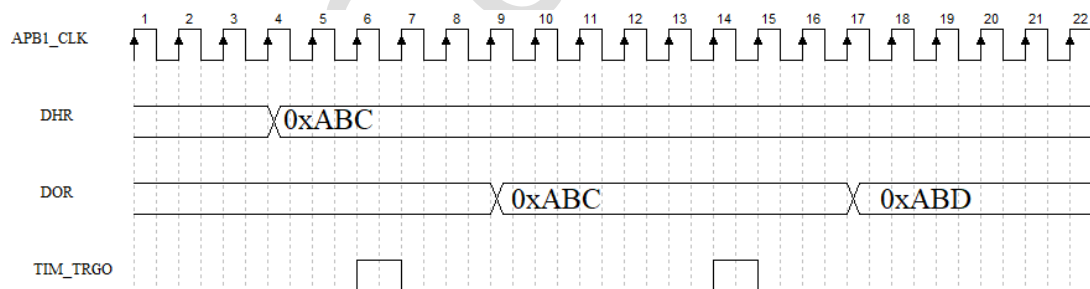


Figure 14-9 DAC conversion with triangular generation (hardware trigger enabled)

Note:

1. To generate a triangular wave, the DAC trigger must be enabled by setting the TEN bit of the DAC_CR1 register to '1';
2. The MAMP[3:0] bits must be set before enabling the DAC; otherwise, their values cannot be modified;
3. If the value of DACDHR changes, the triangular wave counter resets to 0;
4. If TEN changes from '1' to '0', the triangular wave counter also resets to 0.

14.3.12. DAC output clock

The DAC output clock (DAC_CLK) is the clock for sampling DACDOR data on the analog side, and this clock is only valid when EN is '1'.

When hardware trigger is selected, it is set to '1' after 5 APB cycles following the rising edge of the hardware trigger and cleared to '0' after holding for 3 APB cycles;

When software trigger is selected, it is set to '1' after 3 APB cycles following the rising edge of the software trigger and cleared to '0' after holding for 3 APB cycles;

When no trigger is selected, it is set to '1' after 2 APB cycles following a data change and cleared to '0' after holding for 3 APB cycles.

Note: One APB cycle after TEN changes from '1' to '0', the data in DACDOR will be updated to the data in the DACDHR register. If the data in the DACDHR register has been updated, a DAC_CLK will be generated when DACDOR is updated to the value of DACDHR; otherwise, no DAC_CLK will be generated.

14.3.13. DAC output

The DAC result can be externally output to an IO or internally to a comparator, op-amp, or ADC. The output control is as shown in the following table:

Table 14-2 DAC output

EN	BOFF	INTEREN	DACCEN	External output	Internal output	Operating mode
1	1	1	0	High impedance	the BOFF output	Internal BOFF output
1	0	X	1	Non-BOFF output	Non-BOFF output	DAC calibration mode
1	0	X	0	Non-BOFF output	High impedance	External non-BOFF output
1	1	0	0	the BOFF output	High impedance	External BOFF output
0	X	X	X	High impedance	High impedance	Disable

14.4. DAC registers

14.4.1. DAC control register 1 (DAC_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUF_CAL	DACCEN	DMAUDRI	DMAEN	MAMP[3:0]				WAVE[1:0]		TSEL[2:0]			TEN	BOFF	EN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	BUF_CAL	RW	0	DAC internal calibration enable 0: Internal calibration disabled 1: Internal calibration enabled
14	DACCEN	RW	0	DAC external calibration enable (via ADC). This bit is set and cleared by the software. 0: External calibration (via ADC) disabled; 1: External calibration (via ADC) enabled;
13	DMAUDRI	RW	0	DAC DMA underflow interrupt enable. This bit is set and cleared by the software. 0: DAC DMA underflow interrupt disabled; 1: DAC DMA underflow interrupt enabled.

12	DMAEN	RW	0	DAC DMA enable. This bit is set and cleared by the software. 0: DAC DMA mode disabled; 1: DAC DMA mode enabled.
11:8	MAMP[3:0]	RW	0	DAC mask/amplitude selector. These bits are written by software to select mask in wave generation mode or amplitude in triangle generation mode. 0000: LSFR bit 0 not masked/triangle wave amplitude value equals 1 0001: LSFR bits [1:0] not masked/triangle wave amplitude value equals 3 0010: LSFR bits [2:0] not masked/triangle wave amplitude value equals 7 0011: LSFR bits [3:0] not masked/triangle wave amplitude value equals 15 0100: LSFR bits [4:0] not masked/triangle wave amplitude value equals 31 0101: LSFR bits [5:0] not masked/triangle wave amplitude value equals 63 0110: LSFR bits [6:0] not masked/triangle wave amplitude value equals 127 0111: LSFR bits [7:0] not masked/triangle wave amplitude value equals 255 1000: LSFR bits [8:0] not masked/triangle wave amplitude value equals 511 1001: LSFR bits [9:0] not masked/triangle wave amplitude value equals 1023 1010: LSFR bits [10:0] not masked/triangle wave amplitude value equals 2047 ≥1011: LSFR bits [11:0] not masked/triangle wave amplitude value equals 4095
7:6	WAVE[1:0]	RW	0	DAC noise/triangle wave generation enable. These bits are set and cleared by the software. 00: Wave generator off 01: Noise wave generator enabled 1x: Enable triangle wave generator
5:3	TSEL[2:0]	RW	0	DAC trigger selection. These 3 bits are used to select the external trigger event for the DAC. 000: TIM6 TRGO event 001: TIM3 TRGO event 010: TIM7 TRGO event 011: TIM15 TRGO event 100: TIM2 TRGO event 101: Reserved 110: External interrupt line 9 111: Software trigger Note: These 3 bits can only be used when TEN = 1 (DAC trigger enabled).
2	TEN	RW	0	DAC trigger enable. This bit is set and cleared by software to enable/disable the DAC trigger. 0: Disable DAC trigger. Data written to the DACDHR register is transferred to the DACDOR register after 1 APB clock cycle. 1: Enable DAC channel 1 trigger. Data written to the DACDHR register is transferred to the DACDOR register after 3 APB clock cycles. Note: If software trigger is selected, data written to the DAC_DHR register only requires one APB clock cycle to be transferred to the DACDOR register.
1	BOFF	RW	0	Disable DAC output buffer. This bit is set and cleared by software to enable/disable the DAC output buffer. 0: Enable DAC output buffer 1: Disable DAC output buffer
0	EN	RW	0	DAC enable This bit is set and cleared by software to enable/disable the DAC. 0: Disable DAC 1: Enable DAC

14.4.2. DAC control register 2 (DAC_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INTEREN	VREFSEL	SW TRIG
													RW	RW	W

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2	INTEREN	RW	0	DAC internal output enable. 0: DAC outputs to external IO 1: DAC outputs to internal analog module
1	VREFSEL	RW	0	DAC reference voltage selection. 0: Select V_{REFP} as reference voltage (connect to V_{CCA} if V_{REFP} is unavailable) 1: Select V_{REFBUF} as reference voltage
0	SWTRIG	W	0	DAC software trigger This bit is set and cleared by software to enable/disable software trigger. 0: Disable DAC software trigger; 1: Enable DAC software trigger. Note: Once the data in register DACDHR is transferred to register DACDOR, (after 1 APB clock cycle) this bit is cleared by hardware.

14.4.3. DAC 12-bit right-aligned data holding register (DAC_DHR12R)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Res.	Res.	Res.	Res.	DACDHR[11:0]										
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	DACDHR[11:0]	RW	0	DAC 12-bit right-aligned data This bit is written by software to represent the 12-bit DAC data.

14.4.4. DAC 12-bit left-aligned data holding register (DAC_DHR12L)

Address offset: 0x0C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:4	DACDHR[11:0]	RW	0	DAC 12-bit left-aligned data. This bit is written by software to represent the 12-bit DAC data.
3:0	Reserved			Reserved

14.4.5. DAC 8-bit right-aligned data holding register (DAC_DHR8R)

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	DACDHR[7:0]	RW	0	8-bit right-aligned data for DAC This bit is written by software and represents the 8-bit data of the DAC.

14.4.6. DAC data output register (DAC_DOR)

Address offset: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Res.	Res.	Res.	Res.	DACDOR[11:0]										
				R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	DACDOR[11:0]	R	0	DAC output data. This bit is read-only and represents the output data of the DAC.

14.4.7. DAC status register (DAC_SR)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	DMAUDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
		RC_W1													

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13	DMAUDR	RC_W1	0	DAC DMA underflow flag. This bit is set by hardware and cleared by writing 1 in software. 0: No DAC DMA underflow occurred 1: DAC DMA underflow occurred Note: Software can only write '1' to clear '0' after DMAUDR is set to '1'
12:0	Reserved	-	-	Reserved

14.4.8. DAC calibration register (DAC_CALR)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DAC_CR[4:0]				
											RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DAC_DOUT	DAC_OS_TRIMCR_VREFBUF[5:0]							DAC_OS_TRIMCR_VCC[5:0]				

			R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	Reserved
20:16	DAC_CR[4:0]	RW	5'h0	DAC reserved function register
15:13	Reserved	-	-	Reserved
12	DAC_DOUT	R	1'b0	Data for DAC when selecting internal Buffer calibration.
11:6	DAC_OS_TRIMCR_VREFBUF[5:0]	RW	6'h0	Offset calibration value when DAC reference voltage selects V _{REFBUF} .
5:0	DAC_OS_TRIMCR_VCC[5:0]	RW	6'h0	Offset calibration value when DAC reference voltage selects V _{REFP} (V _{CCA}).

15. Comparator (COMP)

15.1. Introduction to COMP

The chip integrates two general-purpose comparators (COMP), namely COMP1 and COMP2. The COMP1/2 module can be used as a separate module or in combination with timer.

The INP/INM used for comparator analog functions must be configured in analog mode in the GPIO registers, and the corresponding bits in the SYSCFG.P*_ANA2EN register must be enabled.

The COMP features:

- Wake-up from low-power mode triggered by analog signals
- Analog signal conditioning
- When connected to the PWM output of a timer, it forms a cycle-by-cycle current control loop

15.2. COMP main features

- Each comparator has configurable positive or negative inputs for flexible voltage selection. Input pin sources include:
 - Multiplexed I/O pins
 - Output of the temperature sensor
 - Supports 64-step voltage division of internal reference voltages V_{REFBUF} and V_{CCA}
 - V_{REFINT}
 - DAC output as INP or INM input
 - OPA output as INP input
- Programmable hysteresis
- Programmable speed and power consumption
- Outputs can be connected to I/O or timer/low-power timer inputs as triggers:
 - OCREF_CLR event (cycle-by-cycle current control)
 - Braking for fast PWM shutdown
 - Timer input capture
 - Low-power timer input trigger
- COMP1 and COMP2 may be combined into a window comparator
- Each comparator has interrupt generation capability for waking the chip from low-power modes (sleep and all stop modes) via EXTI
- Configurable digital filter
- Blank source comparator output
- Write protection for configuration registers (LOCK function)

15.3. COMP functional description

15.3.1. COMP block diagram

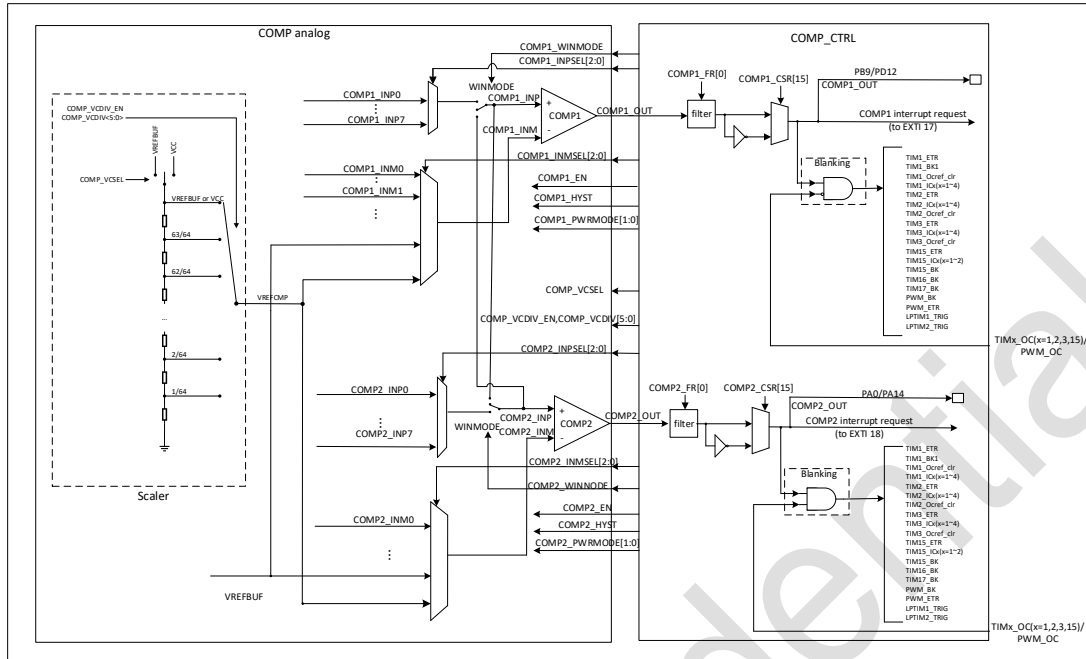


Figure 15-1 COMP block diagram

15.3.2. COMP interface and internal signals

I/Os used as comparator inputs must be configured in analog mode in the GPIO registers, and the corresponding PAD in the SYSCFG.P*_ANA2EN register must be enabled.

The comparator output can be connected to I/O pins through the alternate function (AF) of GPIO.

The output can also be internally connected to inputs of various timers for the following purposes:

- Connecting to brake input for emergency shutdown of PWM signals
- Cycle-by-cycle current control using OCREF_CLR input
- Input capture of timers
- Input triggering of low-power timers

The non-inverting input assignment of Comparator 1 is described in the INPSEL[2:0] bit field of the COMP1_CSR register, and the inverting input assignment is described in the INMSEL[2:0] bit field of the COMP1_CSR register. The non-inverting input assignment of Comparator 2 is described in the INPSEL[2:0] bit field of the COMP2_CSR register, and the inverting input assignment is described in the INMSEL[2:0] bit field of the COMP2_CSR register.

15.3.3. COMP reset and clocks

The COMP module has the following clock sources:

PCLK (APB clock), system bus clock

COMP clock, used to simulate the clock of the circuit after the output of the comparator (analog output latch circuit, glitch filter circuit, etc.), can be selected as PCLK, LSE or LSI. When operation in Stop mode is required, LSE or LSI must be selected.

The reset signal sources for the COMP module are:

- 1) APB reset, system reset
- 2) COMP module soft reset (RCC_APBSTR2.COMPxRST)

15.3.4. Filtering function of COMP

If the working environment of the chip is harsh, the output of the comparator will appear noisy signals. By setting COMPx_FR.FLTENx to enable the digital filter module, noise signals in the comparator output waveform with pulse widths shorter than the time set by COMPx_FR.FLTCNT[15:0] can be filtered out. If the digital filter module is disabled, the output signal of the digital filter module is the same as the input signal.

Note: The COMP filter time setting and enabling the filter should be completed before enabling COMPx_CSR.COMPx_EN.

The filtering schematic is as follows:

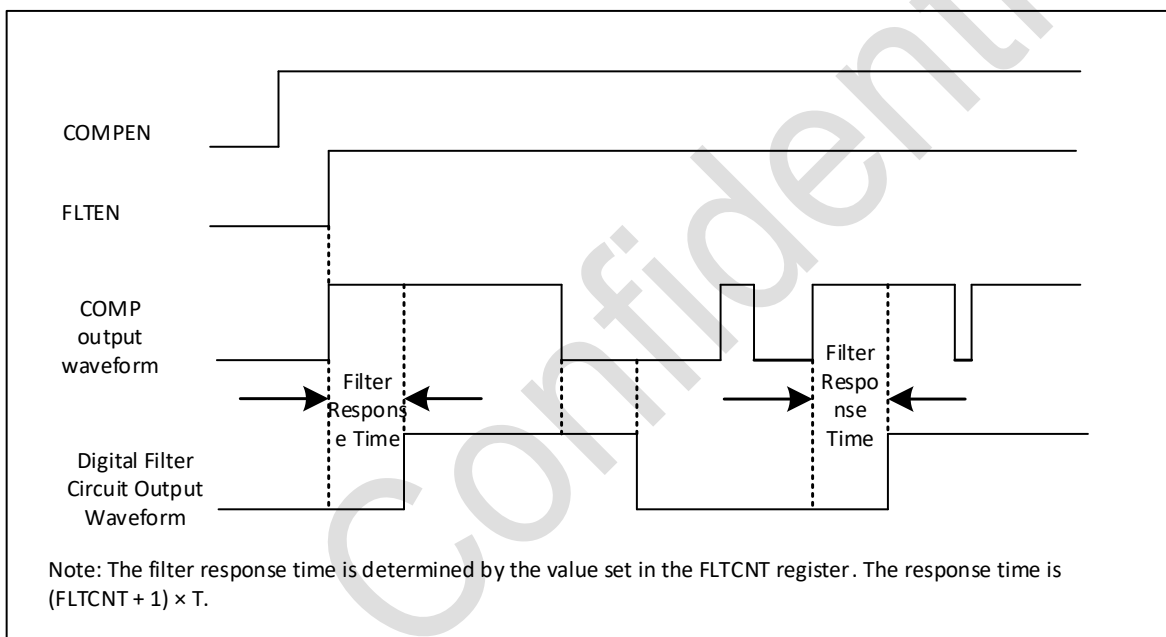


Figure 15-2 COMP filtering schematic diagram

15.3.5. COMP output blanking

The purpose of the blanking function is to prevent current regulation from tripping due to short current spikes at the beginning of the PWM cycle (typically recovery current in the anti-parallel diodes of power switches). This is achieved by setting a blanking window defined by the timer output compare signal. The blanking source for each comparator channel is individually selected by software through configuring the BLANKSEL[2:0] bits in the COMPx_CSR register. The complement of the blanking signal is logically ANDed with the comparator output to generate the comparator channel x output.

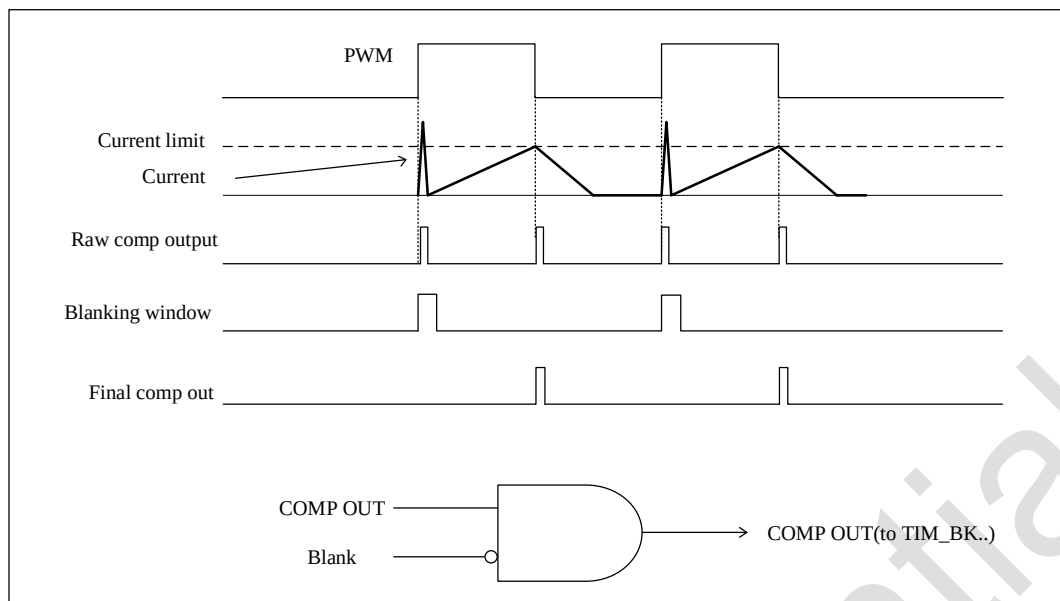


Figure 15-3 Comparator Blanking

15.3.6. COMP locking mechanism

Comparators can be used for safety-critical applications such as overcurrent and thermal protection. For applications with specific functional safety requirements, the comparator configuration can be protected against unintended modifications (e.g., when the program counter is corrupted).

For this purpose, the comparator control and status registers can be write-protected (read-only).

Once the comparator channel configuration is complete, its LOCK bit is set to 1, making the entire register read-only, including the LOCK bit.

The write protection can only be removed by a chip reset signal.

15.3.7. Window comparator

The window comparator monitors whether an analog voltage falls within the range defined by upper and lower threshold limits.

Two embedded comparators can be utilized to create window comparator. The monitored analog voltage is connected to the non-inverting (+) inputs of both comparators, while the upper and lower threshold voltages are connected to the inverting inputs (-) of the respective comparators.

By enabling the WINMODE bit, the non-inverting (+) inputs of both comparators can be connected together, saving one I/O pin.

For example, when WINMODE of Comparator 2 is set to 1, the comparator's positive input will connect to COMP1INP; when WINMODE is cleared to 0, the positive input will connect to COMP2INP. This approach saves one I/O pin.

Note: The WINMODE mode of both COMPs cannot be enabled simultaneously.



15.3.8. Hysteresis

To avoid unintended output transitions in noisy signal conditions, the comparator can enable hysteresis functionality (by setting the HYST bit in COMP1_CSR and COMP2_CSR, the hysteresis for COMP1 and COMP2 can be enabled respectively).



15.3.9. Low-power modes

The table below shows the comparator operation in various low-power modes. In low-power mode, to reduce power consumption, the PWRMODE bit in the COMPx_CSR register must be configured to 1.

Note: INP or INM supports selecting outputs from V_{REFBUF} , V_{REFINT} , TS, DAC, or OPA. When V_{REFBUF} is selected as the reference supply, it may consume significant power. In low-power mode, proper configuration is required.

Table 15-1 Comparator characteristics in low power mode

Mode	Description
Sleep	No impact on the comparator. Comparator interrupts cause the device to exit the Sleep mode.
Low-power run	No effect on the comparator.
Low power sleep	No impact on the comparator.

	Comparator interrupt can wake the device from low-power sleep mode.
Stop	No impact on the comparator. Comparator interrupt can wake the device from all stop modes.
Standby	The comparator controller powers down and must be reinitialized after exiting standby mode.

15.4. COMP interrupts

The comparator outputs are internally connected to the extended interrupts and events controller. Each comparator has a dedicated EXTI Line (Line17, Line18) capable of generating interrupts or events. This mechanism can also be used to wake up from low-power modes.

To enable the COMPx interrupt, follow this sequence:

1. Configure the EXTI Line corresponding to the COMPx output event in interrupt mode and enable it, then select rising edge, falling edge, or both as active.
2. Configure and enable the NVIC IRQ channel mapped to the corresponding EXTI Line.
3. Enable COMPx.

15.5. COMP registers

To configure the COMP1_CSR, COMP1_FR, and COMP_CCSR registers, enable RCC_APBENR1.COMP1EN to turn on the APB clock for COMP1.

To configure the COMP2_CSR and COMP2_FR registers, enable RCC_APBENR1.COMP2EN to turn on the APB clock for COMP2.

15.5.1. Comparator 1 control and status register (COMP1_CSR)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	COMP_OUT	INT_OUT_SEL	Re s.	Res.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	PWRMODE	Re s.	HYST
RW	R	RW											RW		RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLARITY	BLANKSEL [2: 0]			WINMODE	Re s.	Re s.	Re s.	INPSEL[2:0]			INMSEL[2:0]			Re s.	COMP1_EN
RW	RW	RW	RW	RW	-	-	-	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP1_CSR register lock Software set, system reset and clear. When set, it locks all bits of the COMP1_CSR register. 0: Unlocked, can read and write the entire register 1: Lock, entire register read-only
30	COMP_OUT	R	0	COMP1 output This bit is read-only and reflects the output level of COMP1 after polarity selection and blanking.
29	INT_OUT_SEL	RW	0	Comparator 1 internal digital output control. 0: The internal digital output of the comparator selects the direct output of the analog comparator (can be blanked), without filtering or digital synchronization. 1: The internal digital output of the comparator matches the external output (can be filtered, can be blanked).
28:19	Reserved	-	-	Reserved
18	PWRMODE	RW	0	Comparator 1 power mode selector Software readable and writable, selects power consumption and the resulting speed of COMP1. 0: High speed

				1: Medium speed
17	Reserved	-	-	Reserved
16	HYST	RW	0	Comparator 1 hysteresis selector 0: Hysteresis function turned off 1: Hysteresis function enabled
15	POLARITY	RW	0	COMP1 polarity selection Software can be read and written (if not locked) 0: Non-inverted 1: Inverted
14:12	BLANKSEL [2: 0]	RW	0	Comparator 1 Blanking Signal Selection 000: Blanking not enabled 001: TIM1_OC4 010: TIM2_OC3 011: TIM3_OC3 100: TIM15_OC2 101: PWM_OC3 110: Reserved 111: Reserved
11	WINMODE	RW	0	Non-inverting input selection for COMP1 in window mode Software can be read and written (if not locked) 0: Selected via COMP1's INPSEL[2:0] 1: Selects COMP2's COMP2_INP signal Note: The WINMODE mode of both COMPs cannot be enabled simultaneously.
10:8	Reserved	-	-	Reserved
7:5	INPSEL[2:0]	RW	00	Comparator 1 signal selector for non-inverting input 000: COMP1_INP0(OPA_VOUT) 001: COMP1_INP1(DAC_VOUT) 010: COMP1_INP2(PC15) 011: COMP1_INP3(PD0) 100: COMP1_INP4(PD1) 101: COMP1_INP5 (reserved) 110: COMP1_INP6 (reserved) 111: COMP1_INP7 (reserved)
4:2	INMSEL[2:0]	RW	00	Comparator 1 signal selector for inverting input 000: COMP1_INM0(PC5) 001: COMP1_INM1(PA1) 010: COMP1_INM2(DAC_VOUT) 011: COMP1_INM3(Resistor divider output V _{REFCMP}) 100: COMP1_INM4(TS_VIN)(Requires enabling the ADC module's ADC_CCR.TS_EN register) 101: COMP1_INM5(V _{REFINT})(Requires enabling the ADC module's ADC_CCR.VREFINT_EN register) 110: COMP1_INM6(V _{REFBUF})(Requires enabling the V _{REFBUF} module's VREFBUF_EN register) 111: COMP1_INM7(PA13)
1	Reserved	-	-	Reserved
0	COMP1_EN	RW	0	Comparator 1 enable Software can be read and written (if not locked) 0: Disabled 1: Enabled

15.5.2. COMP1 filter register (COMP1_FR)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT1 [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN1
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT1 [15:0]	RW	0x0	Comparator 1 sampling filter counter

				The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of samples reaches the filter count value, the results are output consistently. Sample count cycle = FLTCNT [15: 0]
15:1	Reserved	-	-	Reserved
0	FLTEN1	RW	0x0	Comparator 1 digital filter function configuration 0: Digital filtering function disabled 1: Digital filtering function enabled Note: This bit must be set when COMP1_EN is 0.

15.5.3. COMP Common Control and Status Register (COMP_CCSR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SSEL	VCDIV[5:0]					VREFCMP_EN	
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	SSEL	RW	0	VREFCMP voltage source selection 0: VREFBUF 1: VCCA
6:1	VCDIV[5:0]	RW	0	COMP1,COMP2 divider selection 00_0000: 1/64 VREF 00_0001: 2/64 VREF 00_0010: 3/64 VREF ... 11_1110: 63/64 VREF 11_1111: VREF
0	VREFCMP_EN	RW	0	VREFCMP enable 0: Disabled 1: Enabled

15.5.4. COMP2 Control and Status Register (COMP2_CSR)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	COMP_OUT	INT_OUT_SEL	Re s.	Res.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	PWRMODE	Re s.	HYST
RW	R	RW											RW		RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLARITY	BLANKSEL [2: 0]			WINMODE	Re s.	Re s.	Re s.	INPSEL[2:0]			INMSEL[2:0]			Re s.	COMP2_EN
RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP2_CSR register lock Software set, system reset and clear. When set, all bits of the COMP2_CSR register are locked. 0: Unlocked, can read and write the entire register 1: Lock, entire register read-only
30	COMP_OUT	R		COMP2 output This read-only flag reflects the level of the comparator 2 output before the polarity selector.
29	INT_OUT_SEL	RW	0	Comparator 2 internal digital output control. 0: Comparator internal digital output selects direct output from analog path (can be blanked), without filtering or digital synchronization 1: The internal digital output of the comparator matches the external output (can be filtered, can be blanked).

28:19	Reserved	-	-	Reserved
18	PWRMODE	RW	0	Comparator 2 power mode selector The software is readable and writable, selecting the power consumption and the resulting speed of COMP2. 0: High speed 1: Medium speed
17	Reserved	-	-	Reserved
16	HYST	RW	0	Comparator 2 hysteresis selector 0: Hysteresis function turned off 1: Hysteresis function enabled
15	POLARITY	RW	0	COMP2 polarity selection Software can be read and written (if not locked) 0: Non-inverted 1: Inverted
14:12	BLANKSEL [2: 0]	RW	0	Comparator 2 Blanking Signal Selection 000: Blanking not enabled 001: TIM1_OC4 010: TIM2_OC3 011: TIM3_OC3 100: TIM15_OC2 101: PWM_OC3 110: Reserved 111: Reserved
11	WINMODE	RW	0	Non-inverting input selection for COMP2 in window mode Software can be read and written (if not locked) 0: Selected via COMP2's INPSEL[2:0] 1: Selects the COMP1_INP signal of COMP1 Note: The WINMODE mode of both COMPs cannot be enabled simultaneously.
10:8	Reserved	-	-	Reserved
7:5	INPSEL[2:0]	RW	000	Signal selection for the non-inverting input of COMP2, readable and writable by software 000: COMP2_INP0 (DAC_VOUT) 001: COMP2_INP1 (OPA_VOUT) 010: COMP2_INP2(PB14) 011: COMP2_INP3(PC3) 100: COMP2_INP4(PC4) 101: COMP2_INP5 (reserved) 110: COMP2_INP6 (reserved) 111: COMP2_INP7 (reserved)
4:2	INMSEL[2:0]	RW	000	Signal selection for the inverting input of COMP2 000: COMP2_INM0(PA13) 001: COMP2_INM1(PB15) 010: COMP2_INM2 (DAC_VOUT) 011: COMP2_INM3 (Resistor divider output V _{REFCMP}) 100: COMP2_INM4 (TS_VIN) (Requires enabling the ADC_CCR.TS_EN register of the ADC module) 101: COMP2_INM5 (V _{REFINT}) (Requires enabling the ADC_CCR.VREFINT_EN register of the ADC module) 110: COMP2_INM6 (V _{REFBUF}) (Requires enabling the VREFBUF_EN register of the VREFBUF module) 111: COMP2_INM7(PC0)
1	Reserved	-	-	Reserved
0	COMP2_EN	RW	0	Comparator 2 enable Software can be read and written (if not locked) 0: Disabled 1: Enabled

15.5.5. COMP2 filter register (COMP2_FR)

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLT CNT2 [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN2
																RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT2 [15:0]	RW	0x0	Comparator 2 sampling filter counter The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of samples reaches the filter count value, the results are output consistently. Sample count cycle = FLTCNT [15: 0]
15:1	Reserved	-	-	Reserved
0	FLTEN2	RW	0x0	Comparator 2 digital filter function configuration 0: Digital filtering function disabled 1: Digital filtering function enabled Note: This bit must be set when COMP2_EN is 0

16. Operational amplifier (OPA)

16.1. Introduction

The OPA module is suitable for simple amplifier applications. The internal single operational amplifier can be cascaded using external resistors. The input range of the OPA is from 0V to V_{CCA} , and the output range is from 0.2V to $V_{CCA}-0.2V$.

For I/Os used as OPA analog functions, they must be configured in analog mode in the GPIO registers, and the OPA-mapped PAD in the SYSCFG.P*_ANA2EN register must be enabled.

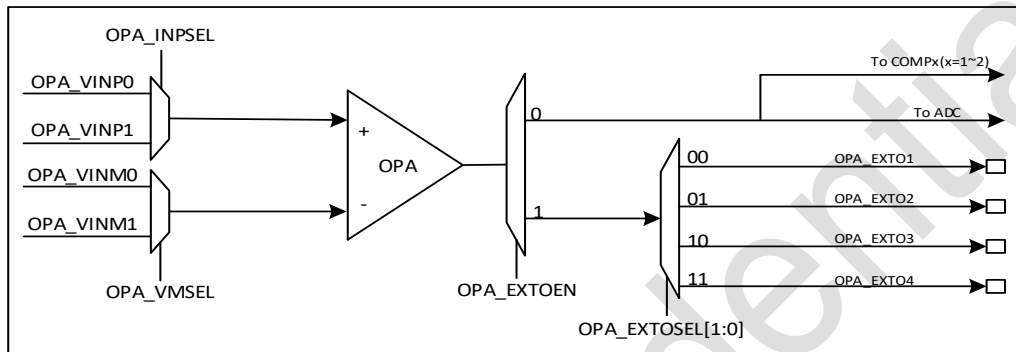


Figure 16-1 OPA block diagram

16.2. OPA main features

- One independently configured operational amplifier
- VINP and VINM can be configured to select 2 channels respectively.
- The input range of the OPA is to V_{CCA} , and the output range is from 0.2V to $V_{CCA}-0.2V$.
- Can be internally output to COMP1~2 and ADC, and externally output can be configured to select 4 I/Os.
- Can only be configured in general-purpose operational amplifier mode.

16.2.1. OPA functional description

The OPA can amplify small analog input signals by using external components to form an amplifier, with the output being the amplified signal.

16.3. OPA registers

16.3.1. OPA Output Control Register (OPA_OCR)

Offset address: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPA_EXTSEL[1:0]	OPA_EXTOE	OPA_EXTOE
.	.	.	.	.	.	.	.	.	.	.	.	.	0	N	N
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2:1	OPA_EXTSEL[1:0]	RW	0	OPA PAD External Output Selection 00: OPA external output to PD4 01: OPA external output to PD1 10: OPA external output to PC8 11: OPA external output to PC9
0	OPA_EXTOEN	RW	0	OPA output enable 0: No output internally or externally; 1: OPA external output to PAD, and internal output to COMP1~2 and ADC; Note: When this register is configured to 1, and the OPA outputs to internal COMP or ADC, the corresponding bit in the SYSCFG.Px_ANA2EN register must be configured to enable ANA_PAD2 (PC8/PC9/PD1/PD4).

16.3.2. OPA Control Register (OPA_CR)

Offset Address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res.	Res	Res	Res.	Res	Res.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	OPAE N	Res	Res	OPA_INMSE L	Res	OPA_INPSE L
.	.	.	.	.	.	.	.	.	.	RW	.	.	RW	.	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	OPAEN	RW	0	OPA module enable, active high
4:3	Reserved	-	-	Reserved
2	OPA_VINMSEL	RW	0	OPA VINM selection control. 0: OPA VINM selects PD2 1: OPA VINM selects PC1
1	Reserved	-	-	Reserved
0	OPA_VINPSEL	RW	0	OPA VINP selection control. 0: OPA VINP selects PD3 1: OPA VINP selects PC2

17. Liquid Crystal Display Controller (LCD)

17.1. Introduction

The LCD controller is a digital controller/driver designed for monochrome passive liquid crystal displays (LCDs), supporting up to 8 common terminals (COM) and 40 segment terminals (SEG) to drive 160 (4x40) or 288 (8x36) LCD picture elements. The exact number of terminals depends on the device pins described in the data manual. The LCD is composed of several segments (pixels or symbols), which can all be lit or off. Each segment contains a layer of liquid crystal molecules aligned between the two electrodes. When a voltage higher than the threshold is applied to the liquid crystal, the corresponding segment becomes visible. The segment voltage must be alternating current to avoid the surge effect in the liquid crystal (which affects display quality). After that, waveforms must be generated across the section to avoid DC.

17.2. LCD main features

The main features of the LCD are as follows:

- Highly flexible frame rate control
- Supports static, 1/2, 1/3, 1/4, 1/6, and 1/8 duty cycles
- Supports 1/2, 1/3, and 1/4 bias voltages
- Supported display modes: Type A or Type B
- Three driving waveform generation methods: internal resistor divider, external resistor divider, and external capacitor divider
 - With internal resistor voltage division, power consumption can be reduced by using software to configure the conduction time of the internal high - drive voltage - dividing resistors, thus matching the capacitance charge required by the LCD panel. The contrast of the LCD can also be configured via software to adjust the brightness of the LCD panel
 - For external resistor voltage division, the capacitance charge required by the LCD panel can be matched by adjusting the resistance value of the external resistors
 - In the case of external capacitor voltage division, the capacitance charge required by the LCD panel can be matched by using software to configure the number of capacitor driving times
- Supports LCD blinking function with configurable blinking frequencies
- Adjusts display brightness through programmable inter-frame dead time
- Unused LCD segments and public pin can be configured to digital or analog functions
- Dual buffer memory allows the data in the LCD_RAM registers to be updated at any time through the application firmware without affecting the integrity of the displayed data
 - Up to 16x32-bit register LCD data RAM
- Supports low-power modes: LCD controller can display in Sleep, Low-power Run, Low-power Sleep, and Stop modes
- Configurable frame interrupt

17.3. LCD functional description

When used as LCD I/O, must be configured in analog mode in GPIO registers.

The specific functions implemented by the LCD module are as follows:

- External low-speed crystal or internal low-speed RC clock can be selected as the operating clock (generated by RCC module)
- Generates COM port switching waveforms based on LCD clock;
- Generates raw SEG switching waveforms based on current register settings
- LCD RAM is used to store LCD display data and can also be used as general data registers
- After modifying data in LCD RAM, wait for UDD flag to be set before LCD display changes
- Generates control timing supporting analog internal boost mode

17.3.1. LCD block diagram

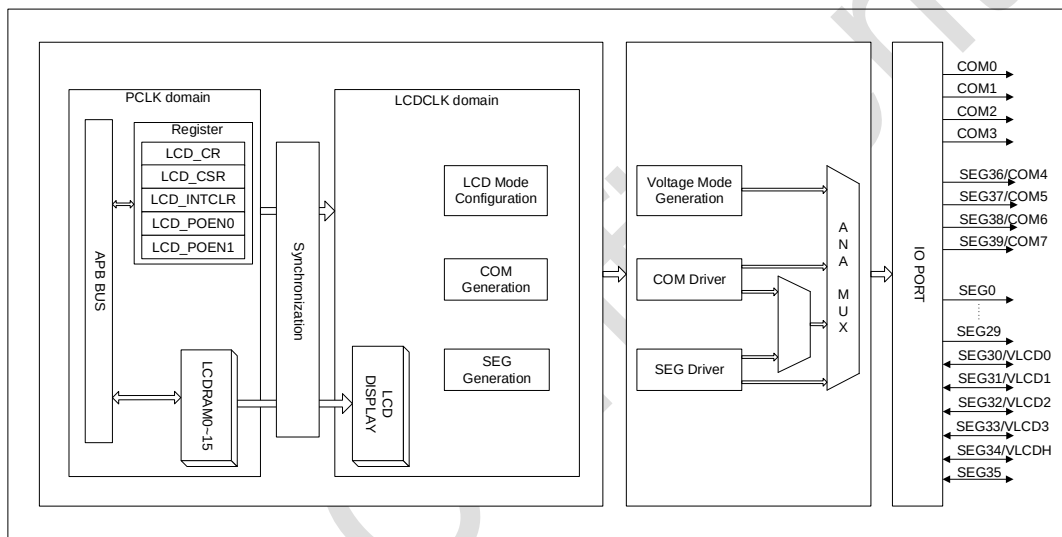


Figure 17-1 LCD Functional Block Diagram

17.3.2. LCD clock (refresh frequency)

LCD_CR.LCDCLK register can select LCD refresh frequency (frame transmission time), with four options:

00: 64 Hz

01: 128 Hz

10: 256 Hz

11: 512 Hz

After clock input (external low-speed crystal/internal low-speed RC, 32.768 kHz), frequency division is performed based on selected frequency.

17.3.3. LCD scanning waveform

Taking 1/8 Duty, 1/3 Bias as an example, TypeA and TypeB scanning waveforms are shown below.

Eight COM terminals become active sequentially. During each COM terminal's active period, SEG outputs appropriate voltage levels. Combined with COM levels applied to LCD panel, segments with larger voltage differences are displayed while those with smaller differences remain hidden.

1/3 Bias means LCD drive circuit can output four drive voltage levels, obtained by evenly distributing bias voltage.

17.3.3.1. LCD TypeA scanning waveform

The figure below only shows three COM terminals.

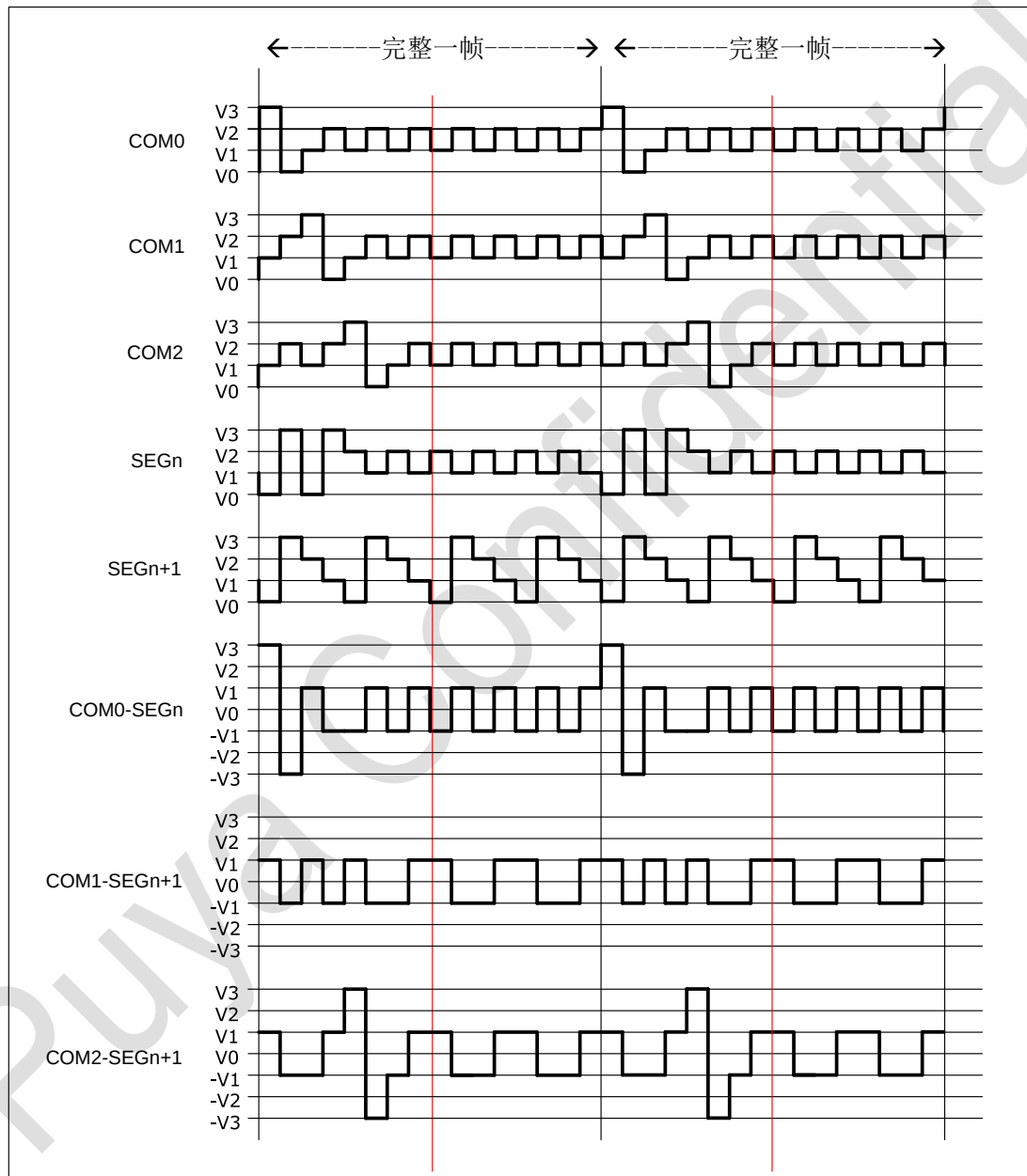


Figure 17-2 LCD 1/8 Duty, 1/3 Bias, TypeA scanning waveform

17.3.3.2. LCD TypeB scanning waveform

The figure below only shows three COM terminals.

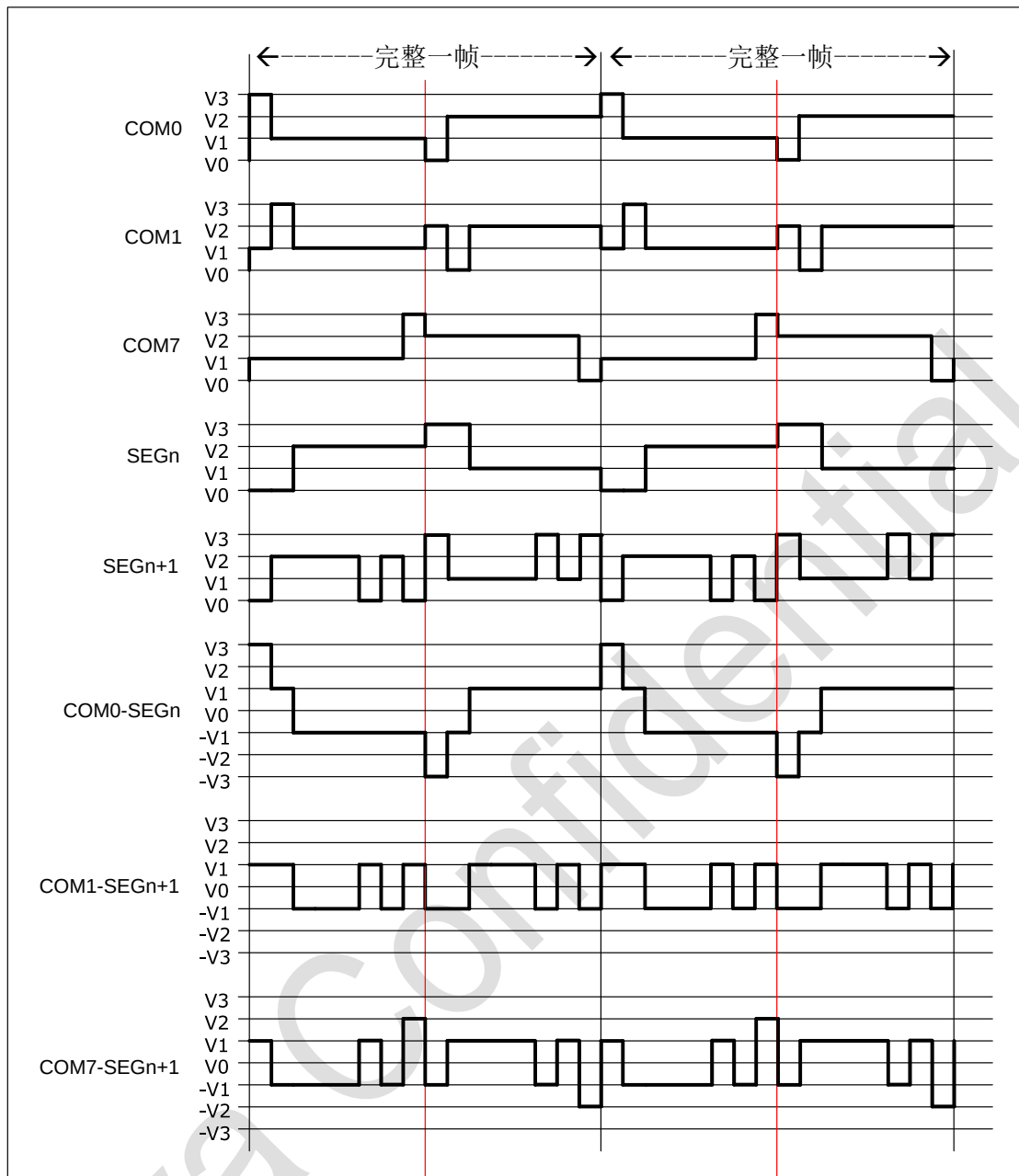


Figure 17-3 LCD 1/8 Duty, 1/3 Bias, TypeB scanning waveform

17.3.4. LCD contrast control

17.3.4.1. Waveform Amplitude

The contrast can be adjusted by modifying the waveform amplitude through the LCD_CR.CONTRAST register. Effective only when the bias voltage source is selected as internal resistor divider.

17.3.4.2. Dead Time

The contrast can be controlled by programming the inter-frame dead time. During the dead time, the COM and SEG values are set to VSS. The DEAD[2:0] bits in the LCD_CR register can be used to program a period up to seven phase cycles. This dead time reduces contrast without modifying the frame rate. The figure below shows the waveform of COM0 in typeA and typeB with 1/4 bias and DEAD[2:0]=3.

Figure 17-5 Waveform of COM0 in typeB

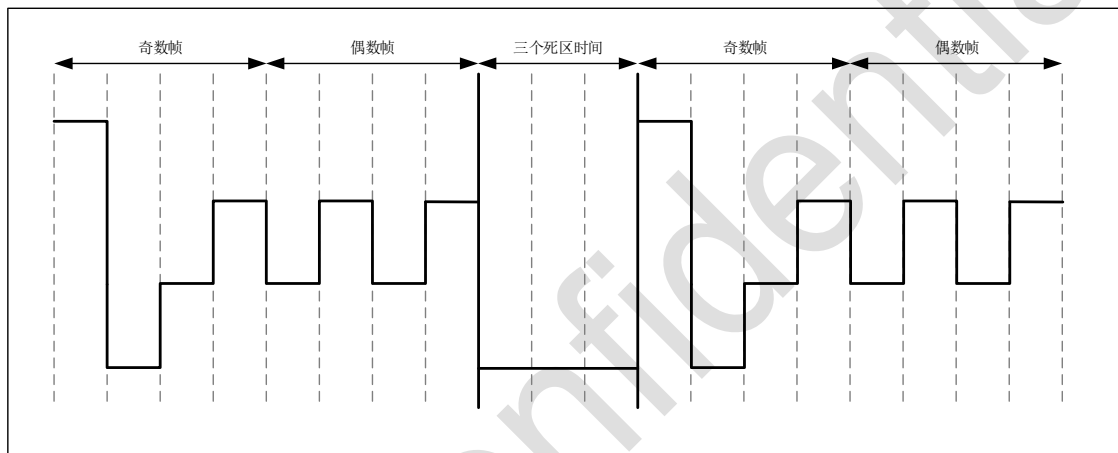


Figure 17-5 Waveform of COM0 in typeB

17.3.5. LCD Display Data Storage Mode

In different display modes, the available space of the LCD display data storage register (LCDRAM) is as follows:

17.3.5.1. Mode 1

- In 1/8 duty cycle mode, the maximum storage space is 36×8 bits.

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0																						
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0																						
COM0																																	LCDRAM0																					
COM1																																	LCDRAM1																					
COM2																																	LCDRAM2																					
COM3																																	LCDRAM3																					
COM4																																	LCDRAM4																					
COM5																																	LCDRAM5																					
COM6																																	LCDRAM6																					
COM7																																	LCDRAM7																					
																									SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32																						
																							COM0																															LCDRAM8
																							COM1																															LCDRAM9
																							COM2																															LCDRAMA
																							COM3																															LCDRAMB
																							COM4																															LCDRAMC
																							COM5																															LCDRAMD
																							COM6																															LCDRAME
																							COM7																															LCDRAMF

- In 1/6 duty cycle mode, the maximum storage space is 38×6 bits.

In 1/8 duty cycle mode, the maximum storage space is 66-0 bits.																																		
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0		
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0		
COM0																																LCDRAM		
COM1																																LCDRAM		
COM2																																LCDRAM		
COM3																																LCDRAM		
COM4																																LCDRAM		
COM5																																LCDRAM		
COM6																																LCDRAM		
COM7																																LCDRAM		
																										SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
COM0																																LCDRAM		
COM1																																LCDRAM		
COM2																																LCDRAM		
COM3																																LCDRAM		
COM4																																LCDRAM		
COM5																																LCDRAM		
COM6																																LCDRAM		
COM7																																LCDRAM		

- [illegible]

- [illegible]

- [illegible]

- [illegible]

- 1/6 duty cycle

276 / 672

- 1/4 duty cycle

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	LCDRAM0	
SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	LCDRAM1	
SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	LCDRAM2	
SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	LCDRAM3	
SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	LCDRAM4	
SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	LCDRAM5	
SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	LCDRAM6	
SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	LCDRAM7	
																							SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	LCDRAM8		
																							SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	LCDRAM9		
																							SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	LCDRAMA		
																							SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	LCDRAMB		
																							SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	LCDRAMC		
																							SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	LCDRAMD		
																							SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	LCDRAME		
																							SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	LCDRAMF		

- 1/3 duty cycle

[illegible]

[illegible][illegible]

17.3.6. LCD Bias Circuit

The bias voltage for LCD has three sources: internal resistor divider, external resistor divider, and external capacitor divider. When the internal resistor divider is selected, the chip automatically switches internal circuits to generate voltages that meet the bias and duty requirements. When the external resistor divider or external capacitor divider is selected, users need to build the corresponding circuit on the peripheral pins of the chip.

17.3.6.1. Internal resistance mode

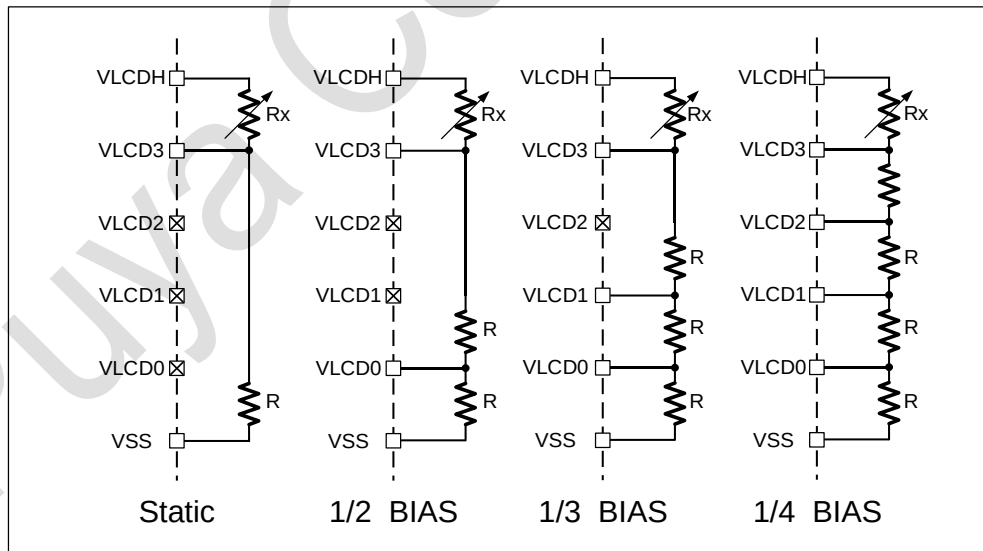
In internal resistor mode, VLCDH, VLCD3~VLCD0 can be used as LCD SEG outputs or IO ports.

In internal resistor mode, the LCD drive voltage is controlled by LCD_CR.CONTRAST, as shown in the following table:

Table 17-1 Internal resistance mode

LCD_CR.CONTRAST	VLCD(ALL bias)
0	$1.00 \times V_{CC}$
1	$0.92 \times V_{CC}$
2	$0.86 \times V_{CC}$
3	$0.8 \times V_{CC}$
4	$0.75 \times V_{CC}$
5	$0.71 \times V_{CC}$
6	$0.67 \times V_{CC}$
7	$0.63 \times V_{CC}$
8	$0.6 \times V_{CC}$
9	$0.57 \times V_{CC}$
10	$0.55 \times V_{CC}$
11	$0.52 \times V_{CC}$
12	$0.5 \times V_{CC}$
13	$0.48 \times V_{CC}$
14	$0.46 \times V_{CC}$
15	$0.44 \times V_{CC}$

17.3.6.2. External resistance mode



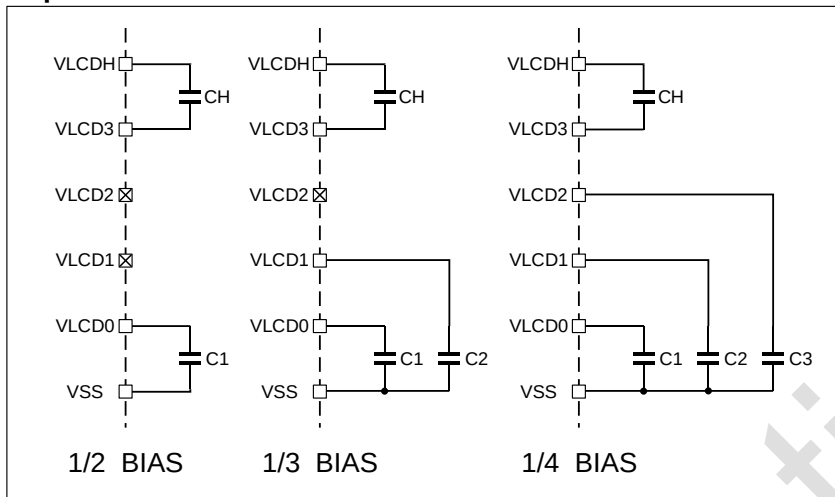
When using the external resistor drive method, external circuit support is required, occupying five additional IO pins (VLCDH, VLCD3, VLCD2, VLCD1, VLCD0) on the MCU for external resistor division.

Note:

1. Rx is an adjustable resistor used to adjust the LCD display contrast. Select the appropriate resistor value based on the LCD screen.
2. Choose the appropriate resistor R based on the LCD screen used, recommended $\geq 30K\Omega$

3. X indicates that the corresponding VLCDX channel is not used and can be used as an LCD SEG output or IO port.

17.3.6.3. External Capacitor Mode



When using the external capacitor drive method, external circuit support is required, occupying five additional IO pins (VLCDH, VLCD3, VLCD2, VLCD1, VLCD0) on the MCU for external capacitor division.

Note:

1. Recommended external capacitors: $C1=C2=C3=CH=0.1\mu F$
2. X indicates that the corresponding VLCDX channel is not used and can be used as an LCD SEG output or IO port.

17.3.7. Double Buffer Memory

Using this double buffer memory, the LCD controller ensures display continuity without requiring interrupts to control LCD_RAMx modifications.

Application software can access the first buffer level (LCD_RAMx) via the APB interface. Once LCD_RAMx is modified, the UDR flag in the LCD_CSR register is set to 1. The UDR flag (Update Display Request) requests to move updated information to the second buffer level (LCD_DISPLAYx). Before updating to LCD_DISPLAYx (after the current frame ends and before the next frame begins), LCD_RAMx is write-protected and the UDR flag remains high. Once the update is complete, another flag UDD (Update Display Done) is set to 1, and an interrupt is generated (if the UDDIE bit in the LCD_CSR register is set to 1).

17.4. LCD interrupts

Table 17-2 LCD interrupts

Interrupt event	Event flag	Enable control bit
Update Display Data Done interrupt	UDD	UDDIE
Start of Frame interrupt	SOF	SOFIE
Start of "Off" phase in each blink interrupt	BKF	BKFIE

17.5. Software configuration process

17.5.1. LCD initialization process

1. First, set the clock source: LCD module clock, where the clock source is selected by the RCC_BDCR.LSCSEL register to configure LSI or LSE. Then configure RCC_APBENR2.LCDEN=1 to enable the LCD module clock.
2. Set the LCDCLK scan clock frequency via the LCD_CR.LCDCLK register
3. Set the LCD RAM display mode selection (RAM to COM/SEG mapping) via the LCD_CSR.MODE register,
4. Set the dead time and blink frequency via the LCD_CR.DEAD and LCD_CSR.BLINKEN/BLINKCNT registers
5. Set the LCD output waveform selection (TypeA or TypeB) via the LCD_CR.COMSEG_DRV_TYPE register
6. Select V_{CCA} power mode (default)
7. Set the LCD waveform duty cycle via the LCD_CR.DUTY register

8. For non-static mode, set the bias mode via the LCD_CR.BIAS register, then select the bias voltage mode via the LCD_CR.BSEL register
9. For non-static mode, only when the bias voltage selects internal resistor division, the contrast (LCD_CR.CONTRAST) and the charge time of the fast charge mode for internal resistors (LCD_CR.FCCTL) can be set.
10. For non-static mode, only when the bias voltage selects an external capacitor, the number of external capacitor drive cycles per LCDCLK period can be set via the LCD_CSR.SWCAP_DRV_NUM register
11. Clear LCD_CR.EN=0 and LCD_CSR.SOF=0
12. Clear LCD_RAM0~15 data by writing all zeros
13. Set LCD_CSR.UDR to 1
14. Enable LCD_CR.EN=1
15. Wait until LCD_CSR.ENS becomes 1
16. Wait for hardware to configure LCD_CSR.SOF=1 (indicating first frame data write)
17. Clear LCD_CSR.SOF to 0
18. Initialization configuration completed.

17.5.2. LCD data update

The LCD RAM data update frequency must be lower than the frame frequency. Additionally, setting LCD_CSR.UDR to 1 after writing to LCD_RAM will automatically update the LCD display at the end of the current frame, and new data will be displayed in the next frame. The process is as follows:

1. Wait until LCD_CSR.UDR becomes 0
2. Write new data to LCD_RAM0~15
3. Clear LCD_CSR.UDD to 0
4. Set LCD_CSR.UDR to 1
5. Wait until LCD_CSR.UDD becomes 1
6. For subsequent data writes, return to step 1

17.5.3. Procedure for disabling the LCD

1. Set LCD_CR.EN to 0
2. Wait until LCD_CSR.ENS becomes 0
3. Clear LCD_CSR.SOF to 0

17.6. LCD registers

17.6.1. LCD Control Register (LCD_CR)

Offset address: 0x00

Reset value: 0x0086_00C2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	DEAD[2:0]			FCCTL[2:0]			COMSEG_DRV_TYPE	Res.	Res.	Res.	Res.
					RW	RW	RW	RW	RW	RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONTRAST[3:0]				Res.	BSEL[1:0]		DUTY[3:0]			BIAS[1:0]		Res.	LCD_CLK [1:0]		EN
RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:27	Reserved	-	-	Reserved
26:24	DEAD[2:0]	RW	3'b000	Dead time duration These bits are written by software to configure the length of the inter-frame dead time. During the dead time, the voltage levels of COM and SEG remain at 0 V to reduce contrast without modifying the frame rate. 000: No dead time 001: 1 phase cycle dead time 010: 2 phase cycles dead time 111: 7 phase cycles dead time When updating this register with EN=0, the update frequency should be less than or equal to 5 times the slowest clock (LCDCLK/PCLK), meaning the interval between two updates must be greater than or equal to 5 slow clock cycles.

				This register can also be updated with EN=1, but the update frequency should be less than the frame frequency, and the new value of this register is always applied at the start of the next frame.
23:21	FCCTL[2:0]	RW	3'b100	Fast charge mode, charge time control bits. (1 LCD COM cycle = 1/2 LCDCLK) 000: 1/2 LCD COM cycle 001: 1/4 LCD COM cycle 010: 1/8 LCD COM cycle 011: 1/16 LCD COM cycle 100: 1/32 LCD COM cycle 101: 1/64 LCD COM cycle 110: 1/128 LCD COM cycle 111: Fast charge mode invalid Note: This register is for internal resistance mode; configurations are invalid for other modes. This register can only be updated when EN=0.
20	COMSEG_DRV_TYPE	RW	1'b0	LCD waveform type selection 0: TypeA 1: TypeB This register can only be updated when EN=0.
19:16	Reserved	-	-	Reserved
15:12	CONTRAST[3:0]	RW	4'b0	LCD contrast adjustment The larger the CONTRAST value, the smaller the amplitude of the LCD waveform. 0x0: Maximum LCD waveform amplitude and maximum contrast; 0xF: Minimum LCD waveform amplitude and minimum contrast; Note: Only valid when the bias voltage source is selected as internal resistor divider.
11	Reserved	-	-	Reserved
10:9	BSEL	RW	2'b0	Bias voltage mode selection 00: Internal resistor drive mode 01: External resistor drive mode 10: External capacitor drive mode 11: Reserved (function follows internal resistor drive mode)
8:6	DUTY	RW	3'b011	LCD duty cycle selection 000: Static 001: 1/2 duty cycle 010: 1/3 duty cycle 011: 1/4 duty cycle 100: Reserved 101: 1/6 duty cycle 110: Reserved 111: 1/8 duty cycle
5:4	BIAS	RW	2'b0	00: 1/4 bias 01: 1/2 bias 10: 1/3 bias 11: Reserved (function follows 1/4 bias)
3	Reserved	-	-	Reserved
2:1	LCDCLK	RW	2'b01	LCD scanning frequency selection 00: 64 Hz 01: 128 Hz 10: 256 Hz 11: 512 Hz Note: LCD frame frequency = LCD scan frequency × Duty This register can only be updated when EN=0.
0	EN	RW	1'b0	LCD enable control 1: Enabled 0: Disabled When EN is enabled, registers such as COMSEG_DRV_TYPE, BIAS, BSEL, and DUTY are write-protected.

17.6.2. LCD Control Status Register (LCD_CSR)

Offset address: 0x04

Reset value: 0x0000_0080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res.	ENS	UDD	SOF	BKF
												R	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UDR	UDDIE	SOFIE	BKFIE	Res.	MODE	SWCAP_DRV_NUM[2:0]		BLINKE	BLINKCNT[5:0]						
RS	RW	RW	R		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19	ENS	R	0	LCD enable status. This bit is set to 1 and cleared by hardware. It indicates the LCD controller status. 0: LCD controller disabled 1: LCD controller enabled Note: When the EN bit in the LCD_CR register changes from 0 to 1, the ENS bit is immediately set to 1. When disabled, this bit reflects the actual state of the LCD, so it becomes 0 at the end of the last displayed frame.
18	UDD	R	0	Update display done This bit is set to 1 by hardware and cleared by writing 1 to UDDC in the LCD_INTCLR register. The priority of setting the bit is higher than clearing the bit. 0: No event 1: Update display request completed. If the UDDIE bit in the LCD_CSR register is set to 1, a UDD interrupt will be generated. Note: If the device is in stop mode (PCLK not provided), even if UDDIE = 1, UDD will not generate an interrupt. If the display is not enabled, a UDD interrupt will never occur.
17	SOF	R	0	This bit is set to 1 by hardware at the start of a new frame, while updating display data. This bit is cleared by writing 1 to the SOFC bit in the LCD_INTCLR register. Bit clearing has higher priority than bit setting. 0: No event 1: Start of frame event occurred. If the SOFIE bit is set to 1, an LCD start-of-frame interrupt is generated.
16	BKF	R	0	LCD blink data completion interrupt flag 0: No LCD blink data completion interrupt 1: LCD blink data completion interrupt generated
15	UDR	RS	0	Update display request Each time the software modifies LCD_RAMx, the UDR bit must be set to 1 to transfer the updated data to the second-level buffer (LCD_DISPLAYx). The UDR bit remains set to 1 until the update is complete, during which LCD_RAMx is write-protected. 0: No effect 1: Update display request When the display is enabled, updates are performed only for active storage units (depending on the duty cycle configuration). Note: When this bit is already 1, writing 0 or 1 has no effect. This bit can only be cleared by hardware. This bit can only be cleared when LCD_CR.EN=1. When EN is disabled, LCD_DISPLAY updates are completed in 3~4 slow clocks; when EN is enabled, LCD_DISPLAY updates are completed at the start of a frame, and UDR is cleared when the update is complete.
14	UDDIE	RW	0	Update display completion interrupt enable This bit is set to 1 and cleared by the software. 0: Disable LCD update display completion interrupt 1: Enable LCD update display completion interrupt
13	SOFIE	RW	0	Frame start interrupt enable This bit is set and cleared by software. 0: Disable frame start interrupt

				1: Enable frame start interrupt
12	BKFIE	RW	0	Display completion interrupt enable This bit is set and cleared by software. 0: Disable flicker data completion interrupt 1: Enable flicker data completion interrupt
11	Reserved	-	-	Reserved
10	MODE	RW	0	LCD RAM display mode delection 0: Mode 0 1: Mode 1
9:7	SWCAP_DRV_NUM	RW	3'b001	In external capacitor mode, the number of times the external capacitor is driven within one LCDCLK. More drive times result in stronger load. 000: 4 001: 8 010: 16 011: 32 100: 64 Others: Reserved Note: This register is for external capacitor mode. Configuring this register is invalid in other modes. This register can only be updated when EN=0.
6	BLINKEN	RW	0	LCD flicker enable 0: Disabled 1: Enabled
5:0	BLINKCNT	RW	0	Splash screen frequency and LCD interruption interval setting LCD blink frequency = LCD frame frequency / (BlinkCnt+1); LCD interrupt interval = (BlinkCnt+1)*(1/LCD frame frequency). Different COM ports need to be displayed under different duty cycles. This BLINKCNT represents the number of times all COM ports are displayed under a duty cycle. For example, with a 1/8 duty cycle and BLINKCNT=6'd31, the data on all 8 COM ports will be refreshed 32 times before triggering a BKF interrupt. When updating this register with EN=0, the update frequency should be less than or equal to 5 times the slowest clock (LCD_CLK/PCLK), meaning the interval between two updates must be at least 5 slow clock cycles. This register can also be updated with EN=1, but the update frequency should be less than the frame frequency. The new value will only take effect in the next frame, and the blink frame counter will start counting from zero. Note 1: When the LCD is not enabled, after the first update of this register, the second update will be blocked within 5 LSI/LSE clock cycles. Note 2: After the LCD is enabled, the second update will be blocked during the current frame after the first update of this register. The updated value will only take effect in the new frame.

17.6.3. LCD interrupt clear register (LCD_INTCLR)

Offset address: 0x08

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDDC	SOFC	BKFC
													W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	UDDC	W	0	UDDF interrupt flag clear. Write 1 to clear, write 0 has no effect.

17	SOFC	W	0	SOF interrupt flag clear. Write 1 to clear, write 0 has no effect.
16	BKFC	W	0	BKF interrupt flag clear. Write 1 to clear, write 0 has no effect.
15:0	Reserved	-	-	Reserved

17.6.4. LCD output configuration register 0 (LCD_POEN0)

Offset address: 0x0C

Reset value: 0xFFFF_FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31	S30	S29	S28	S27	S26	S25	S24	S23	S22	S21	S20	S19	S18	S17	S16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	Sx	RW	0xFFFFFFFF	SEG31~0 output control bit 0: SEG31~0 output enabled 1: SEG31~0 output disabled, other functions can be used

17.6.5. LCD output configuration register 1 (LCD_POEN1)

Offset address: 0x10

Reset value: 0x0000_0FFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	C3	C2	C1	C0	S39C7	S38C6	S37C5	S36C4	S35	S34	S33	S32
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:8	Cx	RW	0xF	COMx output control bit (COM3~COM0) 0: COM output enabled 1: COM output disabled
7:4	SxCy	RW	0xF	SEGx/COMy output control bit 0: SEG39~36/COM7~4 output enable 1: SEG39~36/COM7~4 output disable SEG/COM pin function selection is determined by LCD_CR.DUTY
3:0	Sx	RW	0xF	SEG35~32 output control bits 0: SEG output enable 1: SEG output disable

17.6.6. LCD_RAM 0 to 7

Offset address: 0x14-0x30

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	Dx	RW	0	LCD dot output, displaying reference LCD display mode 0 The corresponding SEG/COM intersection does not light up; 1 The corresponding SEG/COM intersection lights up;

17.6.7. LCD_RAM8~15

Offset address: 0x34~0x50

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	D7	D6	D5	D4	D3	D2	D1	D0
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	Dx	RW	0	LCD dot output, displaying reference LCD display mode 0 The corresponding SEG/COM intersection does not light up; 1 The corresponding SEG/COM intersection lights up;

18. Advanced Timer (TIM1)

18.1. TIM1 introduction

The advanced-control timer (TIM1) contains a 16-bit auto-reload counter driven by a programmable prescaler.

It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM, complementary PWM with dead-time insertion).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.

The advanced-control timer (TIM1) and general-purpose timer (TIMy) are completely independent of each other and do not share any resources. They can operate synchronously together.

18.2. TIM1 main features

- 16-bit up, down, and up/down auto-reload counter
- 16-bit programmable prescaler with any division factor between 1 and 65535 for the counter clock frequency
- Up to 4 independent channels
 - Input capture
 - Output compare
 - PWM generation (edge-aligned or center-aligned mode)
 - One-pulse mode output
 - Retriggerable single-pulse mode output
- Complementary outputs with programmable dead-time
- Synchronization circuit for controlling the timer with external signals and enabling interconnection of multiple timers
- Repetition counter to update the timer registers only after a given number of cycles of the counter.
- The break input can place the timer's output signals into a user-selectable safe configuration
- Interrupt/DMA request generation on the following events:
 - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
 - Trigger event
 - Input capture
 - Output compare
 - Break input
- Support incremental (quadrature) encoders and Hall sensor circuits for positioning.
- Trigger input as an external clock or cycle-by-cycle current management

18.3. TIM1 functional description

18.3.1. Functional block diagram

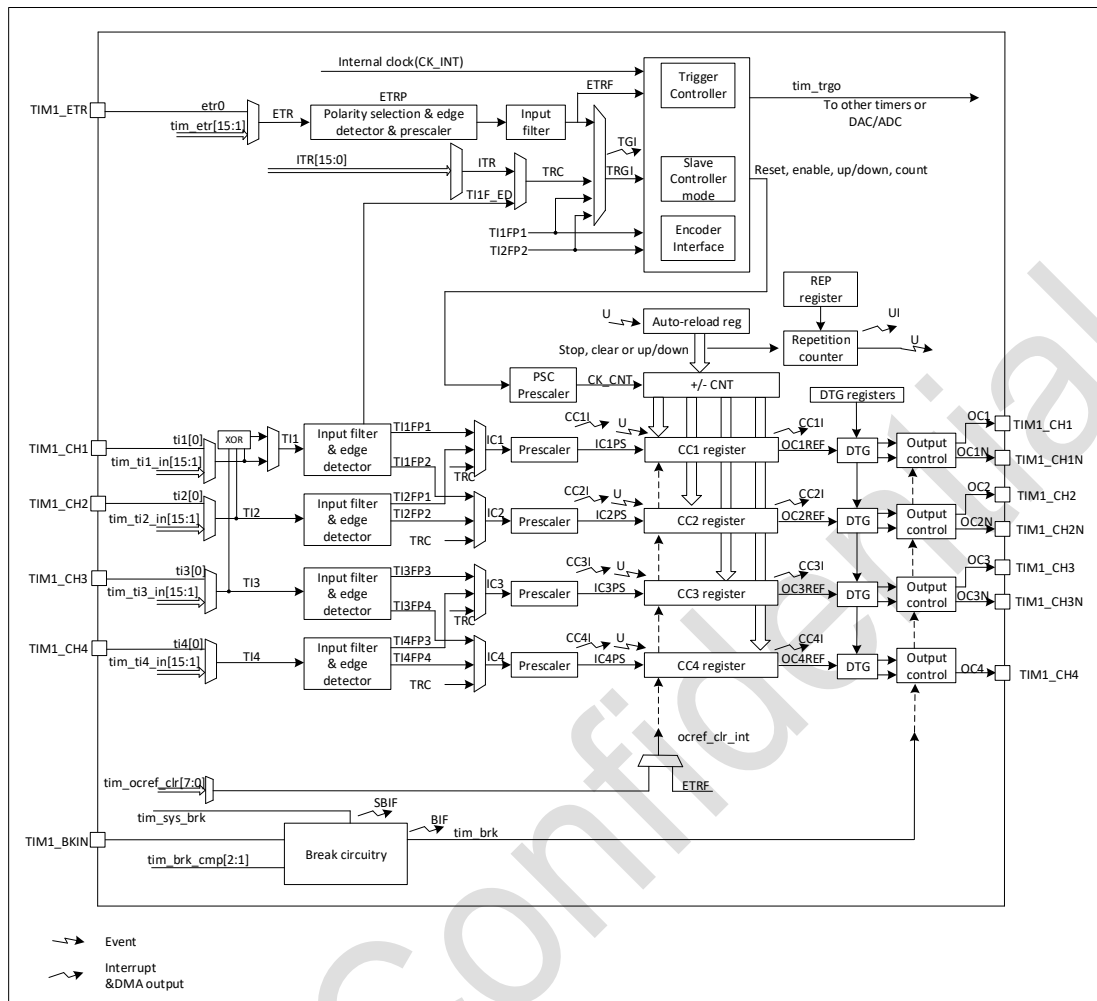


Figure 18-1 Advanced-control timer block diagram

18.3.2. Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The counter clock can be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIM1_CNT)
- Prescaler register (TIM1_PSC)
- Auto-reload register (TIM1_ARR)
- Repetition counter register (TIM1_RCR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM1_CR1 register. An update event is sent when the counter reaches the overflow value (or under-flow value in down-counting mode) and the UDIS bit in the TIM1_CR1 register is 0. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in the TIM1_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIM1_CR1 register.

18.3.2.1. Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIM1_PSC register). It can be changed

on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

The following two figures provide examples of changing counter parameters while the prescaler is running.

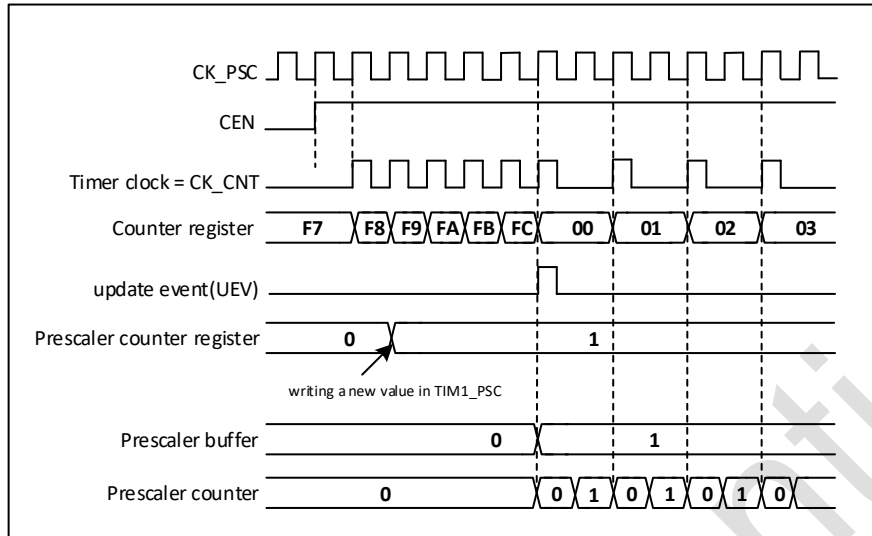


Figure 18-2 Counter timing diagram when prescaler division changes from 1 to 2

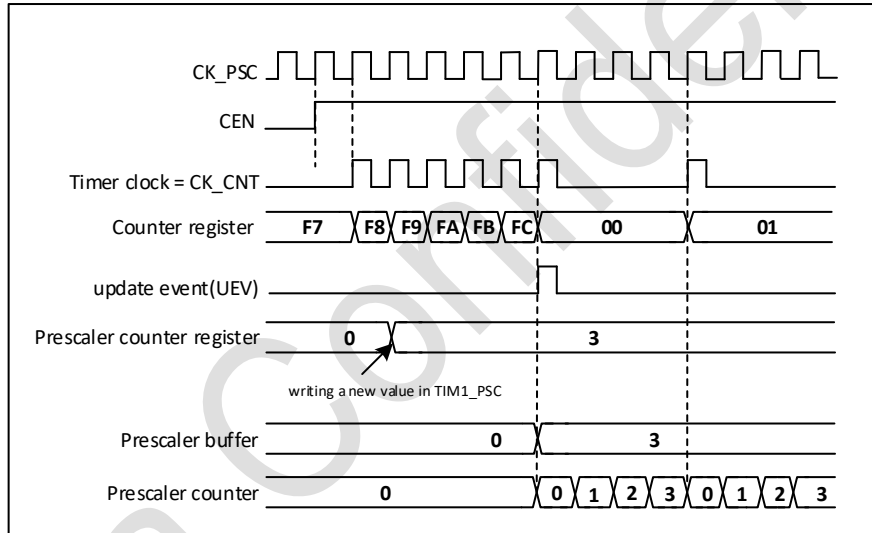


Figure 18-3 Counter timing diagram when prescaler division changes from 1 to 4

18.3.3. Counting modes

18.3.3.1. Upcounting mode

In up-counting mode, the counter counts from 0 to the auto-reload value (TIM1_ARR register), then restarts from 0 and generates a counter overflow event.

If the repetition counter is used, an update event (UEV) is generated when the number of up-counting repetitions reaches the value set in the repetition counter register plus one ((TIM1_RCR) + 1). Else the update event is generated at each counter overflow.

Setting the UG bit in the TIM1_EGR register (via software or using the slave mode controller) can also generate an update event.

Setting the UDIS bit in the TIM1_CR1 register disables update events (UEV); this also prevents updating shadow registers when writing new values to the preload registers. Then no update event occurs until the UDIS bit has been written to 0. Even so, the counter is still cleared to '0' when an update event should occur, and the prescaler count is also cleared (though the prescaler value remains unchanged). Additionally, if the URS bit in the TIM1_CR1 register is set (update request selection), setting the UG bit generates an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all registers are updated and the update flag (UIF bit in TIM1_SR register) is set to 1 (depending on the URS bit).

- The repetition counter is reloaded with the content of the TIM1_RCR register.

- The auto-reload shadow register is updated with the preload value (TIM1_ARR).
 - The buffer of the prescaler is reloaded with the preload value (content of the TIM1_PSC register).
- The following figure shows some examples of counter operation at different clock frequencies when TIM1_ARR=0x36.

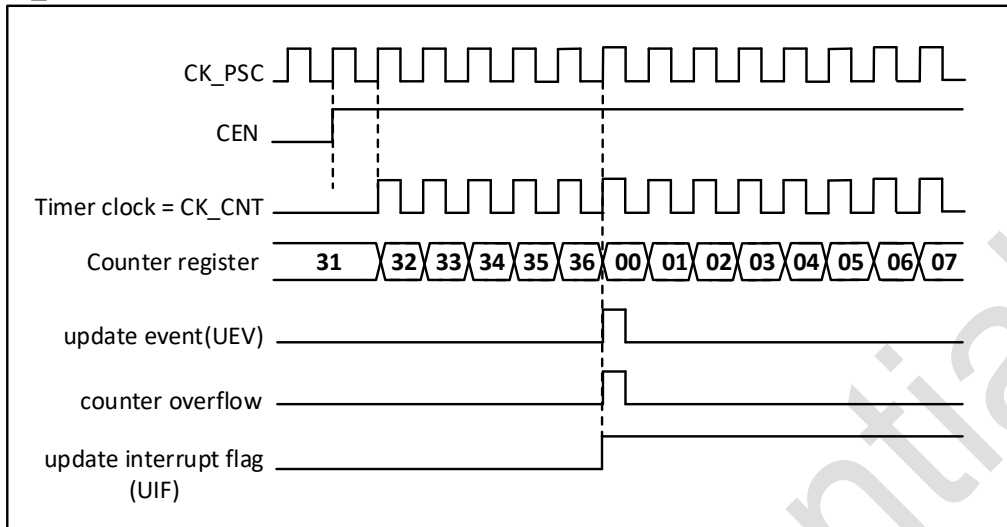


Figure 18-4 Counter timing diagram, internal clock divided by 1

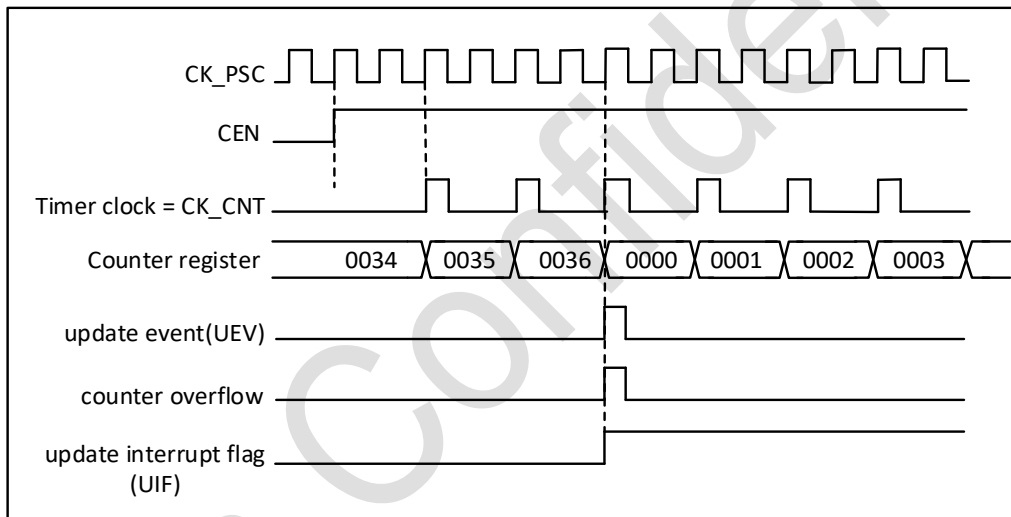


Figure 18-5 Counter timing diagram, internal clock divided by 2

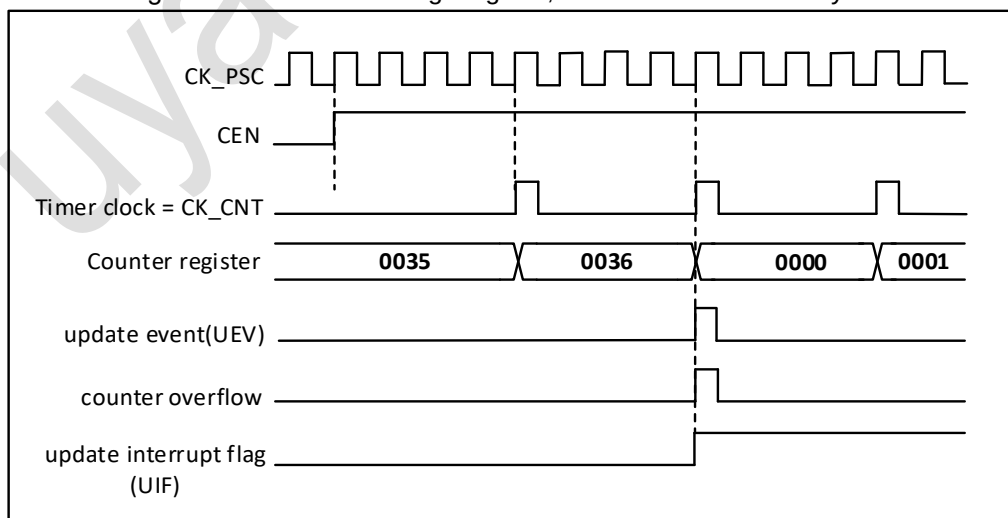


Figure 18-6 Counter timing diagram, internal clock divided by 4

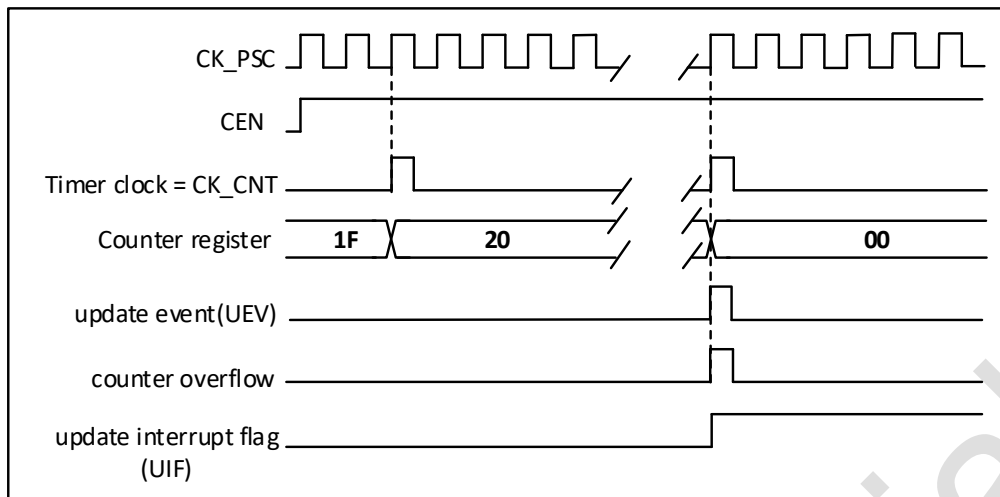


Figure 18-7 Counter timing diagram, internal clock divided by N

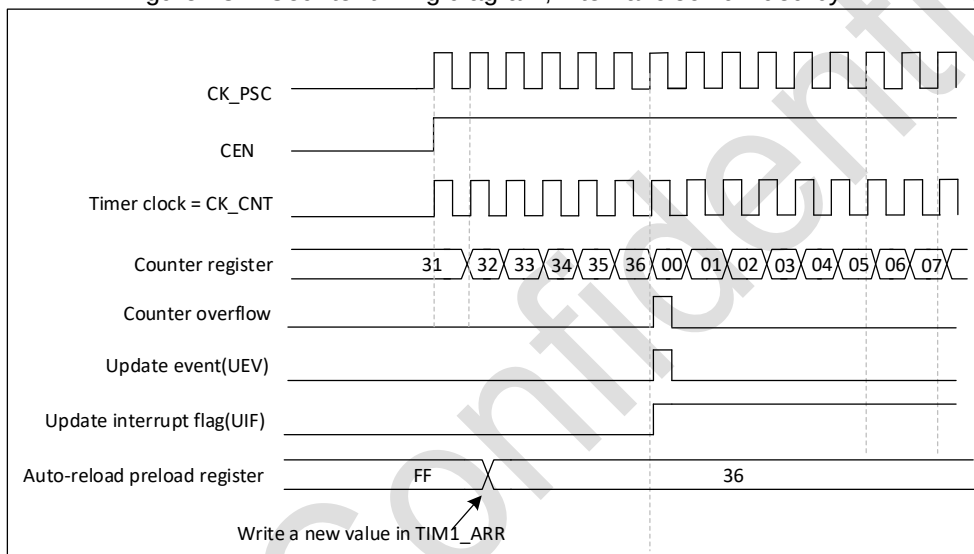


Figure 18-8 Counter timing diagram, update event when ARPE=0 (TIM1_ARR not preloaded)

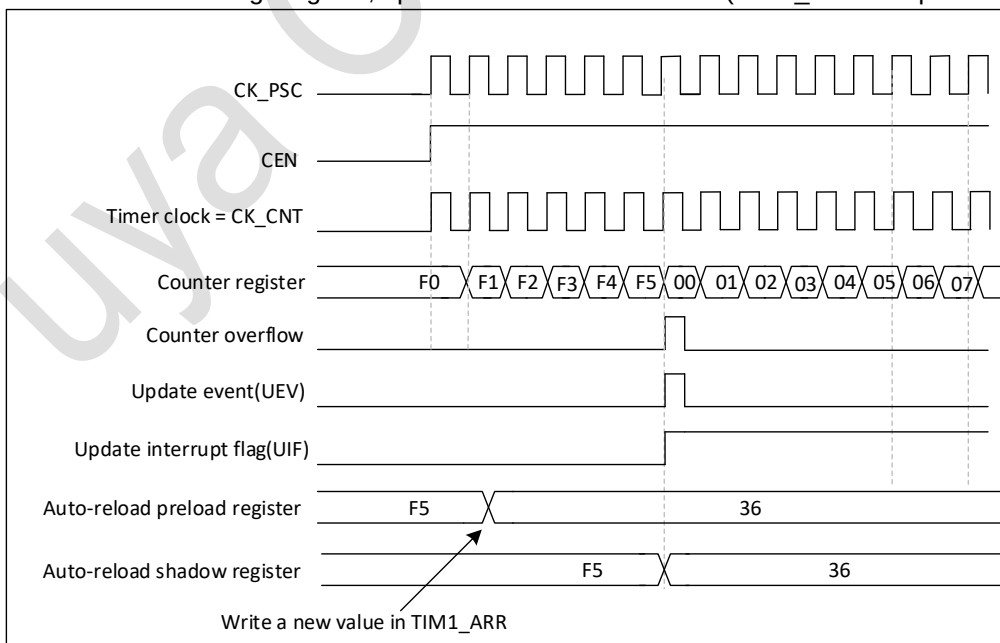


Figure 18-9 Counter timing diagram, update event when ARPE=1 (TIM1_ARR preloaded)

18.3.3.2. Downcounting mode

In downcounting mode, the counter counts down from the auto-reload value to 0, then restarts counting down from the auto-reload value and generates a downcount overflow event.

If the repetition counter is used, an update event (UEV) is generated when the number of downcount repetitions reaches the value set in the repetition counter register plus one (TIM1_RCR +1). Otherwise, an update event is generated only when the counter underflows.

Setting the UG bit in the TIM1_EGR register (via software or using the slave mode controller) can also generate an update event.

Setting the UDIS bit in the TIM1_CR1 register can disable update events (UEV); this prevents updating shadow registers when writing new values to the preload registers. Then no update event occurs until the UDIS bit has been written to 0. Even so, the counter is still cleared to '0' when an update event should occur, and the prescaler count is also cleared (though the prescaler value remains unchanged).

Additionally, if the URS bit in the TIM1_CR1 register is set (update request selection), setting the UG bit generates an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all registers are updated and the update flag (UIF bit in TIM1_SR register) is set to 1 (depending on the URS bit).

- The repetition counter is reset to the content of the TIM1_RCR register.
- The prescaler buffer is loaded with the preload value (content of the TIM1_PSC register).
- The current auto-reload register is updated with the preload value (content of the TIM1_ARR register).

Note that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

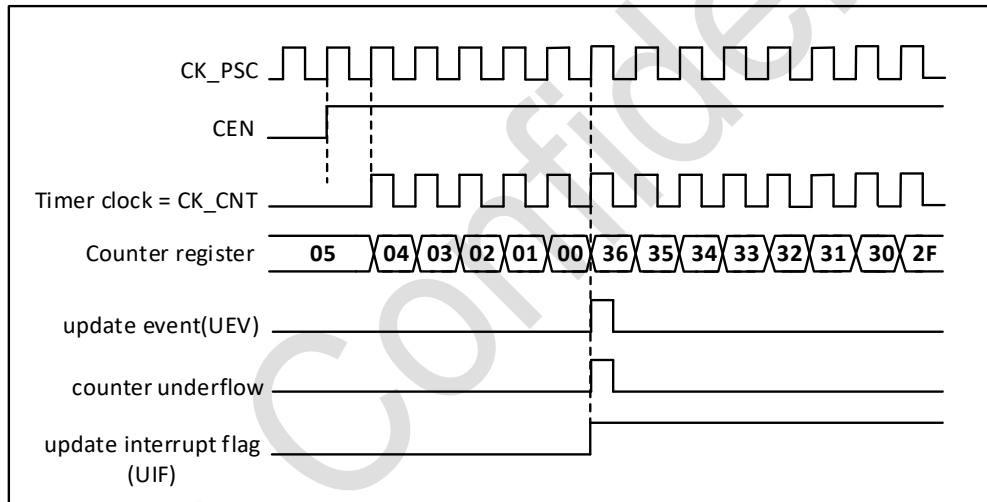


Figure 18-10 Counter timing diagram, internal clock divided by 1

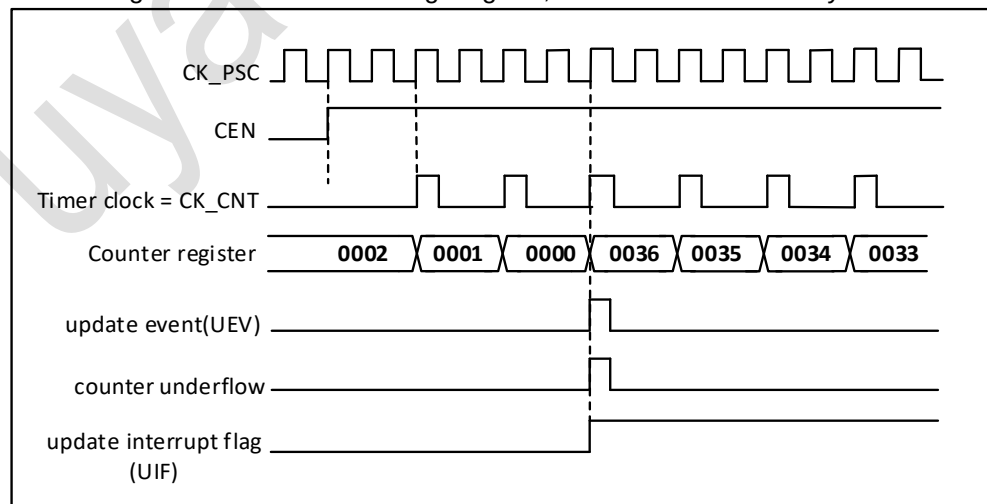


Figure 18-11 Counter timing diagram, internal clock divided by 2

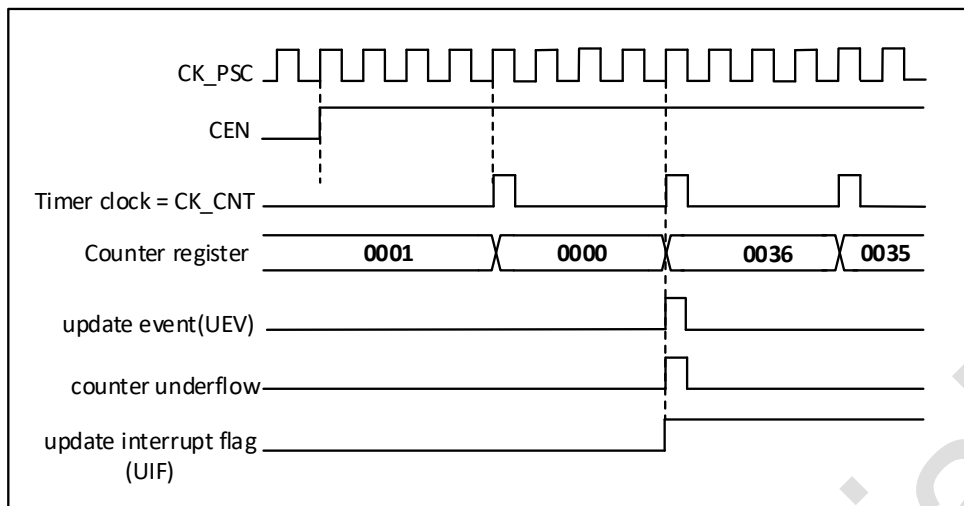


Figure 18-12 Counter timing diagram, internal clock divided by 4

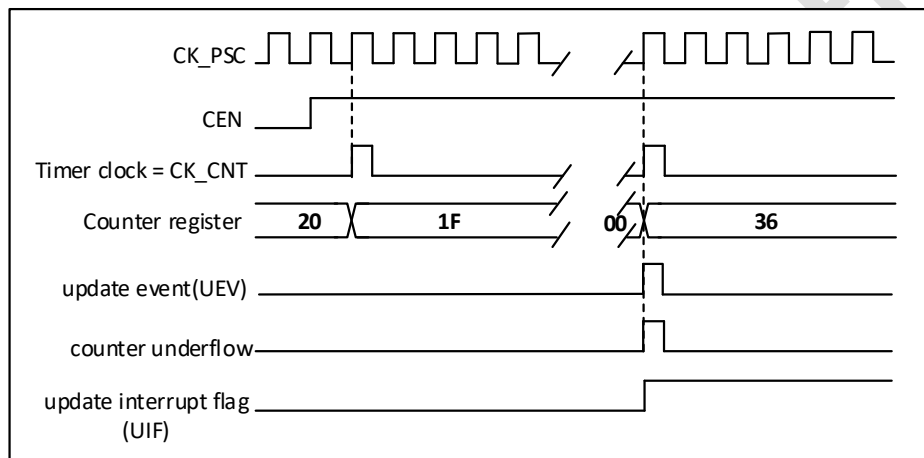


Figure 18-13 Counter timing diagram, internal clock divided by N

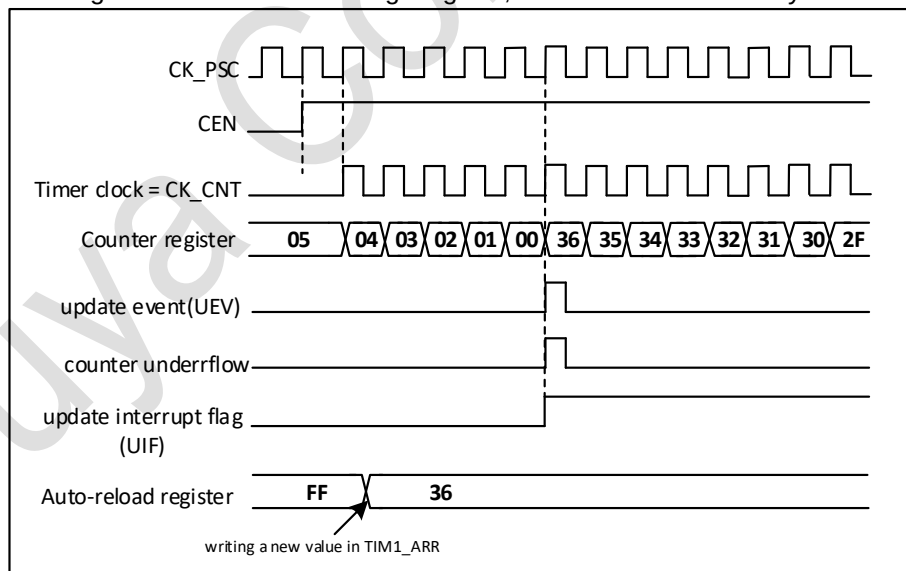


Figure 18-14 Counter timing diagram, update event when repeat counter is not used

18.3.3.3. Center-aligned mode

In center-aligned mode, the counter counts from 0 up to the auto-reload value (TIM1_ARR register) -1, generating a counter overflow event; then counts down from the auto-reload value to 1 and generates a counter underflow event; then starts counting again from 0.

Center-aligned mode is active when the CMS bits in TIM1_CR1 register are not equal to '00'. When a channel is configured in output mode, its output compare interrupt flag will be set to 1 in the follow-

ing modes: counter counting down (center-aligned mode 1, CMS = "01"), counter counting up (center-aligned mode 2, CMS = "10"), and counter counting up/down (center-aligned mode 3, CMS = "11").

In this mode, the DIR direction bit in the TIM1_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter.

An update event can be generated on each counter overflow and each counter underflow; it can also be generated by setting the UG bit in the TIM1_EGR register (via software or using the slave mode controller). In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV event can be disabled by software by setting the UDIS bit in the TIM1_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter will continue counting up or down based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIM1_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag (thus no interrupt and DMA requests are sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all registers are updated and the update flag (UIF bit in TIM1_SR register) is set to 1 (depending on the URS bit).

- The repetition counter is reset to the content of the TIM1_RCR register.
- The prescaler buffer is loaded with the preload value (content of the TIM1_PSC register).
- The current auto-reload register is updated with the preload value (content of the TIM1_ARR register).

Note: If an update occurs due to counter overflow, the auto-reload will be updated before the counter is reloaded, so the next cycle will have the expected value (the counter is loaded with the new value).

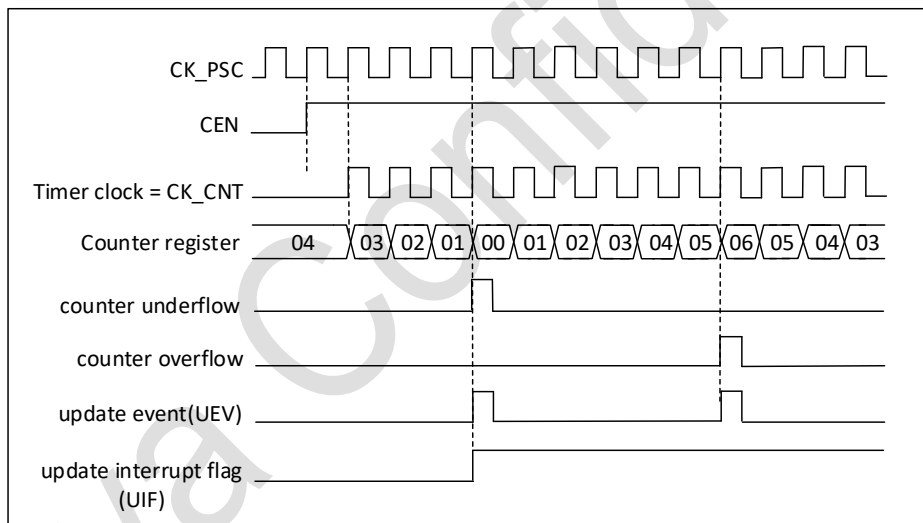


Figure 18-15 Counter timing diagram, internal clock divided by 1, TIM1_ARR = 0x6

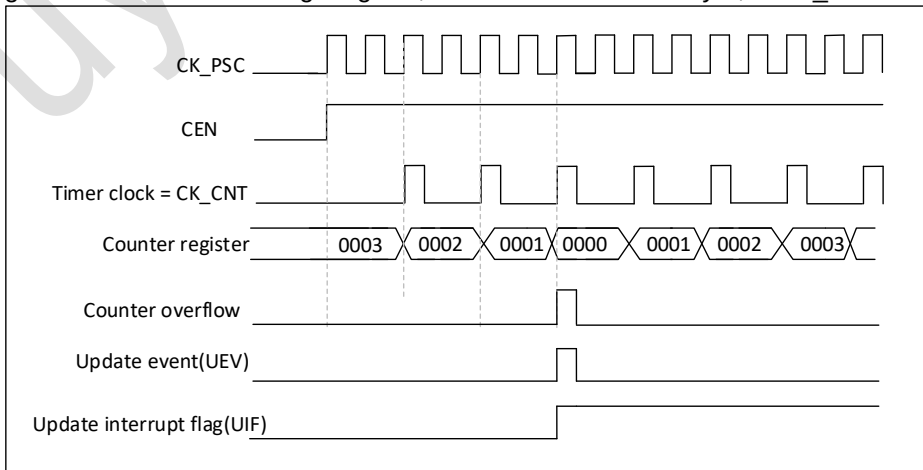


Figure 18-16 Counter timing diagram, internal clock divided by 2

Figure 18-17 Counter timing diagram, internal clock divided by 4, TIM1_ARR=0x36

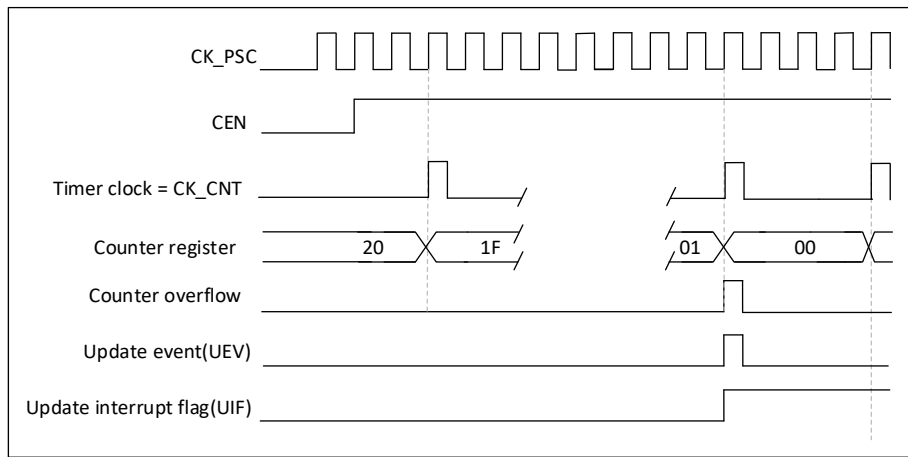


Figure 18-18 Counter timing diagram, internal clock divided by N

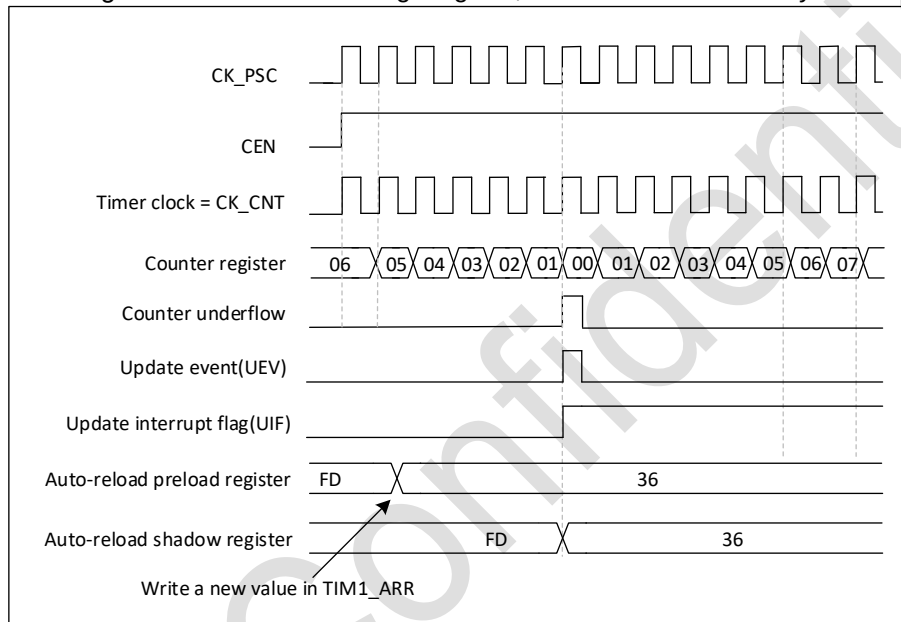


Figure 18-19 Counter timing diagram, update event with ARPE=1 (counter underflow)

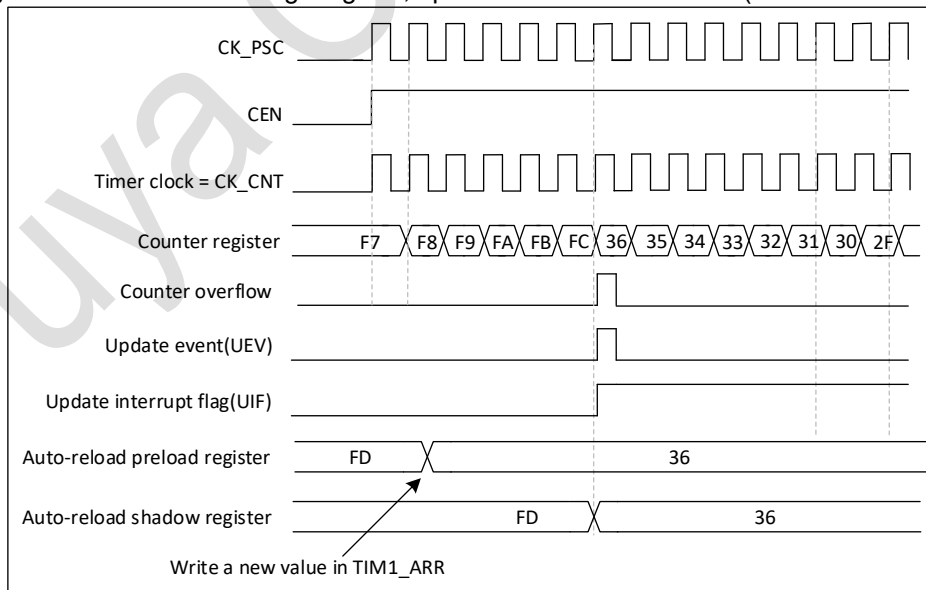


Figure 18-20 Counter timing diagram, update event when ARPE=1 (counter overflow)

18.3.4. Repetition counter

The time-base unit describes how update events (UEV) are generated during counter overflow or underflow. It is actually generated only when the repetition counter has reached zero. This is useful for generating PWM signals.

This means that on every $N+1$ overflow or underflow (where N is the value in the TIM1_RCR repetition counter register), data is transferred from the preload register to the shadow register (TIM1_ARR auto-reload register, TIM1_PSC prescaler register, and the capture/compare register TIM1_CCRx in compare mode).

The repetition counter is decremented:

- On each counter overflow in upcounting mode
- On each counter underflow in downcounting mode
- On each overflow and underflow in center-aligned mode. Although this limits the maximum PWM cycle period to 128, it allows updating the duty cycle twice per PWM period. In center-aligned mode, because the waveform is symmetric, if the compare register is refreshed only once per PWM period, the maximum resolution is $2 \times T_{ck}$.

The repetition counter is auto-loaded, and the repetition rate is defined by the value in the TIM1_RCR register. When an update event is generated by software (by setting the UG bit in TIM1_EGR) or by the hardware slave mode controller, the update event occurs immediately regardless of the repetition counter value, and the content of the TIM1_RCR register is reloaded into the repetition counter.

In center-aligned mode, if the RCR value is odd, the update event occurs on overflow or underflow, depending on when the RCR register is written and when the counter is started: if RCR is written before starting the counter, UEV occurs on overflow. If RCR is written after starting the counter, UEV occurs on underflow.

For example, when $RCR = 3$, the update event is generated on the 4th overflow or underflow event (depending on when the RCR value is written).

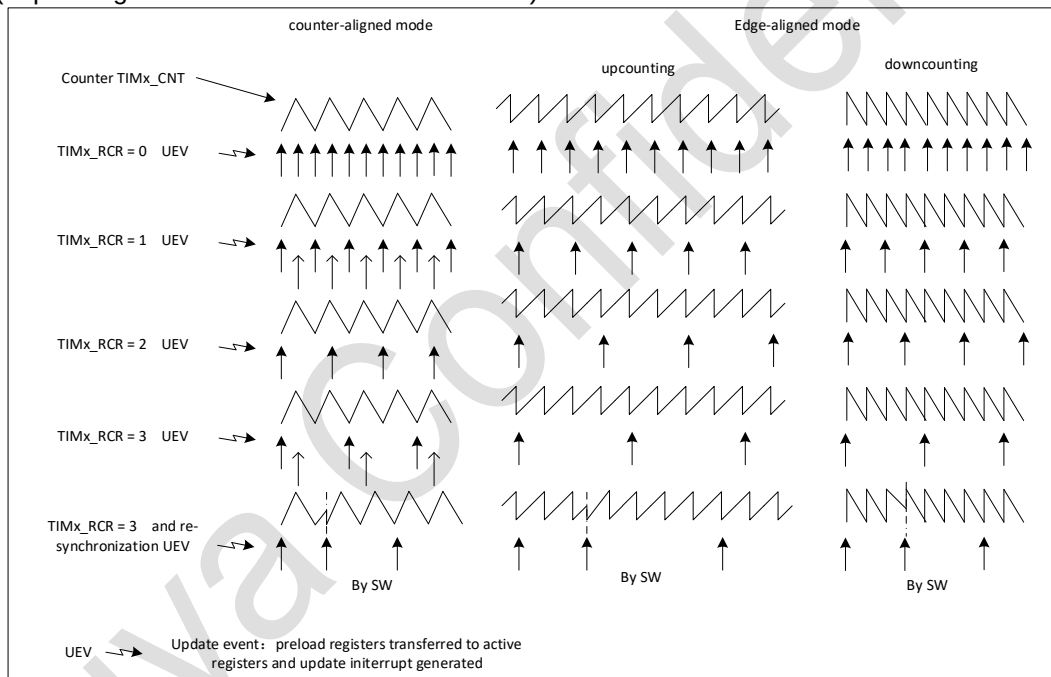


Figure 18-21 Example of update frequency under different modes and TIM1_RCR register settings

18.3.5. External trigger input

The timer features an external trigger input ETR. It can be used as:

- External clock (external clock mode 2)
- Trigger signal for slave mode
- PWM reset input for cycle-by-cycle current regulation

The diagram below illustrates the conditioning process of the ETR input. The input polarity is defined by the ETP bit in the TIM1_SMCR register. The trigger signal can be prescaled by a division ratio programmed via the ETPS[1:0] bit field, then digitally filtered through the ETF[3:0] bit field.

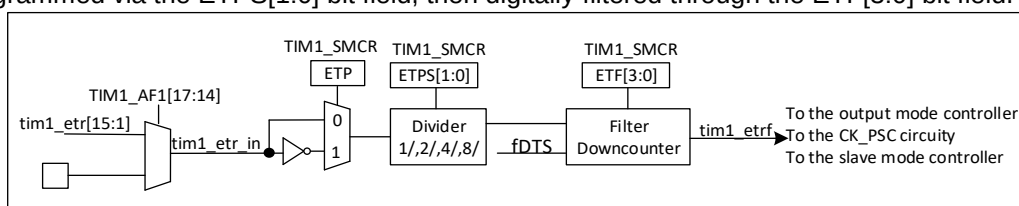


Figure 18-22 External Trigger Input

The ETR input comes from multiple sources: the input pin (default configuration), comparator output, and analog watchdog. The selection is done with the ETRSEL[3:0] bitfield. Refer to the definition of the TIM1_AF1.ETRSEL register for specific selections.

18.3.6. Clock sources

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode1: external input pin
- External clock mode2: external trigger input ETR
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer. For example, Timer 1 can be configured as a prescaler for another Timer 3.

18.3.6.1. Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS=000), the CEN, DIR (TIM1_CR1 register), and UG bits (TIM1_EGR register) are the actual control bits and can only be modified by software (except UG, which remains auto-cleared). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

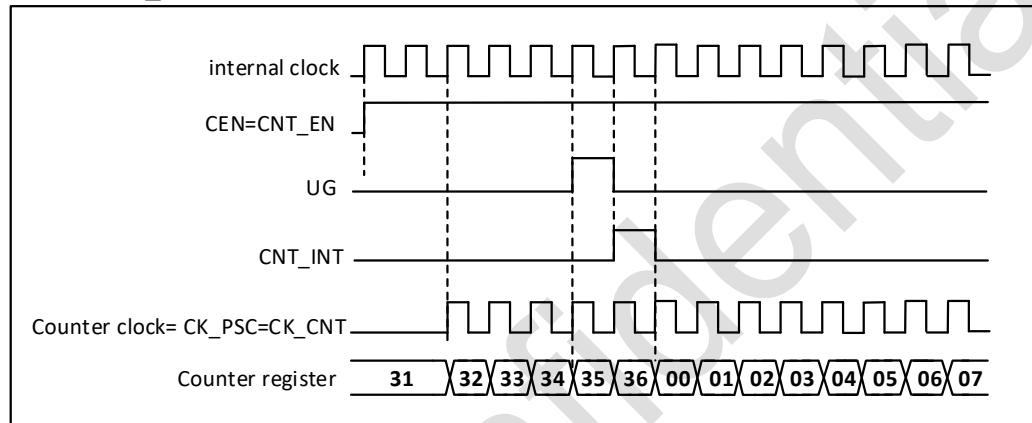


Figure 18-23 Control Circuit in Normal Mode, Internal Clock Divided by 1

18.3.6.2. External clock source mode 1

This mode is selected when SMS=111 in the TIM1_SMCR register. The counter can count at each rising or falling edge on a selected input.

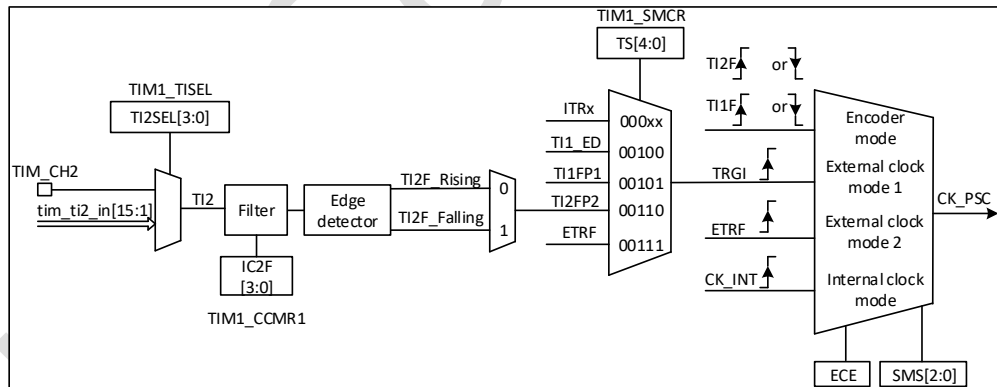


Figure 18-24 Example of TI2 External Clock Connection

For example, to configure the upcounter to count in response to a rising edge on the TI2 input, use the following procedure:

1. Select the correct TI2x source (internal or external) by configuring the TI2SEL[3:0] bits in the TIM1_TISEL register.
2. Configure Channel 2 to detect the rising edge of the TI2 input by writing CC2S="01" in the TIM1_CCMR1 register.
3. Configure the input filter bandwidth by writing the IC2F[3:0] bits in the TIM1_CCMR1 register (if no filter is needed, keep IC2F=0000).
4. Select the rising edge polarity by writing CC2P=0 and CC2NP=0 in the TIM1_CCER register.
5. Set the timer to operate in external clock mode 1 by writing SMS=111 in the TIM1_SMCR register.
6. Select TI2 as the trigger input source by writing TS=00110 in the TIM1_SMCR register.
7. Enable the counter by writing CEN=1 in the TIM1_CR1 register.

Note: The capture prescaler is not used for triggering, so the user does not need to configure it. When a rising edge occurs on TI2, the counter increments once and the TIF flag is set to 1. The delay between the rising edge of TI2 and the actual counter clock is caused by the resynchronization circuit of the TI2 input.

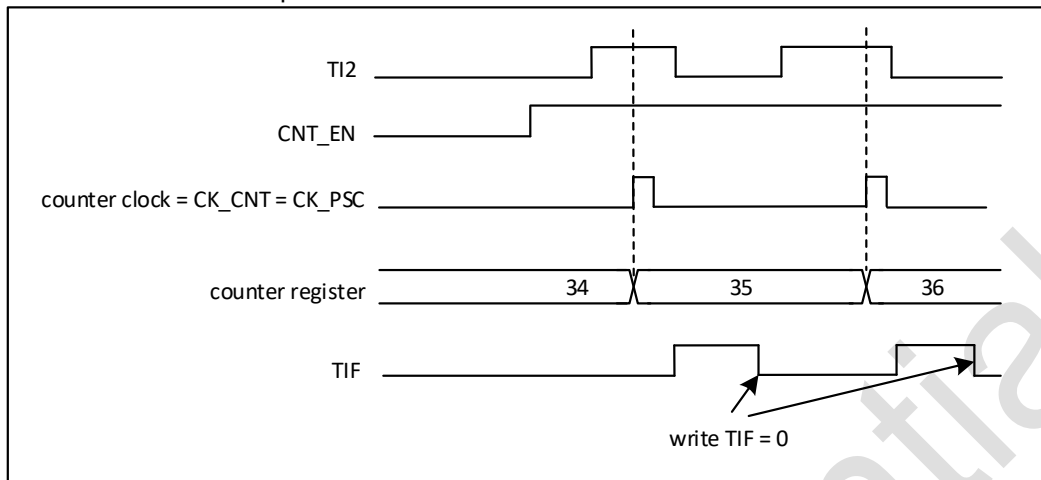


Figure 18-25 Timing of External Clock Mode 1 Control Circuit

18.3.6.3. External clock source mode 2

This mode is selected by writing ECE=1 in the TIM1_SMCR register. The counter can count at each rising or falling edge on the external trigger input ETR.

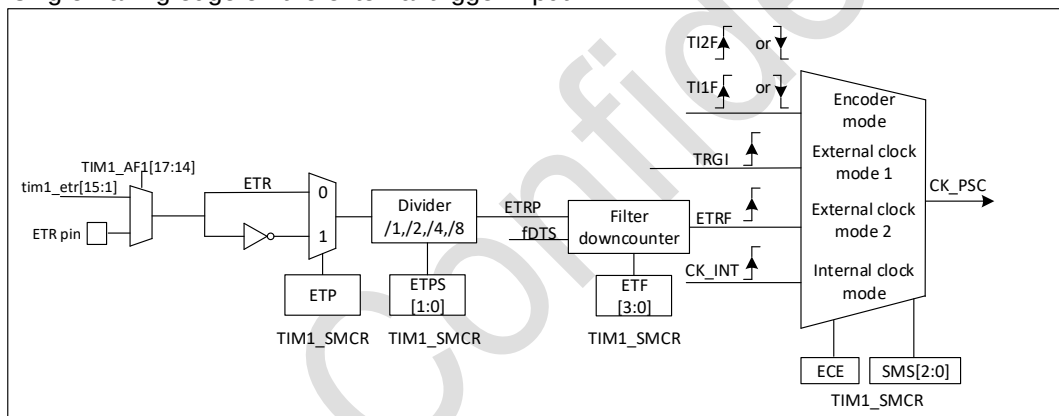


Figure 18-26 External Trigger Input Module

For example, to configure the upcounter to count each 2 rising edges on ETR, use the following procedure:

1. Since no filter is needed in this example, write ETF[3:0]=0000 in the TIM1_SMCR register.
2. Set the prescaler by writing ETPS[1:0]=01 in the TIM1_SMCR register.
3. Select rising edge detection on the ETR pin by writing ETP=0 in the TIM1_SMCR register.
4. Enable external clock mode 2 by writing ECE=1 in the TIM1_SMCR register.
5. Enable the counter by writing CEN=1 in the TIM1_CR1 register.

The counter counts once each 2 ETR rising edges.

The delay between the rising edge on ETR and the actual clock of the counter is due to the resynchronization circuit on the ETRP signal.

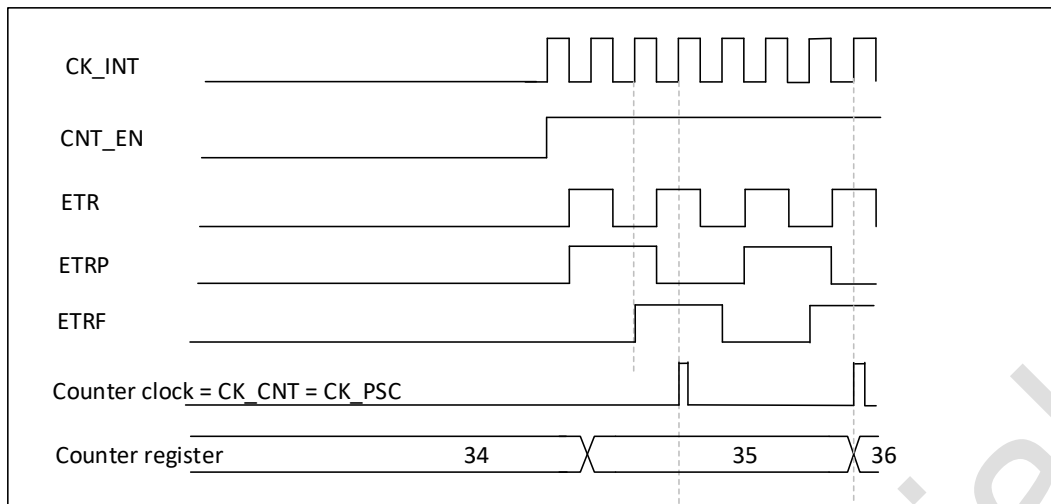


Figure 18-27 Timing diagram of external clock mode 2 control circuit

18.3.7. Capture/Compare channels

Each capture/compare channel is built around a capture/compare register (including a shadow register), a capture input section (digital filter, multiplexer, and prescaler), and an output section (comparator and output control).

The input stage samples the corresponding TIx input to generate a filtered signal $TIxF$. Then, an edge detector with polarity selection generates a signal ($TIxFPx$) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register ($ICxPS$).

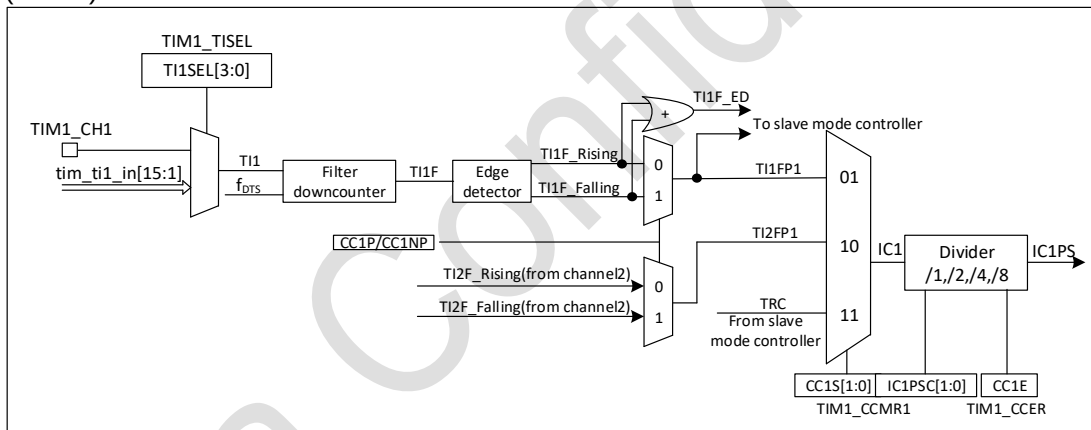


Figure 18-28 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: $OCxRef$ (active high). The polarity acts at the end of the chain.

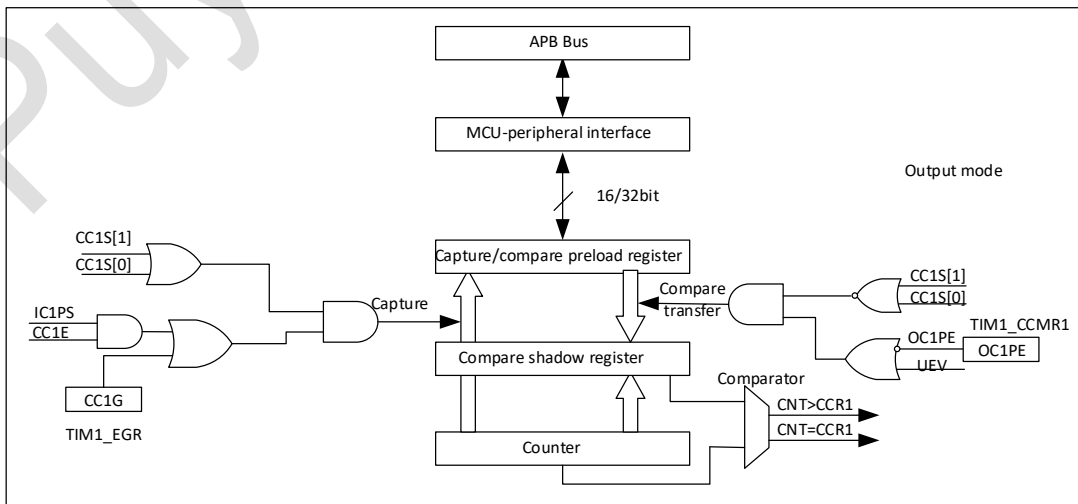


Figure 18-29 Main circuit of capture/compare channel 1

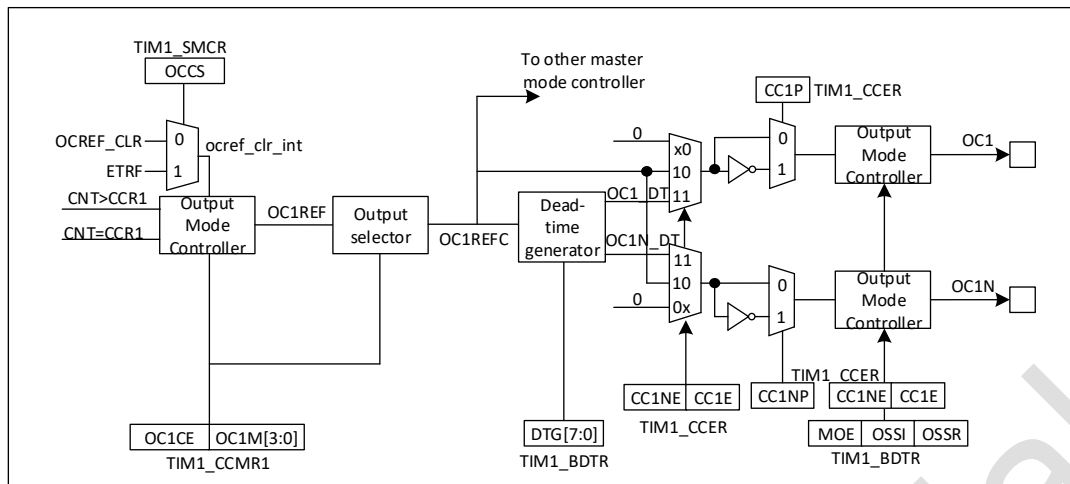


Figure 18-30 Output stage of capture/compare channels (channels 1 to 3)

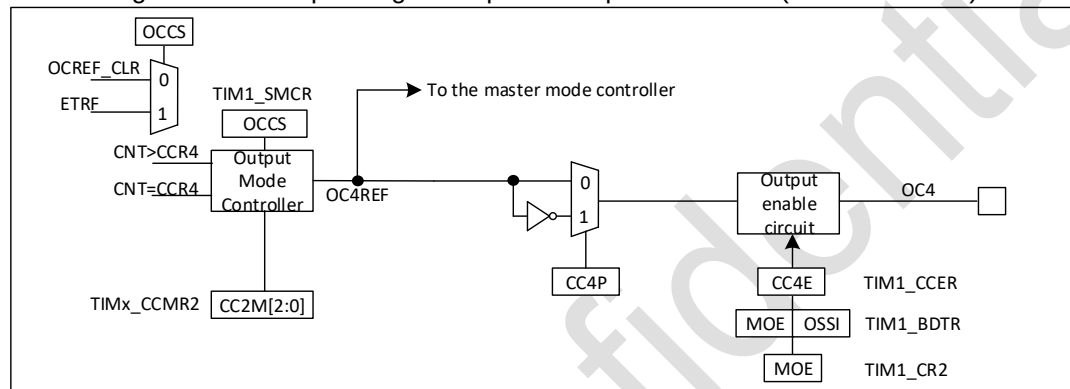


Figure 18-31 Output stage of capture/compare channel (channel 4)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, the capture occurs on the shadow register, which is then copied to the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

18.3.8. Input capture mode:

In Input capture mode, the capture/compare registers are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture event occurs, the corresponding CCxIF flag (TIM1_SR register) is set to 1, and if interrupts and DMA operations are enabled, an interrupt or DMA request will be generated. If the CCxIF flag is already high when a capture event occurs, the repeat capture flag CCxOF (TIM1_SR register) is set to 1. Writing CCxIF=0 clears CCxIF, or reading the captured data stored in the TIM1_CCRx register also clears CCxIF. CCxOF is cleared when you write it to '0'.

The following example shows how to capture the counter value in TIM1_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the correct TI1x source (internal or external) using the TI1SEL[3:0] bits in the TIM1_TISEL register.
- Select the active input: TIM1_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIM1_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIM1_CCR1 register becomes read-only.
- Configure the input filter for the desired bandwidth based on the characteristics of the input signal (i.e., the input filter control bits are ICxF in the TIM1_CCMRx register when the input is TIx). Assuming the input signal jitters for up to 5 internal clock cycles, we must configure the filter bandwidth to be longer than 5 clock cycles; after detecting 8 consecutive samples with a new level (at fDTS frequency), a transition edge on TI1 can be confirmed, then write IC1F=0011 to the TIM1_CCMR1 register.
- Select the active transition edge (in this case, the rising edge) on TI1 by writing CC1P and CC1NP bits to 0 in the TIM1_CCER register.
- Program the input prescaler. In this example, we want captures to occur at every valid level transition, so the prescaler should be disabled (write IC1PSC=00 in the TIM1_CCMR1 register).

- Set CC1E=1 in the TIM1_CCER register to allow capturing the counter value into the capture register.
- If needed, enable the corresponding interrupt request by setting the CC1IE bit in the TIM1_DIER register, and enable DMA requests by setting the CC1DE bit in the TIM1_DIER register.

When an input capture occurs:

- The TIM1_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). When at least two consecutive captures occur but CC1IF is not cleared, CC1OF is also set to 1.
- An interrupt is generated depending on the CC1IE bit.
- If the CC1DE bit is set, a DMA request will be generated.

To handle capture overflow, it is recommended to read the data before reading the capture overflow flag to avoid losing overflow information that may occur between reading the flag and reading the data.

Note: Setting the corresponding CCxG bit in the TIM1_EGR register can generate an input capture interrupt and/or DMA request via software.

18.3.9. PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- Select one of the TIxFP signals as the trigger input signal, while the slave mode controller is configured in reset mode.
- For example, when measuring the length (TIM1_CCR1 register) and duty cycle (TIM1_CCR2 register) of a PWM signal input to TI1, the specific steps are as follows (depending on the CK_INT frequency and prescaler value).
- Use the TI1SEL[3:0] bits in the TIM1_TISEL register to select the correct TI1x source (internal or external).
- Select the active input for TIM1_CCR1: write the CC1S bits to 01 in the TIM1_CCMR1 register (TI1 selected).
- Select the active polarity of TI1FP1 (used to capture data into TIM1_CCR1 and clear the counter) by setting CC1P and CC1NP to 0 (rising edge active).
- Select the active input for TIM1_CCR2: write the CC2S bits to 10 in the TIM1_CCMR1 register (TI1 selected).
- Select the active polarity of TI1FP2 (capturing data into TIM1_CCR2): set CC2P=1 and CC2NP=0 (falling edge active).
- Select the active trigger input signal: set TS=00101 in the TIM1_SMCR register (selecting TI1FP1).
- Configure the slave mode controller for reset mode: set SMS=0100 in TIM1_SMCR.
- Enable capture: set CC1E=1 and CC2E=1 in the TIM1_CCER register.

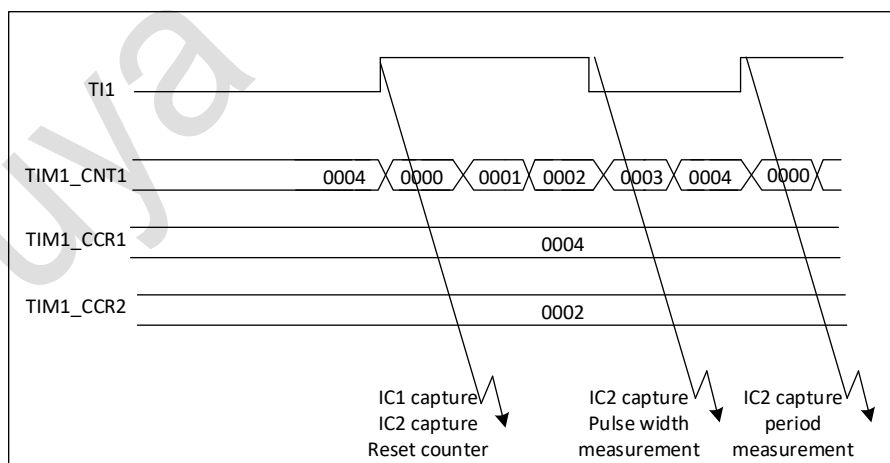


Figure 18-32 PWM input mode timing

18.3.10. Forced Output Mode

In output mode (CCxS=00 in the TIM1_CCMRx register), the output compare signal (OCxREF and corresponding OCx/OCxN) can be forced to an active or inactive state directly by software, independent of the comparison result between the output compare register and the counter. Set the corresponding OCxM=0101 in the TIM1_CCMRx register to force the output compare signal (OCxREF/OCx) to an active state. Thus OCxREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level. The OCxREF signal can be forced low by writing the OCxM bits to 0100 in the TIM1_CCMRx register.

Anyway, the comparison between the TIM1_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

18.3.11. Output compare mode

This feature is used to control an output waveform or indicate that a given time period has elapsed. When a match is found between the capture/compare register and the counter, the output compare function:

- Output the value defined by the output compare mode (OCxM bit in the TIM1_CCMRx register) and output polarity (CCxP bit in the TIM1_CCER register) to the corresponding pin. During compare match, the output pin can maintain its level (OCxM=0000), be set to the active level (OCxM=0001), be set to the inactive level (OCxM=0010), or toggle (OCxM=0011).
- Set the flag in the interrupt status register (CCxIF bit in the TIM1_SR register).
- If the corresponding interrupt enable is set (CCxIE bit in the TIM1_DIER register), an interrupt is generated.
- If the corresponding DMA enable bit is set (CCxDE bit in the TIM1_DIER register, with the CCDS bit in the TIM1_CR2 register selecting the DMA request function), a DMA request is generated.

The TIM1_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM1_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing accuracy can reach one counting cycle of the counter. The output compare mode can also be used to output a single pulse (in one-pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the corresponding data to the TIM1_ARR and TIM1_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:
 - Set OCxM=0011 to toggle the OCx output pin when the counter matches CCRx,
 - Disable the preload register by setting OCxPE = 0
 - Write CCxP = 0 to select active high polarity
 - Write CCxE = 1 to enable the output
5. Enable the counter by setting the CEN bit in the TIM1_CR1 register.

The TIM1_CCRx register can be updated via software at any time to control the output waveform, provided the preload register is not enabled (OCxPE='0'; otherwise, the shadow register of TIM1_CCRx can only be updated upon the next update event). An example is given in the figure below.

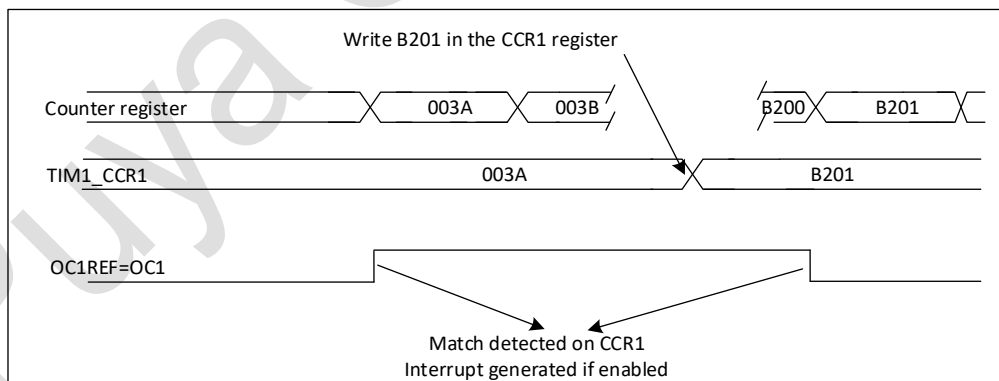


Figure 18-33 Output Compare Mode, Toggling OC1

18.3.12. PWM mode

Pulse Width Modulation (PWM) mode allows generating a signal with a frequency determined by the TIM1_ARR register and a duty cycle determined by the TIM1_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '0110' (PWM mode 1) or '0111' (PWM mode 2) in the OCxM bits in the TIM1_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM1_CCMRx register, and finally, the ARPE bit in the TIM1_CR1 register must be set to 1 to enable the auto-reload preload register (in up-counting or center-aligned mode).

Since the preload register can only be transferred to the shadow register upon an update event, all registers must be initialized by setting the UG bit in the TIM1_EGR register before the counter starts counting.

The polarity of OCx can be set via software using the CCxP bit in the TIM1_CCER register, which can be configured as active high or active low. The output of OCx is controlled by the combination of the CCxE, CCxNE, MOE, OSSI, and OSSR bits (in the TIM1_CCER and TIM1_BDTR registers). Refer to the description of the TIM1_CCER register for details.

In PWM mode (Mode 1 or Mode 2), TIM1_CNT and TIM1_CCRx are continuously compared to determine whether $TIM1\_CCR_x \leq TIM1\_CNT$ or $TIM1\_CNT \leq TIM1\_CCR_x$ (depending on the counter's counting direction).

Depending on the state of the CMS bit in the TIM1_CR1 register, the timer can generate edge-aligned PWM signals or center-aligned PWM signals.

18.3.12.1. PWM edge-aligned mode

Upcounting configuration

Up-counting is performed when the DIR bit in the TIM1_CR1 register is low. In the following example, we consider PWM mode 1. When $TIM1\_CNT < TIM1\_CCR_x$, the PWM reference signal OCxREF is high; otherwise, it is low. If the compare value in TIM1_CCRx is greater than the auto-reload value (in TIM1_ARR) then OCxREF is held at '1'. If the compare value is 0 then OCxREF is held at '0'. The figure below shows an example of edge-aligned PWM waveforms when $TIM1\_ARR=8$.

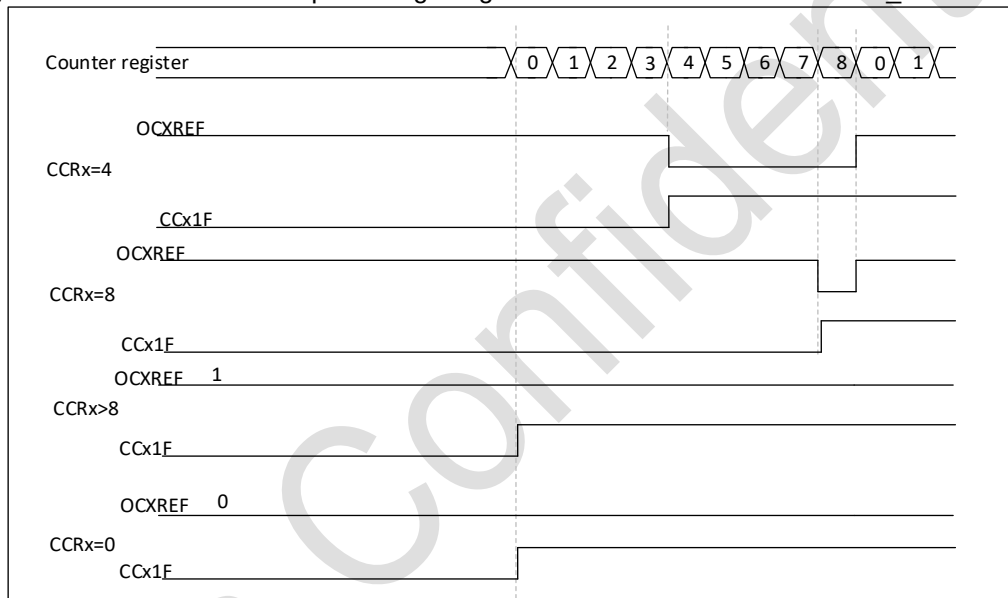


Figure 18-34 Edge-aligned PWM output waveform (ARR=8)

Downcounting configuration

Down-counting is performed when the DIR bit in the TIM1_CR1 register is high.

In PWM mode 1, when $TIM1\_CNT > TIM1\_CCR_x$, the reference signal OCxREF is low; otherwise, it is high. If the compare value in TIM1_CCRx is greater than the auto-reload value in TIM1_ARR, OCxREF remains at '1'. 0% PWM is not possible in this mode.

18.3.12.2. PWM center-aligned mode

When the CMS bit in the TIM1_CR1 register is not '00', it is in center-aligned mode (all other configurations have the same effect on the OCxREF/OCx signal). Depending on the CMS bit settings, the compare flag can be set to 1 during up-counting, down-counting, or both up and down-counting. The count direction bit (DIR) in the TIM1_CR1 register is updated by hardware and cannot be modified by software.

The figure below shows some examples of center-aligned PWM waveforms.

- TIM1_ARR = 8
- PWM mode is the PWM mode 1
- When CMS=01 in the TIM1_CR1 register, in center-aligned mode, the compare flag is set to 1 during down-counting.

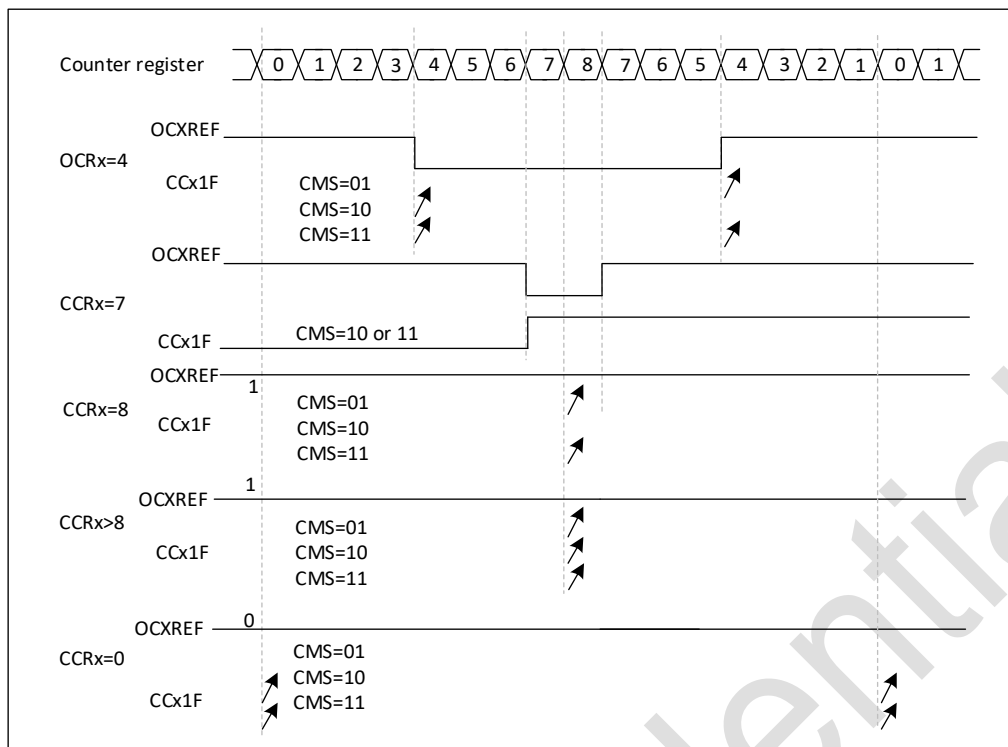


Figure 18-35 Center-aligned mode waveform (ARR=8)

Hints on using center-aligned mode:

- When entering center-aligned mode, the current up/down counting configuration is used; this means the counting direction (up or down) depends on the current value of the DIR bit in the TIM1_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- It is not recommended to rewrite the counter while running in center-aligned mode, as this may produce unpredictable results. In particular:
 - If the value written to the counter is greater than the auto-reload value (TIM1_CNT > TIM1_ARR), the direction is not updated. For example, if the counter is up-counting, it will continue to up-count.
 - If 0 or the TIM1_ARR value is written to the counter, the counting direction is updated, but no update event (UEV) is generated.
- The safest way to use center-aligned mode is to generate a software update (set the UG bit in the TIM1_EGR register) before starting the counter and avoid modifying the counter value during counting.

18.3.13. Complementary outputs and dead-time insertion

The advanced-control timers (TIM1) can output two complementary signals and manage the switching-off and the switching-on instants of the outputs. This period is typically called the dead-time, and users must adjust the dead-time based on the connected output devices and their characteristics (level transition delay, power switch delay, etc.).

By configuring the CCxP and CCxNP bits in the TIM1_CCER register, the polarity (main output OCx or complementary output OCxN) can be independently selected for each output.

The complementary signals OCx and OCxN are controlled through a combination of the following control bits: the CCxE and CCxNE bits in the TIM1_CCER register, and the MOE, OISx, OISxN, OSS1, and OSSR bits in the TIM1_BDTR and TIM1_CR2 registers. Refer to the table after the TIM1_CCER register description for details on the control bits of the complementary output channels OCx and OCxN with brake functionality. Specifically, when transitioning to the IDLE state (MOE falling to 0), the dead-time remains effective.

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. There is one 8-bit dead-time generator for each channel. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is identical to the reference signal, except that its rising edge is delayed relative to the rising edge of the reference signal.
- The OCxN output signal is inverted from the reference signal, except that its rising edge is delayed relative to the falling edge of the reference signal.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose CCxP=0, CCxNP=0, MOE=1, CCxE=1 and CCxNE=1 in these examples)

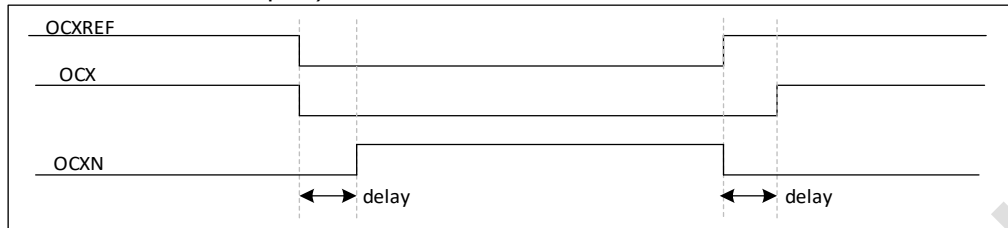


Figure 18-36 Complementary output with dead-time insertion

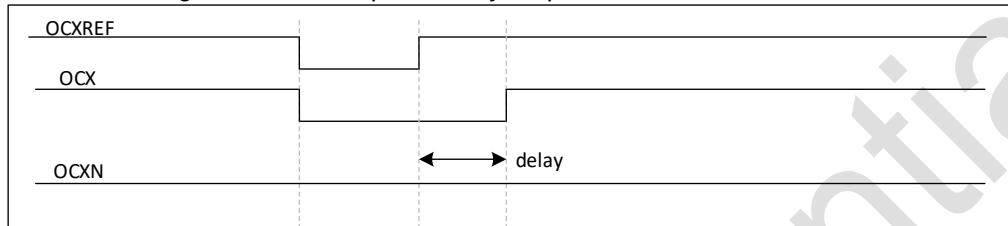


Figure 18-37 Dead-time waveform where the delay time is greater than the negative pulse width

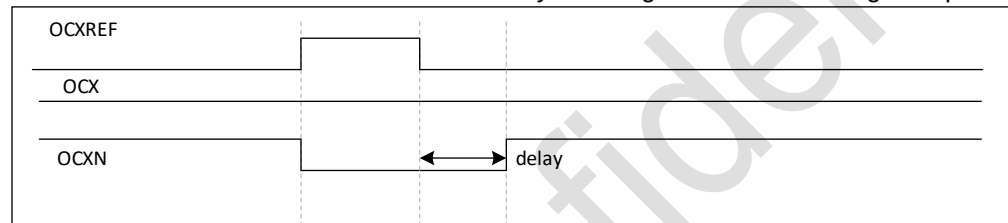


Figure 18-38 Dead-time waveform where the delay time is greater than the positive pulse width
The dead-time delay for each channel is the same and is programmed by the DTG bits in the TIM1_BDTR register.

18.3.13.1. Redirect OCxREF to OCx or OCxN

In output mode (forced mode, output compare mode, or PWM mode), OCxREF can be redirected to the output of OCx or OCxN by configuring the CCxE and CCxNE bits in the TIM1_CCER register. This feature allows sending a special waveform (e.g., PWM or static active level) on one output while keeping the complementary output at an inactive level when the complementary output is inactive. Alternatively, both outputs can be kept at inactive levels simultaneously, or both can be kept at active levels with complementary outputs and dead-time.

Note: When only OCxN is enabled (CCxE=0, CCxNE=1), the outputs are not complementary, and OCxN immediately goes high when OCxREF is active. For example, if CCxNP=0 then OCxN=OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE=CCxNE=1), OCx is active when OCxREF is high, while OCxN is active when OCxREF is low (inverted behavior).

18.3.14. Using the break function

The purpose of the break function is to protect power switches driven by PWM signals generated with the TIM1 timer. The two break inputs are usually connected to fault outputs of power stages and 3-phase inverters. When activated, the break circuitry shuts down the PWM outputs and forces them to a predefined safe state. A number of internal MCU events can also be selected to trigger an output shut-down.

A break channel which gathers both system-level fault (clock failure, parity error,...) and application fault (from input pins and built-in comparator), and can force the outputs to a predefined level (either active or inactive) after a deadtime duration.

The output enable signal and output levels during break are depending on several control bits:

- The MOE bit in the TIM1_BDTR register allows software to enable/disable the output and is reset during a brake event.
- The OSS1 bit in the TIM1_BDTR register defines whether the timer controls the output in an inactive state or releases control to the GPIO controller (typically placing it in a high-impedance state).
- The OISx and OISxN bits in the TIM1_CR2 register set the output to the shutdown level (active or inactive). Regardless of the values of OISx and OISxN, it is not possible to set both OCx and OCxN outputs to an active level simultaneously at a given time.

When exiting from reset, the break circuit is disabled and the MOE bit is low. The brake function is enabled by setting the BKE bit in the TIMx_BDTR register. The polarity of the brake input can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time.

Because the falling edge of MOE can be asynchronous, a resynchronization circuit is implemented between the actual signal (acting on the output) and the synchronous control bit (in the TIM1_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if you write MOE to 1 whereas it was low, you must insert a delay (dummy instruction) before reading it correctly. This is because the write acts on the asynchronous signal whereas the read reflects the synchronous signal.

The brake source can be either the brake input pin or the following internal sources (see the "Peripheral Interconnection" section for details):

- the CPU LOCKUP output
- the PVD output
- a clock failure event generated by the CSS detector
- the output from a comparator
- SRAM byte parity error
- FLASH ECC error

A brake event can also be generated by software via the BG bit in the TIM1_EGR register. Regardless of the value of the BKE enable bit, a brake can be generated via software using BG.

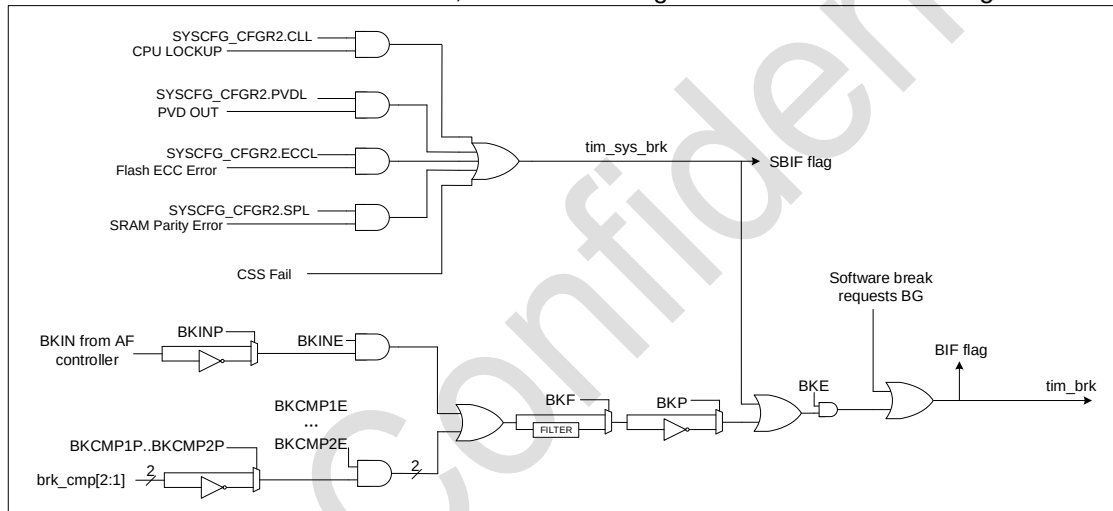


Figure 18-39 Brake circuit diagram

When a break occurs (selected level on the break input):

- The MOE bit is asynchronously cleared, placing the output in an inactive state, idle state, or releasing control to the GPIO (selected by the OSS1 bit). This feature functions even if the MCU oscillator is off.
- Once MOE=0, each output channel is driven at the level set by the OISx bit in the TIM1_CR2 register. If OSS1=0, the timer releases output control (taken over by the GPIO controller); otherwise, the output is enabled to remain high.
- When complementary outputs are used:
 - The output is first placed in an inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
 - If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note: Due to MOE resynchronization, the dead time is slightly longer than usual (approximately 2 timer clock cycles).
 - If OSS1=0, the timer releases the enabled output (taken over by the GPIO controller forcing a high-impedance state); otherwise, the enabled output remains high or immediately goes high if either the CCxE or CCxNE bit is high.
- If the BIE bit in the TIM1_DIER register is set, an interrupt is generated when the brake status flags (SBIF and BIF bits in the TIM1_SR register) are '1'.
- If the AOE bit in the TIM1_BDTR register is set, the MOE bit is automatically set again at the next update event (UEV); for example, this can be used for shaping. Otherwise, MOE remains low until the application sets it to '1' again. In this case, this feature ensures safety by allowing the brake input to be connected to a power-driven alarm output, thermal sensor, or other safety device.

Note: The break inputs are active on level. Therefore, when the brake input is active, the MOE bit cannot be set to 1 (automatically or via software). In the meantime, the status flag BIF can not be cleared.

The brake can be generated by the BRK input, whose active polarity is programmable and enabled by the BKE bit in the TIM1_BDTR register.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). Users can select one of three protection levels via the LOCK bit in the TIM1_BDTR register. The LOCK bits can be written only once after an MCU reset.

The figure below shows an example of behavior of the outputs in response to a break.

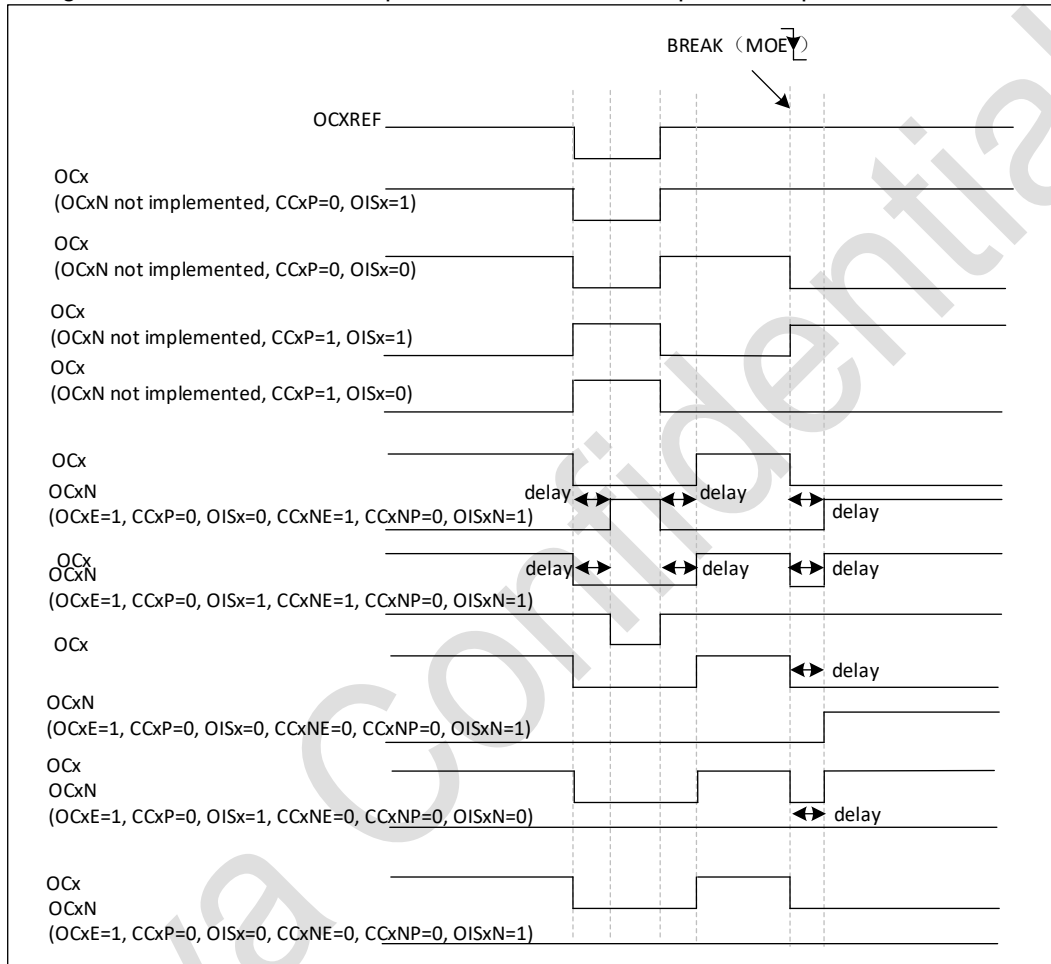


Figure 18-40 Different output behaviors in response to a brake event (OSSI=1)

18.3.15. Clearing the OCxREF signal upon an external event

For a given channel, applying a high level to the ocref_clr_int input (with the OCxCE enable bit in the corresponding TIM1_CCMRx register set to 1) clears the OCxREF signal. The OCxREF signal remains low until the next update event (UEV) occurs. This feature can only be used in output compare mode and PWM mode. It does not work in Forced mode. By configuring the OCCS bit in the TIM1_SMCR register, the ocref_clr_int input can be selected between the OCREF_CLR input and ETRF (filtered ETR).

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, ETR must be configured as follows:

1. The external trigger prescaler must be off: ETPS[1:0]=00 in the TIM1_SMCR register.
2. External clock mode 2 must be disabled: ECE=0 in the TIM1_SMCR register.
3. The external trigger polarity (ETP) and the external trigger filter (ETF) can be configured as needed.

The figure below shows the behavior of the OCxREF signal when the ETRF Input becomes High, for both values of the enable bit OCxCE. In this example, the TIM1 timer is placed in PWM mode.

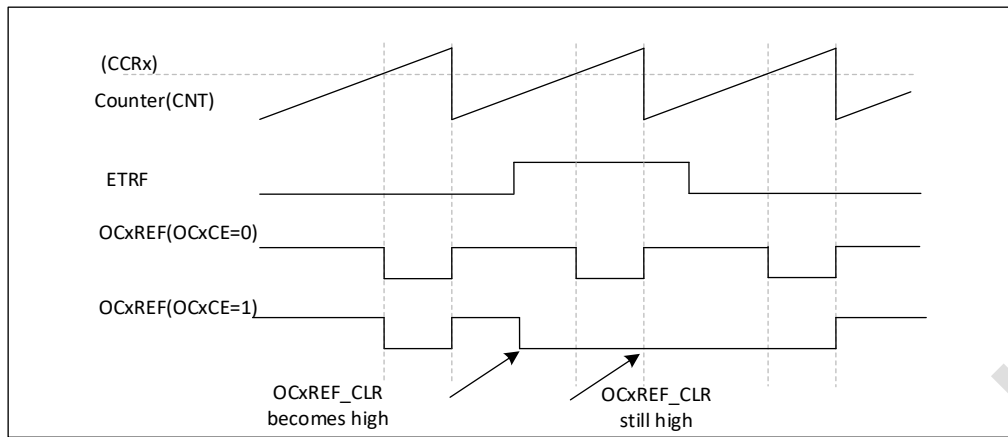


Figure 18-41 Clearing TIM1's OCxREF

Note: If the PWM duty cycle is 100% ($CCR_x > ARR$), the OCxREF will be re-enabled upon the next counter overflow.

18.3.16. 6-step PWM

When complementary outputs are used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. Upon a COM event, these preload bits are transferred to the shadow register bits. Thus, users can preconfigure the settings for the next step and modify the configuration of all channels simultaneously at the same moment. COM can be generated by software by setting the COM bit in the TIM1_EGR register or by hardware on the rising edge of TRGI.

A flag (COMIF bit in the TIM1_SR register) is set when a COM event occurs. If the COMIE bit in the TIM1_DIER register is set, an interrupt is generated; if the COMDE bit in the TIM1_DIER register is set, a DMA request is generated.

The figure below shows the behavior of OCx and OCxN outputs when a COM event occurs.

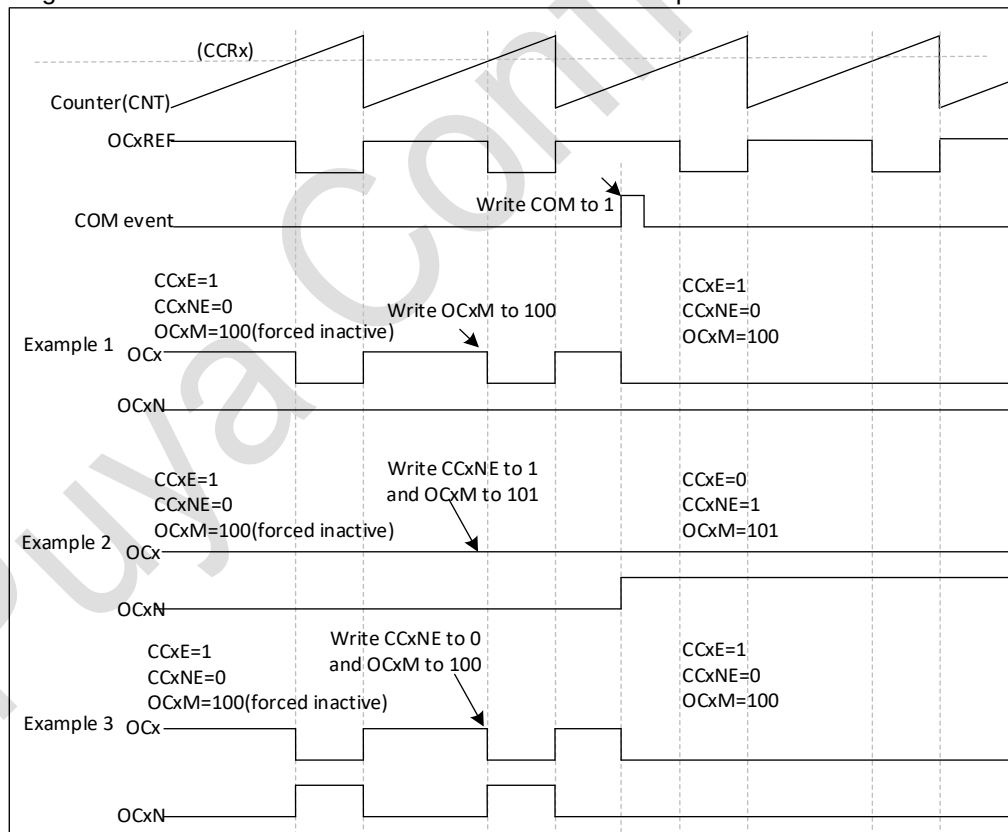


Figure 18-42 Example of COM event generating 6-step PWM (OSSR=1)

18.3.17. One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. Setting the OPM bit in the TIM1_CR1 register selects the One Pulse Mode, which allows the counter to automatically stop when the next update event (UEV) is generated.

A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- Up-counting mode: Counter CNT < CCRx ≤ ARR (specifically, 0 < CCRx)
- Down-counting mode: Counter CNT > CCRx

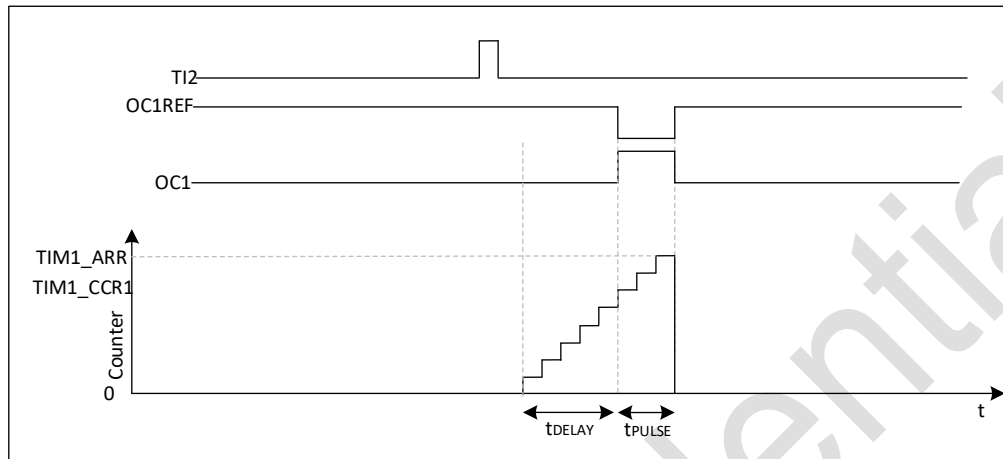


Figure 18-43 Example of One Pulse Mode

For example one may want to generate a positive pulse on OC1 with a length of tPULSE and after a delay of tDELAY as soon as a positive edge is detected on the TI2 input pin.

Using TI2FP2 as the trigger source:

- Select the correct TI2x source using the TI2SEL[3:0] bits in the TIM1_TISEL register.
- Map TI2FP2 on TI2 by writing CC2S=01 in the TIM1_CCMR1 register.
- Set CC2P=0 and CC2NP=0 in the TIM1_CCER register to enable TI2FP2 to detect rising edges.
- Set TS=00110 in the TIM1_SMCR register to use TI2FP2 as the trigger (TRGI) for the slave mode controller.
- Set SMS=110 (Trigger mode) in the TIM1_SMCR register to use TI2FP2 to start the counter.

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The tDELAY is defined by the value written in the TIM1_CCR1 register.
- tPULSE is defined by the difference between the auto-reload value and the compare value (TIM1_ARR – TIM1_CCR1).
- Assuming a waveform transition from 0 to 1 is to be generated upon a compare match, and a transition from 1 to 0 when the counter reaches the preload value; first, set OC1M=0111 in the TIM1_CCMR1 register to enter PWM Mode 2; optionally enable the preload register as needed: set OC1PE=1 in TIM1_CCMR1 and ARPE in the TIM1_CR1 register; then fill in the compare value in the TIM1_CCR1 register and the auto-reload value in the TIM1_ARR register, set the UG bit to generate an update event, and wait for an external trigger event on TI2. CC1P is written to '0' in this example.

In our example, the DIR and CMS bits in the TIM1_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIM1_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0) When OPM=0 in the TIM1_CR1 register, the Repetitive Mode is selected.

18.3.17.1. Particular case: OCx fast enable

In One Pulse Mode, edge detection on the TIx input sets the CEN bit to 1, indicating that the counter is enabled. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations and it limits the minimum delay tDELAY min we can get.

To output the waveform with minimal delay, set the OCxFE bit to 1 in the TIM1_CCMRx register. This forces OCxREF (and OCx) to respond to the excitation signal, regardless of the compare result. Its new level is the same as if a compare match had occurred. OCxFE only takes effect when the channel is configured in PWM1 or PWM2 mode.

18.3.18. Retriggerable one pulse mode (OPM)

This mode allows the counter to start upon a trigger from an excitation signal and generate a pulse with programmable length, differing from the non-retriggerable One Pulse Mode described in the previous section in the following ways:

- The pulse starts immediately upon the trigger (no programmable delay).
- If a new trigger occurs before the pulse generated by the previous trigger is completed, the pulse is extended.

To use the retriggerable one-pulse mode, the timer must be in slave mode with TIM1_SMCR register bits SMS[3:0]="1000" (combined mode + trigger mode), and OCxM[3:0] bits set to "1000" or "1001" (retriggerable OPM mode 1 or 2).

If the timer is configured in Up-counting mode, the corresponding CCRx must be set to 0 (the ARR register sets the pulse length). If the timer is configured in Down-counting mode, CCRx must be above or equal to ARR.

Notes:

1. For compatibility reasons, the OCxM[3:0] and SMS[3:0] bit fields are split into two parts, with the most significant bit not contiguous with the three least significant bits.
2. This mode must not be used with center-aligned PWM mode. It is mandatory to have CMS[1:0] = 00 in TIM1_CR1.

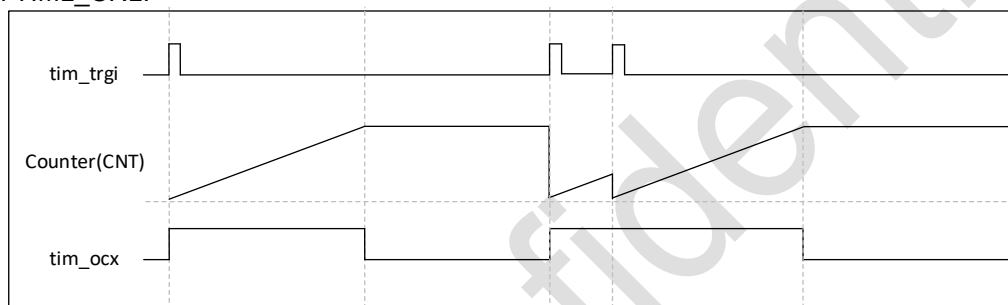


Figure 18-44 Retriggerable One-Pulse Mode

18.3.19. Encoder interface mode

The method to select the encoder interface mode is: if the counter counts only on edges of TI1, set SMS=0001 in the TIM1_SMCR register; if only on edges of TI2, set SMS=0010; if the counter counts on edges of both TI1 and TI2, set SMS=0011.

The polarity of TI1 and TI2 can be selected by setting the CC1P and CC2P bits in the TIM1_CCER register; the input filter can also be programmed if needed. CC1NP and CC2NP must be kept low.

The two inputs TI1 and TI2 are used as the interface for the quadrature encoder. Refer to the table below. Assuming the counter has been started (CEN=1 in the TIM1_CR1 register), the counter clock is driven by each valid signal transition on TI1FP1 or TI2FP2. TI1FP1=TI1 if not filtered and not inverted, TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIMx_CR1 register written to '1'). Based on the transition sequence of the two input signals, counting pulses and direction signals are generated. Depending on the transition sequence of the two input signals, the counter counts up or down, while the hardware sets the DIR bit in the TIM1_CR1 register accordingly. The DIR bit is calculated whenever a signal transition occurs on any input (TI1 or TI2), regardless of whether the counter counts only on edges of TI1 or TI2, or on both TI1 and TI2.

The encoder interface mode is equivalent to using an external clock with direction selection. This means the counter counts continuously only between 0 and the auto-reload value of the TIM1_ARR register (counting from 0 to ARR or from ARR to 0, depending on the direction). Therefore, TIM1_ARR must be configured before starting the count. Similarly, the capture, compare, prescaler, repetition counter, trigger output features, etc., continue to operate as usual. Encoder mode and External clock mode 2 are not compatible and must not be selected together.

Note: When enabling encoder mode, the prescaler must be set to zero.

In this mode, the counter is automatically updated based on the speed and direction of the quadrature encoder, so the counter content always indicates the encoder position. The count direction corresponds to the rotation direction of the connected sensor. The table summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 18-1 Counting direction versus encoder signals

Active edge	Level of opposite signals (TI1FP1 corresponds to TI2, TI2FP2 corresponds to TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Count only at TI1	High	De-creasing	Up	Not count	Not count
	Low	Up	De-creasing	Not count	Not count
Count only at TI2	High	Not count	Not count	Up	De-creasing
	Low	Not count	Not count	De-creasing	Up
Count at both TI1 and TI2	High	De-creasing	Up	Up	De-creasing
	Low	Up	De-creasing	De-creasing	Up

A quadrature encoder can directly connect to the MCU without requiring external interface logic. However, comparators are typically used to convert the encoder's differential output into digital signals, significantly enhancing noise immunity. The third encoder output which indicate the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

The figure below gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- CC1S='01' (TIM1_CCMR1 register, TI1FP1 mapped on TI1).
- CC2S='01' (TIM1_CCMR1 register, TI1FP2 mapped on TI2).
- CC1P='0', CC1NP='0' (TIM1_CCER register, TI1FP1 non-inverted, TI1FP1=TI1)
- CC2P='0', CC2NP='0' (TIM1_CCER register, TI1FP2 non-inverted, TI1FP2=TI2)
- SMS='0011' (TIM1_SMCR register, all inputs are active on both rising and falling edges)
- CEN='1' (TIM1_CR1 register, Counter enabled)

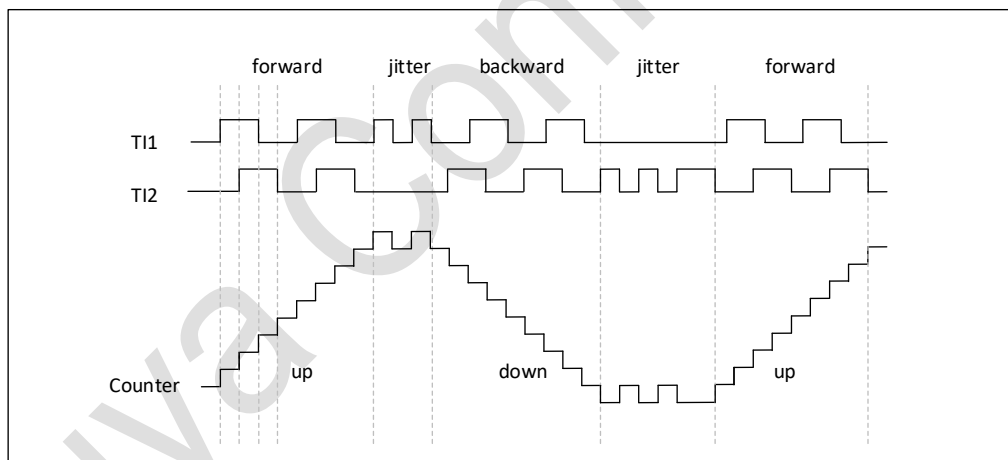


Figure 18-45 Example of counter operation in encoder interface mode

The following example illustrates the counter behavior when TI1FP1 polarity is inverted (same configuration as the previous example except CC1P="1").

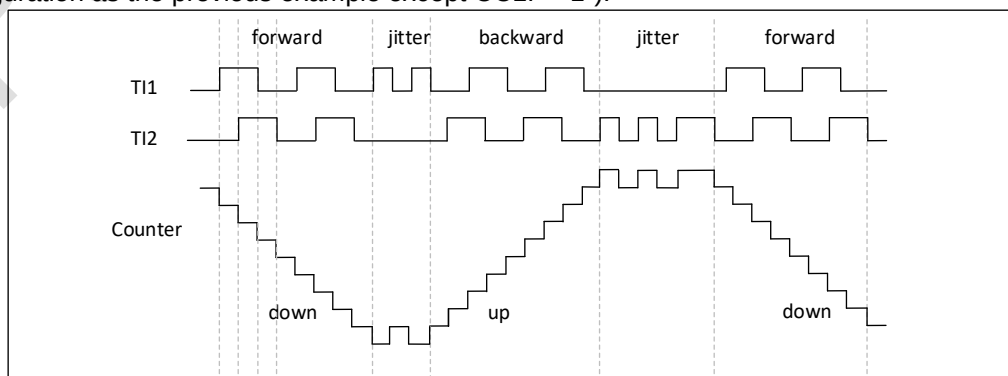


Figure 18-46 Encoder interface mode when TI1FP1 polarity is inverted

The timer, when configured in Encoder Interface mode provides information on the sensor's current position. Dynamic information can be obtained (speed, acceleration, deceleration) by measuring the period between two encoder events using a second timer configured in capture mode. The output of the encoder which indicates the mechanical zero can be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. This can be done by latching the counter value into a third input capture register if available (then the capture signal must be periodic and can be generated by another timer). when available, it is also possible to read its value through a DMA request generated by a real-time clock.

18.3.20. Timer input XOR mode

The TI1S bit in the TIM_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIM1_CH1, TIM1_CH2 and TIM1_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture. This facilitates measuring the interval between the rising edges of two input signals.

18.3.21. Connecting Hall sensors

When using an advanced timer (TIM1) to generate PWM signals for motor drive, another general-purpose timer (TIM2) can be used as an "interface timer" to connect Hall sensors. The three timer input pins (TIM2_CH1, TIM2_CH2, and TIM2_CH3) are connected to the TI1 input channel via an XOR gate (selected by setting the TI1S bit in the TIM2_CR2 register), and this signal is captured by the "interface timer."

The slave mode controller is configured in reset mode; the slave input is TI1F_ED. Thus, each time one of the 3 inputs toggles, the counter restarts counting from 0. This creates a time base triggered by any change on the Hall inputs.

The capture/compare channel 1 on the "interface timer" is configured in capture mode, with TRC as the capture signal. The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

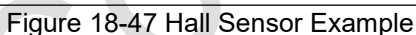
The "interface timer" can be used to generate a pulse in output mode, which can (by triggering a COM event) modify the properties of each channel in the advanced timer TIM1, which generates PWM signals for motor drive. Therefore, the interface timer channel must be programmed to generate a positive pulse after a specified delay (in output compare or PWM mode), and this pulse is sent to the advanced timer TIM1 via the TRGO output.

Example: Hall inputs are connected to the TIM2 timer, requiring the PWM configuration of the advanced control timer TIM1 to be changed at a specified time after any change in the Hall inputs.

- Set the TI1S bit in the TIM2_CR2 register to '1', so the three timer inputs are XORed and fed into the TI1 input channel.
- Timebase programming: Set TIM2_ARR to its maximum value (the counter must be cleared by changes in TI1). Set the prescaler to achieve a maximum counter period longer than the interval between two changes on the sensor.
- Configure channel 1 in capture mode (select TRC): Set CC1S=01 in the TIM2_CCMR1 register, and optionally configure a digital filter if needed.
- Configure channel 2 in PWM2 mode with the desired delay: Set OC2M=111 and CC2S=00 in the TIM2_CCMR1 register.
- Select OC2REF as trigger output on TRGO: write the MMS bits in the TIM2_CR2 register to '101'.

In the advanced control register TIM1, the correct ITR input must be selected as the trigger input.

The timer is programmed to generate PWM signals, with capture/compare control signals preloaded (CCPC=1 in the TIM1_CR2 register), and COM events controlled by the trigger input (CCUS=1 in the TIM1_CR2 register). After a COM event, the PWM control bits (CCxE, OCxM) are written with the next configuration, which can be implemented in the interrupt subroutine triggered by the rising edge of OC2REF.



The TIM1 timer can be synchronized with an external trigger in various modes: reset mode, gated mode, and trigger mode.

When a trigger input event occurs, the counter and its prescaler can be reinitialized. Additionally, if the URS bit in the TIM1_CR1 register is low, an update event (UEV) is generated, and all the preload registers (TIM1_ARR, TIM1_CCRx) are updated.

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). Since the capture prescaler is not used for trigger operations, no configuration is required. The CC1S bits select the input capture source only, CC1S=01 in TIM1_CCMR1 register. Set CC1P=0 and CC1NP=0 in the TIM1_CCER register to determine the polarity (only rising edges are detected).

- The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. At the same time, the trigger flag (TIF bit in the TIM1_SR register) is set, generating an interrupt request or a DMA request based on the settings of the TIE (interrupt enable) bit and TDE (DMA enable) bit in the TIM1_DIER register.

313 / 672

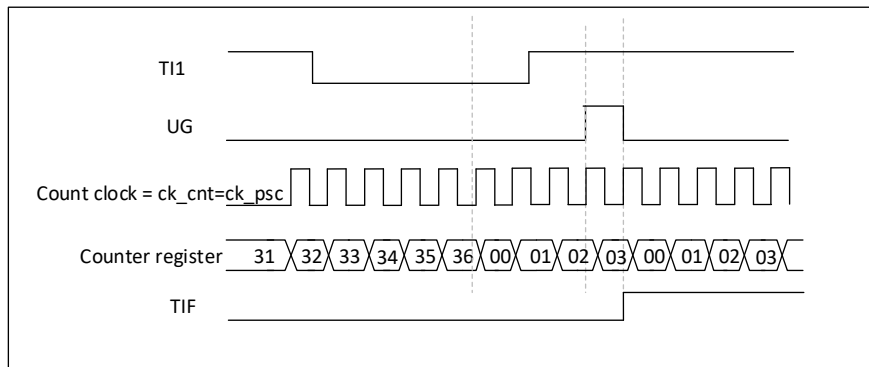


Figure 18-48 Control circuit in reset mode

18.3.22.2. Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the counter increments only when TI1 is low:

- Channel 1 is configured to detect the low level on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIM1_CCMR1 register. Set CC1P=1 in the TIM1_CCER register to determine the polarity (only low levels are detected).
- Set SMS=101 in the TIM1_SMCR register to configure the timer in gated mode. Set TS=00101 in the TIM1_SMCR register to select TI1 as the input source.
- Enable the counter by writing CEN=1 in the TIM1_CR1 register. In gated mode, if CEN=0, the counter cannot start regardless of the trigger input level.

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in TIM1_SR is set when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

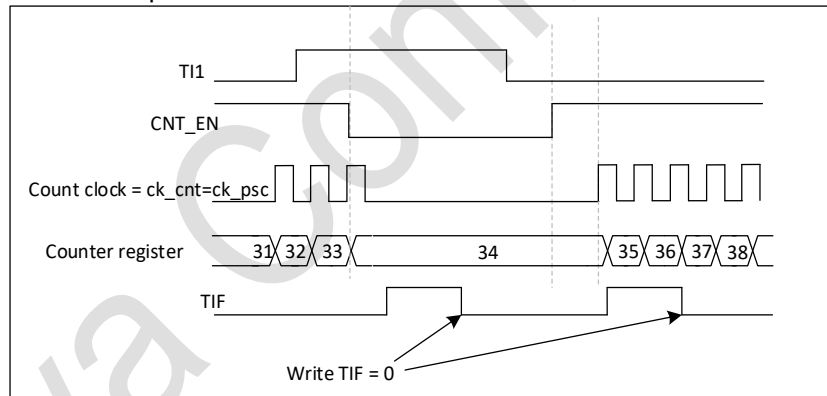


Figure 18-49 Control circuit in graph-gated mode

18.3.22.3. Slave mode: Trigger mode

The selected event on the input enables the counter.

In the following example, the counter starts incrementing on the rising edge of the TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we do not need any filter, so we keep IC2F=0000). The capture prescaler is not used for trigger operations, so no configuration is required. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIM1_CCMR1 register. Set CC2P=1 in the TIM1_CCER register to determine polarity (only detect low level).
- Set SMS=110 in the TIM1_SMCR register to configure the timer in trigger mode; set TS=00110 in the TIM1_SMCR register to select TI2 as the input source.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set. The delay between the rising edge of TI2 and the counter starting to count depends on the resynchronization circuit at the TI2 input.

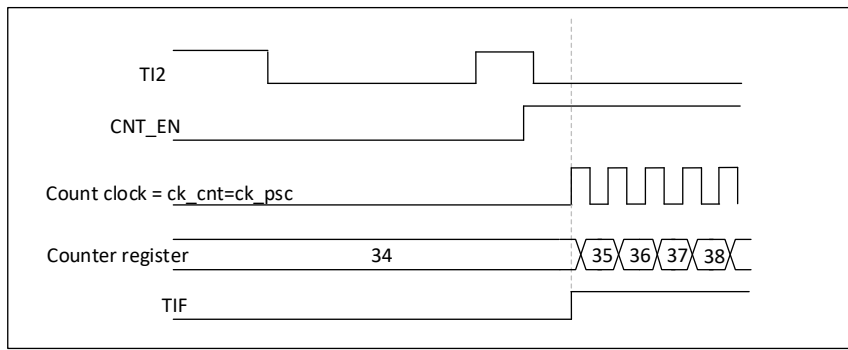


Figure 18-50 Control Circuit in Trigger Mode

18.3.22.4. Slave mode: Combined reset + trigger mode

In this case, a rising edge of the selected trigger input (TRGI) reinitializes the counter, generates an update of the registers, and starts the counter.

This mode is used for one-pulse mode.

18.3.22.5. Slave mode: external clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input when operating in reset mode, gated mode or trigger mode. It is not recommended to use the TS bit of the TIM1_SMCR register to select ETR as TRGI.

In the following example, once a rising edge appears on TI1, the counter increments once at each rising edge of ETR:

1. Configure the external trigger input circuit through the TIM1_SMCR register:
 - ETF = 0000: no filter
 - ETPS = 00: prescaler disabled
 - ETP=0: Detect the rising edge of ETR, set ECE=1 to enable external clock mode 2.
2. Configure the channel 1 as follows, to detect rising edges on TI1:
 - IC1F = 0000: no filter
 - The capture prescaler is not used in trigger operations and does not need to be configured.
 - CC1S=01 in TIM1_CCMR1 register to select only the input capture source
 - Set CC1P=0 and CC1NP=0 in the TIM1_CCER register to determine polarity (only detect rising edges).
3. Set SMS=0110 in the TIM1_SMCR register to configure the timer in trigger mode. Set TS=00101 in the TIM1_SMCR register to select TI1 as the input source.

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges. The delay between the rising edge of the ETR signal and the actual reset of the counter depends on the resynchronization circuit at the ETRP input.

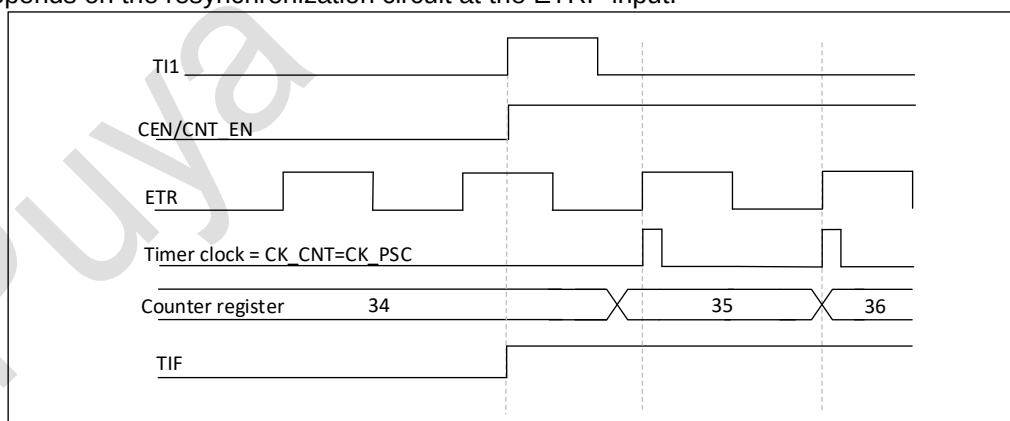


Figure 18-51 Control Circuit in External Clock Mode 2 + Trigger Mode

Note: The clock of the peripheral (timer, ADC, etc.) receiving the TRGO signal must be enabled first before it can receive events from the master timer; and the clock frequency (prescaler) must not be changed in real-time when receiving trigger signals from the master timer.

18.3.23. Timer synchronization

TIM timers are internally connected for timer synchronization or chaining functions. When one Timer is configured in Master mode, it can reset, start, stop or clock the counter of another Timer configured in Slave mode.

Refer to the description in TIM2/TIM3.

18.3.24. DMA Continuous Transfer Mode

The TIM1 timer can generate multiple DMA requests based on an event. The main purpose is to be able to re-program part of the timer multiple times without software overhead, but it can also be used to read several registers in a row, at regular intervals.

The DMA controller target is unique and must point to the virtual register TIM1_DMAR. When a given timer event occurs, the timer initiates a DMA request sequence (burst). Each write to the TIM1_DMAR register is redirected to one of the timer registers.

The DBL[4:0] bits in the TIM1_DCR register set the DMA continuous transfer length. When a read or write access is made to the TIM1_DMAR address, the timer performs a continuous transfer, i.e., the number of transfers (in half-words or bytes).

The DBA[4:0] bits in the TIM1_DCR register define the DMA base address for DMA transfers (when read/write access is performed via the TIM1_DMAR address). DBA is defined as an offset starting from the address of the TIM1_CR1 register:

Example:

00000: TIM1_CR1

00001: TIM1_CR2

00010: TIM1_SMCR

For example, the timer DMA continuous transfer feature is used to update the contents of the CCRx registers (x = 2, 3, 4) with multiple half-words transferred via DMA to the CCRx registers after an update event occurs.

To do this, use the following procedure:

1. Configure the corresponding DMA channel as follows:
 - The DMA channel peripheral address is the DMAR register address.
 - The DMA channel memory address is the RAM buffer address containing the data to be transferred via DMA to the CCRx registers.
 - The amount of data to be transferred = 3 (see note below).
 - Disable circular mode.
2. Configure the DCR register by setting the DBA and DBL bit fields as follows:
DBL = 3 transfers, DBA = 0xE.
3. Enable the TIM1 update DMA request (set the UDE bit in the DIER register).
4. Enable TIM1
5. Enable the DMA channel

This example applies to cases where each CCRx register is updated only once. If every CCRx register is to be updated twice for example, the number of data to transfer should be 6. Below is an example of a RAM buffer containing data1, data2, data3, data4, data5, and data6. Data will be transferred to the CCRx registers as follows: during the first update DMA request, data1 is transferred to CCR2, data2 to CCR3, and data3 to CCR4; during the second update DMA request, data4 is transferred to CCR2, data5 to CCR3, and data6 to CCR4.

Note: A null value can be written to the reserved registers.

18.4. TIM1 Interrupt

Table 18-2 Interrupt description table

Interrupt event	Event flag	Interrupt enable control bit
update	UIF	UIE
Compare/Capture 1	CC1IF	CC1IE
Compare/Capture 2	CC2IF	CC2IE
Compare/Capture 3	CC3IF	CC3IE
Compare/Capture 4	CC4IF	CC4IE
COM	COM	COMIE
Trigger	TIF	TIE
Brake	BIF	BIE
System brake	SBIF	BIE

18.5. Debug mode

When the chip enters debug mode, the TIM1 counter can continue normal operation or stop working based on the DBG_TIM1_STOP setting in the DBG module.

18.6. TIM1 registers

18.6.1. TIM1 control register 1 (TIM1_CR1)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	00	Clock division These 2 bits define the division ratio between the timer clock (CK_INT) frequency and the dead-time generator and sampling clock (t_{DTS}) used by the digital filter (ETR, TIx). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved, do not use this configuration
7	ARPE	RW	0	Auto-reload preload enable 0: TIM1_ARR register is not buffered 1: TIM1_ARR register is buffered
6:5	CMS[1:0]	RW	00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down based on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts alternately up and down. The output compare interrupt flag for channels configured as outputs (CCxS=00 in the TIM1_CCMRx register) is set only when the counter counts down. 10: Center-aligned mode 2. The counter counts alternately up and down. The output compare interrupt flag for channels configured as outputs (CCxS=00 in the TIM1_CCMRx register) is set only when the counter counts up. 11: Center-aligned mode 3. The counter counts alternately up and down. The output compare interrupt flag for channels configured as outputs (CCxS=00 in the TIM1_CCMRx register) is set both when the counter counts up and down. Note: When the counter is enabled (CEN=1), switching from edge-aligned mode to center-aligned mode is not allowed.
4	DIR	RW	0	Counting direction 0: Counter counts up 1: Counter counts down Note: This bit is read-only when the counter is configured in center-aligned mode or encoder mode.
3	OPM	RW	0	One-pulse mode 0: Counter is not stopped at update event

				1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source Software selects the UEV event source through this bit. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: – Counter overflow/underflow – Setting the UG bit – Update event generated by the slave mode controller 1: If update interrupt or DMA request is enabled, only counter overflow/underflow generates an update interrupt or DMA request
1	UDIS	RW	0	Update disable Software enables/disables UEV event generation through this bit. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow – Setting the UG bit – Update event generated by the slave mode controller Shadow registers are loaded with preload values. 1: UEV disabled. The update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). The counter and prescaler are reinitialized if the UG bit is set or a hardware reset is issued by the slave mode controller.
0	CEN	RW	0	Counter enable 0: Counter disabled 1: Enable counter Note: External clock, gated mode, and encoder mode can only be used after software sets the CEN bit. Trigger mode can automatically set the CEN bit to 1 via hardware.

18.6.2. TIM1 control register 2 (TIM1_CR2)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	MMS[3] RW	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Res.	CCPC
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	MMS[3]	RW	0	Refer to MMS[2:0] description
00:15	Reserved	-	-	Reserved
14	OIS4	RW		Output idle state 4 (OC4 output) Refer to OIS1 bit.
13	OIS3N	RW	0	Output idle state 3 (OC3N output) Refer to OIS1N bit
12	OIS3	RW	0	Output idle state 3 (OC3 output) Refer to OIS1 bit.
11	OIS2N	RW	0	Output idle state 2 (OC2N output) Refer to OIS1N bit
10	OIS2	RW	0	Output idle state 2 (OC2 output) Refer to OIS1 bit
9	OIS1N	RW	0	Output idle state 1 (OC1N output) 0: When MOE=0, (if OC1N is active, after dead-time) OC1N=0 1: When MOE=0, (if OC1N is valid, then after dead time) OC1N=1

				Note: this bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register).
8	OIS1	RW	0	Output idle state 1 (OC1 output) 0: OC1=0 after a dead-time when MOE=0 1: OC1=0 after a dead-time when MOE=1 Note: this bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register).
7	TI1S	RW	0	TI1 selection 0: The TIM1_CH1 pin is connected to TI1 input 1: TIM1_CH1, TIM1_CH2, and TIM1_CH3 pins are XORed and connected to the TI1 input.
6:4	MMS[2:0]	RW	3'b0	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 0000: Reset – The UG bit of the TIM1_EGR register is used as the trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 0001: Enable – The counter enable signal CNT_EN is used as the trigger output (TRGO). Sometimes it is necessary to start multiple timers simultaneously or control a slave timer for a period of time. When configured in gated mode, the counter enable is generated by the logical AND of the CEN control bit and the trigger input signal. When the counter enable signal is controlled by the trigger input, there will be a delay on TRGO unless master/slave mode is selected (see the description of the MSM bit in the TIM1_SMCR register). 0010: Update – The update event is used as the trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 0011: Compare pulse – Once a capture or compare occurs successfully, when the CC1IF flag is set to 1, the trigger output sends a positive pulse (TRGO). 0100: Compare – The OC1REF signal is used as the trigger output (TRGO). 0101: Compare – The OC2REF signal is used as the trigger output (TRGO). 0110: Compare – The OC3REF signal is used as the trigger output (TRGO). 0111: Compare – The OC4REF signal is used as the trigger output (TRGO). 1000: Encoder clock output – The encoder clock signal is used as the trigger output, available only when SMS[3:0] is set to 0001, 0010, or 0011.
3	CCDS	RW	0	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
2	CCUS	RW	0	Capture/compare control update selection 0: If the capture/compare control bits are preloaded (CCPC=1), they can only be updated by setting the COMG bit. 1: If the capture/compare control bits are preloaded (CCPC=1), they can be updated by setting the COMG bit or a rising edge on TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved	-	-	Reserved
0	CCPC	RW	0	Capture/compare preloaded control

				0: CCxE, CCxNE and OCxM bits are not preloaded. 1: CCxE, CCxNE, and OCxM bits are preloaded; after setting this bit, these bits are updated only when a commutation event (COM) occurs (COMG bit set to 1 or a rising edge detected on TRGI, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output.

18.6.3. TIM1 slave mode control register (TIM1_SMCR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS[4:3]		Res.	Res.	Res.	SMS[3]
										RW	RW				RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]		OCCS	SMS[2:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
21:20	TS[4:3]	RW	3'b0	See TS description for details
19:17	Reserved	-	-	Reserved
16	SMS[3]	RW	0	See SMS description for details
15	ETP	RW	0	External trigger polarity This bit selects whether ETR or its inversion is used for trigger operation. 0: ETR is not inverted, active high or rising edge 1: ETR inverted, active low or falling edge
14	ECE	RW	0	External clock enable. This enables External Clock Mode 2. 0: External clock mode 2 disabled; 1: Enable External Clock Mode 2, where the counter clock is provided by the active edge of the ETRF signal; Notes: 1: Setting the ECE bit has the same effect as selecting External Clock Mode 1 and connecting TRGI to ETRF (SMS=111 and TS=00111). 2: External Clock Mode 2 can be used simultaneously with the following slave modes: Reset Mode, Gated Mode, and Trigger Mode. However, in such cases, TRGI must not be connected to ETRF (TS bits must not be 00111). 3: If both External Clock Mode 1 and External Clock Mode 2 are enabled, the external clock input is ETRF.
13:12	ETPS[1:0]	RW	2'b0	External Trigger Prescaler The frequency of the external trigger signal ETRP must not exceed 1/4 of the TIM1CLK frequency. The frequency of ETR can be reduced by enabling the prescaler. This method is very useful when inputting a fast external clock. 00: Prescaler OFF 01: ETR frequency divided by 2 10: ETR frequency divided by 4 11: ETR frequency divided by 8

11:8	ETF[3:0]	RW	4'b0	External Trigger Filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. In practice, the digital filter is an event counter that records N events before considering a valid output edge. 0000: No filter, sampling is done at f_{DTS} 0001: $f_{SAMPLING} = f_{CK\_INT}$, $N = 2$ 0010: $f_{SAMPLING} = f_{CK\_INT}$, $N = 4$ 0011: $f_{SAMPLING} = f_{CK\_INT}$, $N = 8$ 0100: $f_{SAMPLING} = f_{DTS}/2$, $N = 6$ 0101: $f_{SAMPLING} = f_{DTS}/2$, $N = 8$ 0110: Sampling frequency $f_{SAMPLING} = f_{DTS}/4$, $N=6$ 0111: Sampling frequency $f_{SAMPLING} = f_{DTS}/4$, $N=8$ 1000: Sampling frequency $f_{SAMPLING} = f_{DTS}/8$, $N=6$ 1001: Sampling frequency $f_{SAMPLING} = f_{DTS}/8$, $N=8$ 1010: Sampling frequency $f_{SAMPLING} = f_{DTS}/16$, $N=5$ 1011: Sampling frequency $f_{SAMPLING} = f_{DTS}/16$, $N=6$ 1100: $f_{SAMPLING} = f_{DTS}/16$, $N = 8$ 1101: $f_{SAMPLING} = f_{DTS}/32$, $N = 5$ 1110: $f_{SAMPLING} = f_{DTS}/32$, $N = 6$ 1111: $f_{SAMPLING} = f_{DTS}/32$, $N = 8$
7	MSM	RW	0	Master/slave mode 0: No operation 1: The event on the trigger input (TRGI) is delayed to achieve perfect synchronization between the current timer and its slave timer (via TRGO). It is useful if we want to synchronize several timers on a single external event.
6:4	TS[2:0]	RW	3'b0	Trigger selection These 5 bits select the trigger input used to synchronize the counter. 00000: Internal Trigger 0 (ITR0) 00001: Internal Trigger 1 (ITR1) 00010: Internal Trigger 2 (ITR2) 00011: Internal Trigger 3 (ITR3) 00100: TI1 edge detection (TI1F_ED) 00101: Filtered timer input 1 (TI1FP1) 00110: Filtered timer input 2 (TI1FP2) 00111 external trigger input (ETRF) 01000: Internal Trigger 4 (ITR4) 01001: Internal Trigger 5 (ITR5) Others: Reserved Note: These bits can only be changed when unused (e.g., SMS=0000) to avoid false edge detection during transitions. Note: For ITRx mapping, refer to the "Peripheral Interconnection" section.
3	OCCS	RW	0	OCCREF clear selection bit. This bit is used to select the clear source for OCCREF. 0: OCCREF_CLR_INT is connected to the OCCREF_CLR input (COMP1 or COMP2 output).

				1: OCREF_CLR_INT is connected to ETRF
2:0	SMS[2:0]	RW	3'b0	Slave mode selection When an external signal is selected, the active edge of the trigger signal (TRGI) depends on the selected polarity of the external input (see descriptions of the input control register and control register). 0000: Slave mode disabled—if CEN=1, the prescaler is directly driven by the internal clock. 0001: Encoder mode 1—the counter counts up/down on edges of TI1FP1 depending on the level of TI2FP2. 0010: Encoder mode 2—the counter counts up/down on edges of TI2FP2 depending on the level of TI1FP1. 0011: Encoder mode 3—the counter counts up/down on edges of TI1FP1 and TI2FP2 depending on the level of the other input. 0100: Reset mode—the rising edge of the selected trigger input (TRGI) reinitializes the counter and generates a register update signal. 0101: Gated mode—the counter clock is enabled while the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 0110: Trigger mode—the counter starts (but is not reset) on the rising edge of the trigger input TRGI, controlling only the counter's start. 0111: External clock mode 1—the rising edge of the selected trigger input (TRGI) drives the counter. Note: Gated mode must not be used when TI1F_EN is selected as the trigger input (TS=00100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal. If SMS[3]=1, SMS[2:0] must be configured as 0. 1000: "Combined reset + trigger" mode—the rising edge of the selected trigger input (TRGI) reinitializes the counter, generates a register update, and starts the counter. Note: In encoder mode, do not use UEV as the TRGO output signal (i.e., MMS must not be configured as 010).

18.6.4. TIM1 DMA/interrupt enable register (TIM1_DIER)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TD	COMD	CC4D	CC3D	CC2D	CC1D	UD	BIE	TIE	COMI	CC4I	CC3I	CC2I	CC1IE	UIE
	E	E	E	E	E	E	E			E	E	E	E		

	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Bit	Name	R/W	Reset Value	Function										
31:15	Reserved	-	-	Reserved										
14	TDE	RW	0	Trigger DMA request enable 0: Trigger DMA request disabled 1: Trigger DMA request enabled										
13	COMDE	RW	0	COM DMA request enable 0: COM DMA request disabled 1: COM DMA request enabled										
12	CC4DE	RW	0	Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled 1: CC4 DMA request enabled										
11	CC3DE	RW	0	Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled 1: CC3 DMA request enabled										
10	CC2DE	RW	0	Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled 1: CC2 DMA request enabled										
9	CC1DE	RW	0	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled 1: CC1 DMA request enabled										
8	UDE	RW	0	Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled										
7	BIE	RW	0	Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled										
6	TIE	RW	0	Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled										
5	COMIE	RW	0	COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled										
4	CC4IE	RW	0	Capture/Compare 4 interrupt enable 0: Capture/Compare 4 interrupt disabled 1: Capture/Compare 4 interrupt enabled										
3	CC3IE	RW	0	Capture/Compare 3 interrupt enable 0: Capture/Compare 3 interrupt disabled 1: Capture/Compare 3 interrupt enabled										
2	CC2IE	RW	0	Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled										
1	CC1IE	RW	0	Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled										
0	UIE	RW	0	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled										

18.6.5. TIM1 status register (TIM1_SR)

Offset address: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Re s.	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	Res.	Res.	Res.	Res.	IC2IR	IC1IR	Res.	Res.
		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0					RC_W0	RC_W0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	Re s.	SBIF	CC4 OF	CC3 OF	CC2 OF	CC1 OF	Res.	BIF	TIF	COMI F	CC4I F	CC3I F	CC2I F	CC1F	UIF
		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29	IC4IF	RC_W0	0	Falling edge capture 4 flag Refer to IC1IF description
28	IC3IF	RC_W0	0	Falling edge capture 3 flag Refer to IC1IF description
27	IC2IF	RC_W0	0	Falling edge capture 2 flag Refer to IC1IF description
26	IC1IF	RC_W0	0	Falling edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by a falling edge. It is cleared by software or by reading TIM1_CCR1. 0: No falling edge capture occurs; 1: A falling edge capture event occurs.
25	IC4IR	RC_W0	0	Rising edge capture 4 flag Refer to IC1IR description
24	IC3IR	RC_W0	0	Rising edge capture 3 flag Refer to IC1IR description
23:20	Reserved	-	-	Reserved
19	IC2IR	RC_W0	0	Rising edge capture 2 flag Refer to IC1IR description
18	IC1IR	RC_W0	0	Rising edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by a rising edge. It is cleared by software or by reading TIM1_CCR1. 0: No rising edge capture occurs; 1: A rising edge capture event occurs.
17:14	Reserved	-	-	Reserved
13	SBIF	RC_W0	0	System break interrupt flag This flag is set by hardware as soon as the system brake input is valid. The system brake input can be cleared via software after becoming invalid. This flag must be reset to allow PWM to resume operation. 0: No brake event has occurred. 1: A valid level has been detected on the system brake input. An interrupt will be generated if the BIE bit of the TIM1_DIER register = 1.
12	CC4OF	RC_W0	0	Capture/Compare 4 overcapture flag Refer to CC1OF description
11	CC3OF	RC_W0	0	Capture/Compare 3 overcapture flag Refer to CC1OF description
10	CC2OF	RC_W0	0	Capture/Compare 2 overcapture flag Refer to CC1OF description
9	CC1OF	RC_W0	0	Capture/Compare 1 overcapture flag This flag can be set to 1 by hardware only when the corresponding channel is configured as input capture. Writing 0 via software clears this bit. 0: No overcapture has occurred; 1: The TIM1_CCR1 register has captured the counter value and CC1IF has been set.
8	Reserved	-	-	Reserved
7	BIF	RC_W0	0	Brake interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No brake event has occurred;

				1: An active level has been detected on the break input. An interrupt will be generated if BIE=1 in the TIM1_DIER register.
6	TIF	RC_W0	0	Trigger interrupt flag In all modes except gated mode, this bit is set to 1 by hardware when a valid edge is detected at the TRGI input after enabling the slave mode controller. In gated mode, this flag will be set to 1 when the counter starts or stops, but it must be cleared by software. 0: No trigger event has occurred; 1: Trigger interrupt and wait for response
5	COMIF	RC_W0	0	COM interrupt flag This bit is set to 1 by hardware once a COM event occurs (when CCxE, CCxNE, OCxM have been updated). It is cleared by software. 0: No COM event has occurred; 1: COM interrupt pending.
4	CC4IF	RC_W0	0	Capture/Compare 4 interrupt flag Refer to CC1IF description
3	CC3IF	RC_W0	0	Capture/Compare 3 interrupt flag Refer to CC1IF description
2	CC2IF	RC_W0	0	Capture/Compare 2 interrupt flag Refer to CC1IF description
1	CC1IF	RC_W0	0	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This bit is set by hardware when the counter value matches the compare value, except in center-aligned mode (refer to the CMS bit in the TIM1_CR1 register). It is cleared by software. 0: No match; 1: The content of the counter TIM1_CNT matches the content of the TIM1_CCR1 register. Or when the value of TIM1_CCR1 is greater than TIM1_ARR, CC1IF will be set to 1 during overflow in up-counting and center-aligned modes, or during underflow in down-counting. If channel CC1 is configured as input: This bit is set by hardware when a capture event occurs and is cleared by software or by reading TIM1_CCR1. 0: No input capture occurred; 1: The TIM1_CCR1 register has captured the counter value (an edge with the selected polarity has been detected on IC1).
0	UIF	RC_W0	0	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update event occurred; 1: Update interrupt pending. This bit is set by hardware when the registers are updated: – If UDIS=0 in the TIM1_CR1 register, an update event is generated when REP_CNT=0 (on overflow or underflow of the repetition down-counter); – If UDIS=0 and URS=0 in the TIM1_CR1 register, an update event is generated when UG=1 in the TIM1_EGR register (software reinitializes CNT); – If UDIS=0 and URS=0 in the TIM1_CR1 register, an update event is generated when CNT is reinitialized by a trigger event. (Refer to TIM1 slave mode control register (TIM1_SMCR))

18.6.6. TIM1 event generation register (TIM1_EGR)

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
								W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	BG	W	0	Break generation This bit is set by software to generate a break event and is cleared by hardware automatically. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt or DMA transfer can occur if enabled.
6	TG	W	0	Trigger generation This bit is set by software to generate a trigger event and is cleared by hardware automatically. 0: No action 1: TIF=1 in the TIM1_SR register, generating the corresponding interrupt and DMA if enabled.
5	COMG	W	0	Capture/Compare control update generation This bit is set by software and cleared by hardware automatically. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit acts only on channels that have a complementary output.
4	CC4G	W	0	Capture/Compare 4 generation Refer to CC1G description
3	CC3G	W	0	Capture/Compare 3 generation Refer to CC1G description
2	CC2G	W	0	Capture/Compare 2 generation Refer to CC1G description
1	CC1G	W	0	Capture/Compare 1 generation This bit is set by software to generate a capture/compare event and is cleared by hardware automatically. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as input: When enabled, the CC1IF flag is set to 1 and sends the corresponding interrupt or DMA request. If channel CC1 is configured as input: The current value of the counter will be captured in the TIMx_CCR1 register. When enabled, the CC1IF flag is set to 1 and the corresponding interrupt or DMA request is generated. The CC1OF flag is set if the CC1IF flag was already set.
0	UG	W	0	Update generation This bit is set by software and cleared by hardware automatically. 0: No action 1: Reinitialize the counter and generate an update of the registers.

				Note: The prescaler counter is also cleared (but the prescaler ratio remains unchanged). If in center-aligned mode or DIR=0 (upcounting), the counter is cleared; if DIR=1 (downcounting), the counter is automatically loaded with the TIM1_ARR value.

18.6.7. TIM1 Capture/Compare Mode Register 1 (TIM1_CCMR1)

Offset address: 0x18

Reset value: 0x0000 0000

The same register can be used for input capture mode or output compare mode. The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function for input capture and for output compare modes. For any given bit, OCxx indicates the function of the bit when the channel is configured as an output, while ICxx indicates the function when the channel is configured as an input.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[3]
							RW								RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2C E	OC2M [2:0]			OC2P E	CO2F E	CC2S [1:0]		OC1C E	OC1M [2:0]			OC1P E	OC1F E	CC1S [1:0]	
IC2F [3:0]				IC2PSC [1:0]				IC1F [3:0]			IC1PSC [1:0]				
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	R W	RW

18.6.7.1. Input capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	IC2F [3:0]	RW	4'b0	Input capture 2 filter. Refer to the description of IC1F[3:0].
11:10	IC2PSC [1:0]	RW	2'b0	Input/Capture 2 prescaler. Refer to the description of IC1PSC[1:0].
9:8	CC2S [1:0]	RW	2'b0	Capture/Compare 2 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM1_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIM1_CCER).
7:4	IC1F [3:0]	RW	4'b0	Input capture 1 filter. This bit field defines the sampling frequency of the TI1 input and the sampling length of the digital filter applied to TI1. The digital filter consists of an event counter, where only every N consecutive events are considered a valid output edge: 0000: No filter, sampling is done at f_{DTS} 0001: $f_{SAMPLING} = f_{CK\_INT}$, N = 2 0010: $f_{SAMPLING} = f_{CK\_INT}$, N = 4 0011: $f_{SAMPLING} = f_{CK\_INT}$, N = 8 0100: $f_{SAMPLING} = f_{DTS}/2$, N = 6 0101: $f_{SAMPLING} = f_{DTS}/2$, N = 8 0110: $f_{SAMPLING} = f_{DTS}/4$, N = 6 0111: $f_{SAMPLING} = f_{DTS}/4$, N = 8 1000: $f_{SAMPLING} = f_{DTS}/8$, N = 6 1001: $f_{SAMPLING} = f_{DTS}/8$, N = 8 1010: $f_{SAMPLING} = f_{DTS}/16$, N = 5 1011: $f_{SAMPLING} = f_{DTS}/16$, N = 6

				1100: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 8$ 1101: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 5$ 1110: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 6$ 1111: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 8$
3:2	IC1PSC [1:0]	RW	2'b0	Input/Capture 1 prescaler. This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0' (TIM1_CCER register). 00: No prescaler, every edge detected on the capture input triggers a capture; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S [1:0]	RW	2'b0	CC1S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM1_SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM1_CCER).

18.6.7.2. Output compare mode

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	OC2M[3]	RW	0	See OC2M description
23:17	Reserved	-	-	Reserved
16	OC1M[3]	RW	0	See OC1M description
15	OC2CE	RW	0	Output Compare 2 clear enable Refer to the description of OC1CE.
14:12	OC2M [2:0]	RW	3'b0	Output Compare 2 mode Refer to the description of OC1M.
11	OC2PE	RW	0	Output Compare 2 preload enable Refer to the description of OC1PE.
10	OC2FE	RW	0	Output Compare 2 fast enable Refer to the description of OC1FE.
9:8	CC2S [1:0]	RW	2'b0	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM1_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIM1_CCER).
7	OC1CE	RW	0	Output Compare 1 clear enable 0: OC1REF is not affected by ETRF or OCREF_CLR input; 1: Once a high level is detected on ETRF or OCREF_CLR input, OC1REF is cleared.
6:4	OC1M [2:0]	RW	3'b0	Output compare 1 mode

				This bit defines the behavior of the output reference signal OC1REF for OC1 and OC1N. OC1REF is active high, while the active level of OC1 and OC1N depends on the CC1P and CC1NP bits. 0000: Frozen. The comparison between the output comparison register TIM1_CCR1 and the counter TIM1_CNT does not work for OC1REF; 0001: Set channel 1 to output active level upon match. When the value of counter TIM1_CNT matches the capture/compare register 1 (TIM1_CCR1), force OC1REF to high level. 0010: Set channel 1 to output inactive level upon match. When the value of counter TIM1_CNT matches the capture/compare register 1 (TIM1_CCR1), force OC1REF to low level. 0011: Toggle When TIM1_CCR1 = TIM1_CNT, OC1REF toggles. 0100: Force inactive level Force OC1REF to low level. 0101: Force active level Force OC1REF to high level. 0110: PWM mode 1—In up-counting, channel 1 is active when TIM1_CNT < TIM1_CCR1, otherwise inactive; in down-counting, channel 1 is inactive (OC1REF=0) when TIM1_CNT > TIM1_CCR1, otherwise active (OC1REF=1). 0111: PWM mode 2—In up-counting, channel 1 is inactive when TIM1_CNT < TIM1_CCR1, otherwise active; in down-counting, channel 1 is active when TIM1_CNT > TIM1_CCR1, otherwise inactive. 1000: Retriggerable OPM mode 1—In up-counting, the channel remains active until a trigger event (TRGI signal) is detected. Then, the comparison is performed as in PWM mode 1, and the channel becomes active again at the next update; in down-counting mode, the channel remains inactive until a trigger event (TRGI signal) is detected. Then, the comparison is performed as in PWM mode 1, and the channel becomes inactive again at the next update. 1001: Retriggerable OPM mode 2—In up-counting, the channel remains inactive until a trigger event (TRGI signal) is detected. Then, the comparison is performed as in PWM mode 2, and the channel becomes inactive again at the next update; in down-counting mode, the channel remains active until a trigger event (TRGI signal) is detected. Then, the comparison is performed as in PWM mode 2, and the channel becomes active again at the next update event. Others: Reserved Note 1: Once the LOCK level is set to 3 (LOCK bits in TIM1_BDTR register) and CC1S=00 (channel configured as output), this bit cannot be modified. Note: In PWM mode 1 or 2, the OCREF level changes when the result of the comparison changes or when the output compare mode switches from frozen to PWM mode. Note 3: When using the recoverable OPM mode, do not configure the counting mode as center-aligned counting mode.
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Disable the preload function of TIM1_CCR1 register. TIM1_CCR1 register can be written at any time, and the new value takes effect immediately. 1: Enable the preload function of TIM1_CCR1 register. Read/write access to the preload register is allowed.

				The preload value of TIM1_CCR1 is loaded into the active register at each update event. Note 1: Once the LOCK level is set to 3 (LOCK bits in TIM1_BDTR register) and CC1S=00 (channel configured as output), this bit cannot be modified. Note 2: PWM mode can only be used without verifying the preload register in single-pulse mode (OPM bit in TIM1_CR1 register set to 1). Else the behavior is not guaranteed.
2	OC1FE	RW	0	Output Compare 1 fast enable This bit is used to accelerate the effect of trigger input events on CC outputs. 0: Even if the trigger is active, CC1 will operate normally based on the counter and CCR1 values. When a trigger input edge occurs, the minimum delay to activate CC1 output is 5 clock cycles. 1: A valid edge on the trigger input is equivalent to a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. The delay between sampling the trigger input and CC1 output is reduced to 3 clock cycles. OCxFE is effective only when the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	RW	2'b0	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM1_SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM1_CCER).

18.6.8. TIM1 Capture/Compare Mode Register 2 (TIM1_CCMR2)

Offset address: 0x1C

Reset value: 0x0000 0000

Refer to the description of TIM_CCMR1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4M[3 RW	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC3 M[3] RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4C E	OC4M [2:0]			OC4P E	CO4F E	CC4S [1:0]		OC3C E	OC3M [2:0]			OC3P E	OC3FE	CC3S [1:0]	
IC4F [3:0]				IC4PSC [1:0]			IC3F [3:0]			IC3PSC [1:0]					
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

18.6.8.1. Input capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	IC4F [3:0]	RW	4'b0	Input capture 4 filter. Refer to the description of IC1F.
11:10	IC4PSC [1:0]	RW	2'b0	Input/Capture 4 prescaler. Refer to the description of IC1PSC.
9:8	CC4S [1:0]	RW	2'b0	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output

				01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3. 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM1_SMCR register). Note: CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIM1_CCER).
7:4	IC3F [3:0]	RW	4'b0	Input capture 3 filter. Refer to the description of IC1F.
3:2	IC3PSC [1:0]	RW	2'b0	Input/Capture 3 prescaler. Refer to the description of IC1PSC.
1:0	CC3S [1:0]	RW	2'b0	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3. 10: CC3 channel is configured as input, IC3 is mapped on TI4. 11: CC3 channel is configured as input, IC1 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM1_SMCR register). Note: CC3S bits are writable only when the channel is OFF (CC3E = '0' in TIM1_CCER).

18.6.8.2. Output compare mode

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	OC4M[3]	RW	0	Refer to the description of OC4M.
23:17	Reserved	-	-	Reserved
16	OC3M[3]	RW	0	Refer to the description of OC3M.
15	OC4CE	RW	0	Output Compare 4 clear enable. Refer to the description of OC1CE.
14:12	OC4M [2:0]	RW	3'b0	Output Compare 4 mode. Refer to the description of OC1M.
11	OC4PE	RW	0	Output Compare 4 preload enable. Refer to the description of OC1PE.
10	OC4FE	RW	0	Output Compare 4 fast enable. Refer to the description of OC1FE.
9:8	CC4S [1:0]	RW	2'b0	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3. 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM1_SMCR register). Note: CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIM1_CCER).
7	OC3CE	RW	0	Output Compare 3 clear enable. Refer to the description of OC1CE.
6:4	OC3M [2:0]	RW	3'b0	Output Compare 3 mode. Refer to the description of OC1M.
3	OC3PE	RW	0	Output Compare 3 preload enable.

				Refer to the description of OC1PE.
2	OC3FE	RW	0	Output Compare 3 fast enable. Refer to the description of OC1FE.
1:0	CC3S [1:0]	RW	2'b0	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3. 10: CC3 channel is configured as input, IC3 is mapped on TI4. 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM1_SMCR register). Note: CC3S bits are writable only when the channel is OFF (CC3E = '0' in TIM1_CCER).

18.6.9. TIM1 capture/compare enable register (TIM1_CCER)

Offset address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13	CC4P	RW	0	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	RW	0	Input/Capture 4 output enable Refer to CC1E description.
11	CC3NP	RW	0	Input/Capture 3 complementary output polarity. Refer to CC1NP description.
10	CC3NE	RW	0	Input/Capture 3 complementary output enable. Refer to CC1NE description.
9	CC3P	RW	0	Input/Capture 3 output polarity Refer to CC1P description.
8	CC3E	RW	0	Input/Capture 3 output enable Refer to CC1E description.
7	CC2NP	RW	0	Input/Capture 2 complementary output polarity. Refer to CC1NP description.
6	CC2NE	RW	0	Input/Capture 2 complementary output enable. Refer to CC1NE description.
5	CC2P	RW	0	Input/Capture 2 output polarity Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 output enable Refer to CC1E description.
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity. 0: OC1N active high. 1: OC1N active low. Note: Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 3 or 2 and CC1S=00 (channel configured as output), this bit cannot be modified.
2	CC1NE	RW	0	Input/Capture 1 complementary output enable 0: Off - OC1N output disabled, its output level depends on the values of MOE, OSSR, OISR1, OISR1N, and CC1E bits.

				1: On - OC1N signal output to the corresponding output pin, its output level depends on the values of MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits. On channels having a complementary output, this bit is preloaded. If the CCPC bit in the TIM1_CR2 register is set to 1, the CC1NE valid bit will only take the new value from the preload bit when a commutation event (COM) is generated.
1	CC1P	RW	0	Input/Capture 1 output polarity. CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: The [CC1NP, CC1P] bits select the polarity of TI1FP1 and TI2FP1 as trigger or capture signals: 00: Non-inverted/rising edge. TIxFP1 rising edge active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 non-inverted (gated mode, encoder mode). 01: Inverted/falling edge. TIxFP1 falling edge active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 inverted (gated mode, encoder mode). 10: reserved, do not use this configuration. 11: Non-inverted/both edges. TIxFP1 rising and falling edges both active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 non-inverted (gated mode). This configuration must not be used in encoder mode. Notes: 1. For complementary output channels, this bit is preloaded. If the CCPC bit in the TIM1_CR2 register is set to 1, the actual valid bit of CC1P will only load the pre-load value when a COM event occurs. Note: Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 3 or 2 and CC1S=00 (channel configured as output), this bit cannot be modified.
0	CC1E	RW	0	Input/Capture 1 output enable CC1 channel configured as output: 0: Off - OC1 output disabled, its output level depends on the values of MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits. 1: On - OC1 signal output to the corresponding output pin, its output level depends on the values of MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIM1_CCR1) or not. 0: Capture disabled 1: Capture enabled Note: For complementary output channels, this bit is preloaded. If the CC1PC bit in the TIM1_CR2 register is set, the actual valid bit of CC1E will only load the pre-load value when a COM event occurs.

Table 18-3 Output control bits for complementary channels OCx and OCxN with brake function

Control bits					Output states	
MOE	OSSI	OSSR	CCxE	CCxNE	OCx output status	OCxN output state
1	x	x	0	0	Output disabled (not timer-driven), OCx=0, OCx_EN=0	Output disabled (not timer-driven), OCxN=0, OCxN_EN=0

		0	0	1	Output disabled (not timer-driven), OCx=0, OCx_EN=0	OCxREF + polarity OCxN=OCxREF XOR CCxNP, OCxN_EN=1
		0	1	0	OCxREF + polarity OCx=OCREF xor CCxP, OCx_EN=1	Output disabled (not timer-driven), OCxN=0, OCxN_EN=0
		x	1	1	OCREF + polarity + dead time, OCx_EN=1	Complement of OCREF (not OCREF) + polarity + dead time, OCxN_EN=1
		1	0	1	Off state (output enabled and at inactive level), OCx=CCxP, OCx_EN=1	OCxREF + polarity OCxN=OCxREF XOR CCxNP, OCxN_EN=1
		1	1	0	OCxREF + polarity OCx=OCxREF XOR CCxP, OCx_EN=1	Off-State (output enabled with inactive state), OCxN=CCxNP, OCxN_EN=1
0	0	x	x	x	Output disabled (not timer-driven)	
	1		0	0		
	1		0	1	Off-State (output enabled with inactive state) Asynchronous: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1 (if BRK is triggered). Subsequently (only valid when BRK is triggered), if the clock is present: after a dead time, assuming OISx and OISxN do not both correspond to the active levels of OCx and OCxN, then OCx=OISx and OCxN=OISxN.	
	1		1	0		
	1		1	1		

18.6.10. TIM1 counter register (TIM1_CNT)

Offset address: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	0	Counter value

18.6.11. TIM1 prescaler register (TIM1_PSC)

Offset address: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	0	Prescaler value. The counter clock frequency (CK_CNT) is equal to $f_{CK\_PSC} / (PSC[15:0] + 1)$. PSC contains the value to be loaded into the active prescaler register each time an update event occurs (including when the counter is cleared via the UG bit in the TIM1_EGR register, or when cleared via the trigger controller in "reset mode").

18.6.12. TIM1 Auto-reload Register (TIM1_ARR)

Offset address: 0x2C

Reset value: 0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	ARR[15:0]	RW	16'hFFFF	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

18.6.13. TIM1 Repetition Counter Register (TIM1_RCR)

Offset address: 0x30

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	REP[7:0]	RW	8'b0	Value of the repetition counter When the preload function is enabled, these bits allow the user to set the update rate of the compare register (i.e., the periodic transfer from the preload register to the active register); when the update interrupt is enabled, it also affects the rate at which update interrupts are generated. Each time the down-counter REP_CNT reaches 0, an update event is generated, and the counter REP_CNT restarts counting from the REP value. Since REP_CNT only reloads the REP value when a repetitive update event (UIF) occurs, a new value written to the TIM1_RCR register will only take effect at the next repetitive update event. This means that in PWM mode, (REP+1) corresponds to: - the number of PWM periods in edge-aligned mode - In center-aligned mode, the number of PWM half-cycles;

18.6.14. TIM1 Capture/Compare Register 1 (TIM1_CCR1)

Offset address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1 [15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR1 [15:0]	RW	16'b0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 contains the value to be loaded into the current capture/compare register 1 (preload value). If the preload feature is not selected in the TIM1_CCMR1 register (OC1PE bit), this value

				takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the counter TIM1_CNT and outputs a signal on the OC1 port. If channel CC1 is configured as input: CCR1 is the counter value at the time of the last input capture 1 event (IC1). At this time, the TIM1_CCR1 register can only be read and cannot be programmed.

18.6.15. TIM1 capture/compare register 2 (TIM1_CCR2)

Offset Address: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2 [15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR2 [15:0]	RW	16'b0	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 contains the value to be loaded into the current capture/compare register 2 (preload value). If the preload feature is not selected in the TIM1_CCMR1 register (OC2PE bit), this value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the TIM1_CNT counter, and outputs the signal on the OC2 port. If channel CC2 is configured as input: CCR2 is the counter value at the time of the last input capture 2 event (IC2). At this time, only reading the TIM1_CCR2 register is possible; programming it is not allowed.

18.6.16. TIM1 capture/compare register 3 (TIM1_CCR3)

Offset Address: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3 [15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR3 [15:0]	RW	16'b0	Capture/Compare 3 value If channel CC3 is configured as output: CCR3 contains the value to be loaded into the current capture/compare register 3 (preload value). If the preload feature is not selected in the TIM1_CCMR2 register (OC3PE bit), this value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the counter TIM1_CNT and outputs a signal on the OC3 port. If channel CC3 is configured as input:

				CCR3 is the counter value at the time of the last input capture 3 event (IC3). At this time, the TIM1_CCR3 register can only be read and cannot be programmed.

18.6.17. TIM1 capture/compare register 4 (TIM1_CCR4)

Offset address: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4 [15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR4 [15:0]	RW	16'b0	Capture/Compare 4 value If channel CC4 is configured as output: CCR4 contains the value to be loaded into the current capture/compare register 4 (preload value). If the preload feature is not selected in the TIM1_CCMR2 register (OC4PE bit), this value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the counter TIM1_CNT and outputs a signal on the OC4 port. If channel CC4 is configured as input: CCR4 is the counter value at the time of the last input capture 4 event (IC4). At this time, only the TIM1_CCR4 register can be read, and it cannot be programmed.

18.6.18. TIM1 break/dead-time register (TIM1_BDTR)

Offset Address: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as one of the break inputs is active. It is set by software or automatically depending on the AOE bit. It is acting only on the channels which are configured in output. 0: Disable OC and OCN outputs or force them to idle state, depending on the OSSI bit; 1: Enable OC and OCN outputs if the corresponding enable bits (CCxE, CCxNE bits in the TIM1_CCER register) are set.
14	AOE	RW	0	Automatic output enable 0: MOE can be set only by software; 1: MOE can be set by software or automatically at the next update event (if none of the break inputs is active). Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
13	BKP	RW	0	Break polarity

				0: Break input BRK is active low; 1: Break input BRK is active high Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
12	BKE	RW	0	Break enable 0: Break function disabled 1: Enable the break input. Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
11	OSSR	RW	0	Off-state selection for Run mode This bit is used when MOE=1 on channels having a complementary output which are configured as outputs. OSSR bit is not implemented if no complementary output is implemented in the timer. See OC/OCN enable description for more details (TIM1 capture/compare enable register (TIM1_CCER)). 0: In inactive state, disable OC/OCN outputs (OC/OCN enable output signal = 0); 1: In inactive state, enable OC/OCN outputs and output the inactive level once CCxE=1 or CCxNE=1. Note: This bit can not be modified as long as LOCK level 2 has been set (LOCK bits in TIM1_BDTR register).
10	OSSI	RW	0	Off-state selection for Idle mode This bit is used when MOE=0 and on channels configured as outputs. See OC/OCN enable description for more details (TIM1 capture/compare enable register (TIM1_CCER)). 0: In inactive state, disable OC/OCN outputs (OC/OCN enable output signal = 0); 1: In inactive state, OC/OCN first outputs its inactive level once CCxE=1 or CCxNE=1. Note: This bit can not be modified as long as LOCK level 2 has been set (LOCK bits in TIM1_BDTR register).
9:8	LOCK[1:0]	RW	2'b0	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: Lock level 1, cannot write to the DTG/BKE/BKP/AOE bits of the TIM1_BDTR register or the OISx/OISxN bits of the TIM1_CR2 register; 10: Lock level 2, cannot write to the bits in lock level 1, nor to the CC polarity bits (once the related channel is set as output via the CCxS bit, the CCxP/CCxNP bits of the TIM1_CCER register) and the OSSR/OSSI bits; 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIM1_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. Note: The LOCK bit can only be written once after a system reset. Once written to the TIM1_BDTR register, its content is frozen until the next system reset.
7:0	DTG[7:0]	RW	8'b0	Dead-time generator setup This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. $DTG[7:5]=0xx \Rightarrow DT=DTG[7:0] \times Tdtg, \quad Tdtg = T_{DTS};$ $DTG[7:5]=10x \Rightarrow DT=(64+DTG[5:0]) \times Tdtg, \text{ where } Tdtg = 2 \times T_{DTS};$ $DTG[7:5]=110 \Rightarrow DT=(32+DTG[4:0]) \times Tdtg, \text{ where } Tdtg = 8 \times T_{DTS};$

				DTG[7:5]=111 => DT=(32+DTG[4:0]) ×Tdtg, where Tdtg = 16 ×T_{DTS}; Example if T_{DTS}=125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps Note: Once the LOCK level (LOCK bits in TIM1_BDTR register) is set to 1, 2, or 3, these bits cannot be modified.

18.6.19. TIM1 input selection register (TIM1_TISEL)

Offset address: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TI4SEL [3:0]				Res.	Res.	Res.	Res.	TI3SEL [3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	TI2SEL [3:0]				Res.	Res.	Res.	Res.	TI1SEL [3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:24	TI4SEL [3:0]	RW	4'b0	TI4 input selection 0000: TIM1_CH4 0001: COMP1_OUT 0010 COMP2_OUT Other: Reserved
23:20	Reserved	-	-	Reserved
19:16	TI3SEL [3:0]	RW	4'b0	TI3 input selection 0000: TIM1_CH3 0001: COMP1_OUT 0010 COMP2_OUT Other: Reserved
15:12	Reserved	-	-	Reserved
11:8	TI2SEL [3:0]	RW	4'b0	TI2 input selection 0000: TIM1_CH2 0001: COMP1_OUT 0010 COMP2_OUT Other: Reserved
7:4	Reserved	-	-	Reserved
3:0	TI1SEL [3:0]	RW	4'b0	TI1 input selection 0000: TIM1_CH1 0001: COMP1_OUT 0010 COMP2_OUT Other: Reserved

18.6.20. TIM1 alternate function option register 1 (TIM1_AF1)

Offset address: 0x60

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETRSEL[3:2]	
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL[1:0]	Res.	Res.	BKCOMP2_P	BKCOMP1_P	BKIN_P	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BKCOMP2_E	BKCOMP1_E	BKIN_E
RW	RW		RW	RW	RW								RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	Reserved
17:14	ETRSEL[3:0]	RW	4'b0	External trigger source selection

				This bit segment is used to select an ETR input source 0000: TIM1_ETR 0001: COMP1_OUT 0010: COMP2_OUT 0011: Reserved 0100: Reserved 0101: ADC_AWD Others: Reserved Note: Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 1, this bit cannot be modified.
13:12	Reserved	-	-	Reserved
11	BKCMP2P	RW	0	Brake input polarity control from comparator 2. This bit selects the brake input polarity from comparator 2 and must be configured together with the BKP polarity bit. 0: Comparator 2 brake input polarity not inverted (active low when BKP=0, active high when BKP=1) 1: Comparator 2 brake input polarity inverted (active high when BKP=0, active low when BKP=1) Note: Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 1, this bit cannot be modified.
10	BKCMP1P	RW	0	Brake input polarity control from comparator 1. This bit selects the brake input polarity from comparator 1 and must be configured together with the BKP polarity bit. 0: Comparator 1 brake input polarity not inverted (active low when BKP=0, active high when BKP=1) 1: Comparator 1 brake input polarity inverted (active high when BKP=0, active low when BKP=1) Note: Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 1, this bit cannot be modified.
9	BKINP	RW	0	BKIN input polarity. This bit selects the alternate function of the BKIN input polarity and must be configured at the same time as the BKP polarity bit 0: BKIN input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: BKIN input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIM1_BDTR register) is set to 1, this bit cannot be modified.
8:3	Reserved	-	-	Reserved
2	BKCMP2E	RW	0	Comparator 2 brake input enable. Comparator 2 brake input is ORed with other brake sources as the brake input. 0: Disable comparator 2 brake input 1: Enable comparator 2 brake input Note: Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 1, this bit cannot be modified.
1	BKCMP1E	RW	0	Comparator 1 brake input enable. Comparator 1 brake input is ORed with other brake sources as the brake input. 0: Disable comparator 1 brake input 1: Enable comparator 1 brake input Note: Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 1, this bit cannot be modified.
0	BKINE	RW	1	BKIN brake input enable. Comparator 1 brake BKIN input is ORed with other brake sources as the brake input. 0: BKIN input off 1: BKIN input on

				Note: Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 1, this bit cannot be modified.

18.6.21. TIM1 alternate function option register 2 (TIM1_AF2)

Offset address: 0x64

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18:16	OCRSEL[2:0]	RW	3'b0	OCREF reset source selection This bit field is used to select the OCREF_CLR input source. 000: COMP1_OUT 001 COMP2_OUT Others: Reserved Note: Once the LOCK level (LOCK bit in the TIM1_BDTR register) is set to 1, this bit cannot be modified.
15:0	Reserved	-	-	Reserved

18.6.22. TIM1 DMA control register (TIM1_DCR)

Offset address: 0x3DC

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12:8	DBL[4:0]	RW	5'b0	DMA burst length These bits define the transfer length of DMA in continuous mode (when reading or writing to the address of the TIM1_DMAR register, the timer performs a continuous transfer). 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers 10001: 18 transfers
7:5	Reserved	-	-	Reserved
4:0	DBA[4:0]	RW	0 0000	DMA base address These bits define the base address of DMA in continuous mode (when reading or writing to the address of the TIM1_DMAR register). DBA is defined as the offset from the address of the TIM1_CR1 register: 00000: TIM1_CR1 00001: TIM1_CR2 00010: TIM1_SMCR

18.6.23. TIM1 DMA continuous transfer register (TIM1_DMAR)

Offset address: 0x3E0

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[15:0]	RW	0	DMA register for burst accesses Reading or writing to the TIM1_DMAR register results in access to the registers at the following address: TIM1_CR1 address + (DBA + DMA pointer) x4, where: TIM1_CR1 address is the address of the control register 1; "DBA" is the base address defined in the TIM1_DCR register; "DMA pointer" is the offset automatically controlled by DMA, which depends on the DBL defined in the TIM1_DCR register.

Note: When using the DMA continuous transfer function, the value of the CNDTR register of the corresponding channel in DMA must correspond to the DBL value in the TIM1_DCR register. Otherwise, the DMA continuous transfer function cannot be used normally.

19. General-purpose timers (TIM2/TIM3)

19.1. TIM2/TIM3 introduction

The TIM2 general-purpose timer consists of a 32-bit auto-reload counter driven by a programmable prescaler.

The TIM3 general-purpose timer consists of a 16-bit auto-reload counter driven by a programmable prescaler.

Except for the difference in counter width, all other functions of TIM2 and TIM3 are identical. The functional description uses TIM2 as an example.

It is suitable for various applications, including measuring the pulse length of input signals (input capture) or generating output waveforms (output compare and PWM).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.

19.2. TIM2/TIM3 main features

The general-purpose TIM2/TIM3 timer functions include:

- 32-bit (TIM2)/16-bit (TIM3) up, down, and up/down auto-reload counter
- 16-bit programmable prescaler used to divide the counter clock frequency by any factor between 1 and 65536
- 4 independent channels for
 - Input capture
 - Output compare
 - PWM generation (edge-aligned or center-aligned mode)
 - One-pulse mode output
- Synchronization circuit for controlling the timer with external signals and enabling interconnection of multiple timers
- Interrupt/DMA generation on the following events:
 - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
 - Trigger event (counter start, stop, initialization or count by internal/external trigger)
 - Input capture
 - Output compare
- Trigger input as external clock or cycle-by-cycle current management

19.3.2. Time-base unit

The main block of the programmable general-purpose timer consists of a 16-bit/32-bit counter and its associated auto-reload register. The counter can count up, down or both up and down. The counter clock can be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. An update event is generated when the counter reaches the overflow value (or underflow value in downcounting) and the UDIS bit in the TIMx_CR1 register is 0. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in the TIMx_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR1 register.

19.3.2.1. Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly:

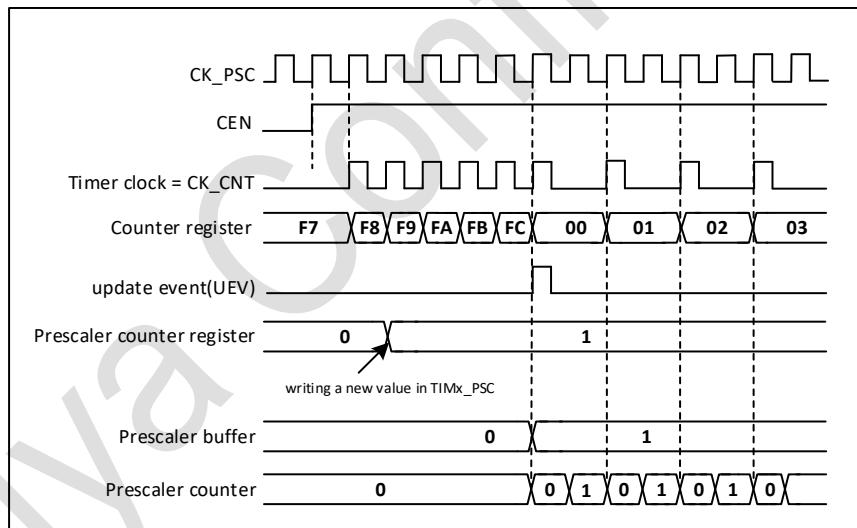


Figure 19-2 Counter timing diagram when prescaler division changes from 1 to 2

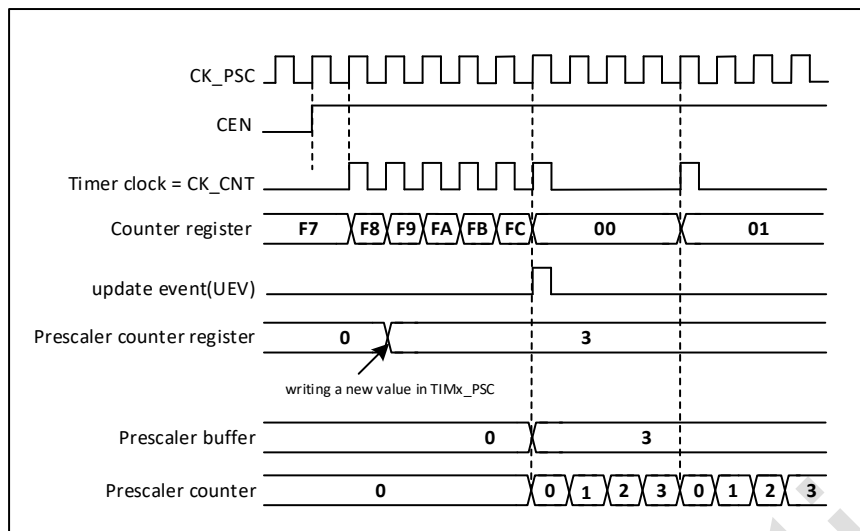


Figure 19-3 Counter timing diagram when prescaler division changes from 1 to 4

19.3.3. Counting modes

19.3.3.1. Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (TIMx_ARR register), then restarts from 0 and generates a counter overflow event.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

Setting the UDIS bit in the TIMx_CR1 register disables the update event (UEV); this also prevents updating the shadow registers when writing new values to the preload registers. Then no update event occurs until the UDIS bit has been written to 0. Even so, the counter is still cleared to '0' when an update event should occur, and the prescaler count is also cleared (though the prescaler value remains unchanged). Additionally, if the URS bit (update request selection) in the TIMx_CR1 register is set, setting the UG bit generates an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all registers are updated, and the update flag (UIF bit in the TIMx_SR register) is set to 1 (depending on the URS bit).

- The auto-load shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

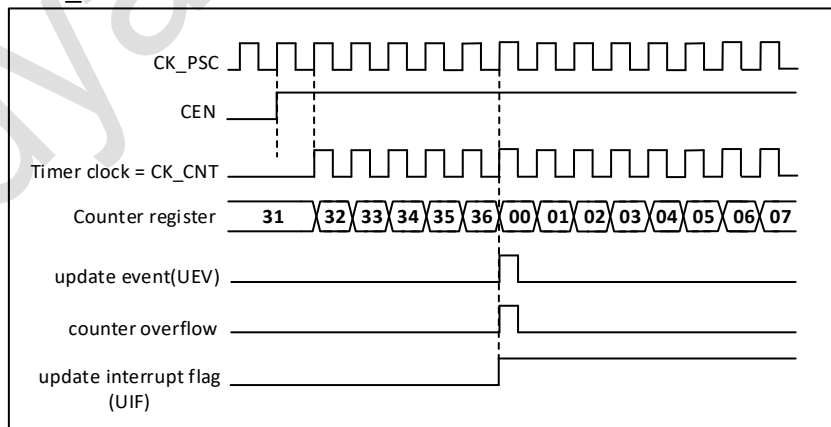


Figure 19-4 Counter timing diagram, internal clock divided by 1

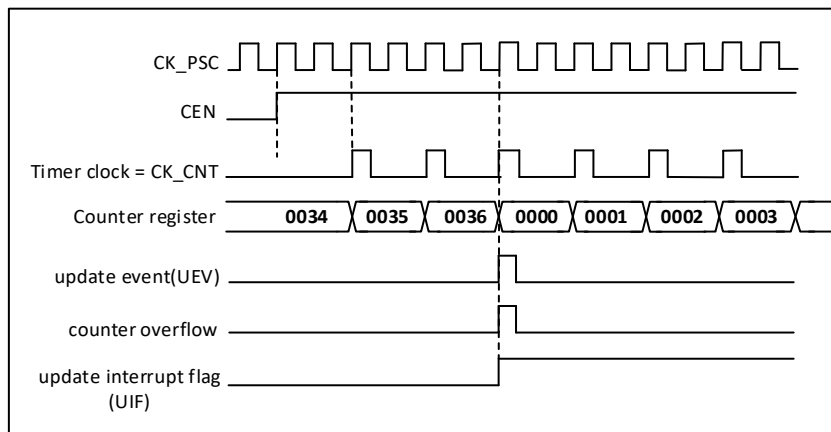


Figure 19-5 Counter timing diagram, internal clock divided by 2

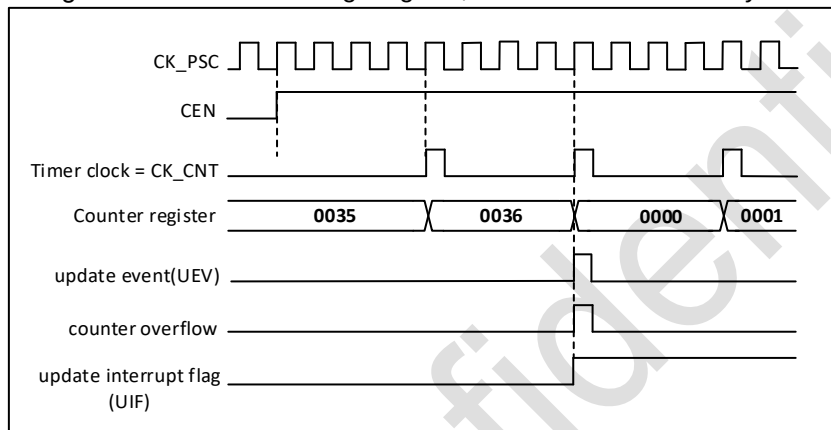


Figure 19-6 Counter timing diagram, internal clock divided by 4

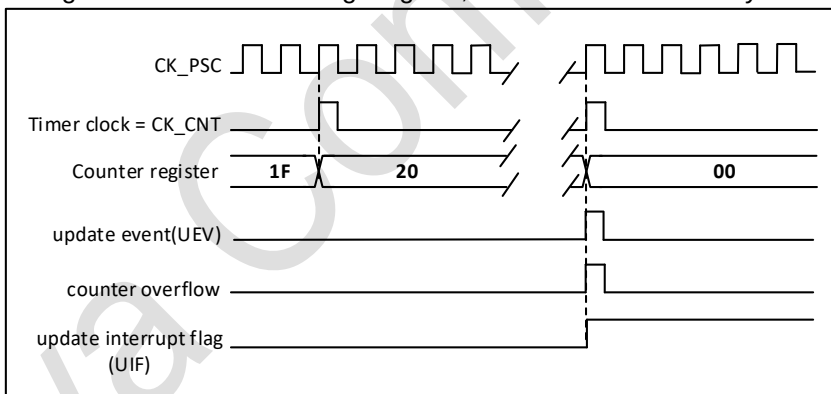


Figure 19-7 Counter timing diagram, internal clock divided by N

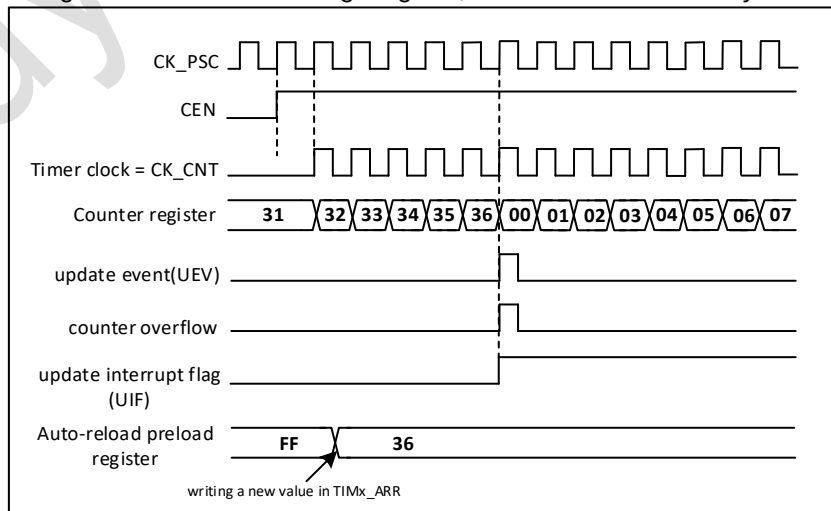


Figure 19-8 Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

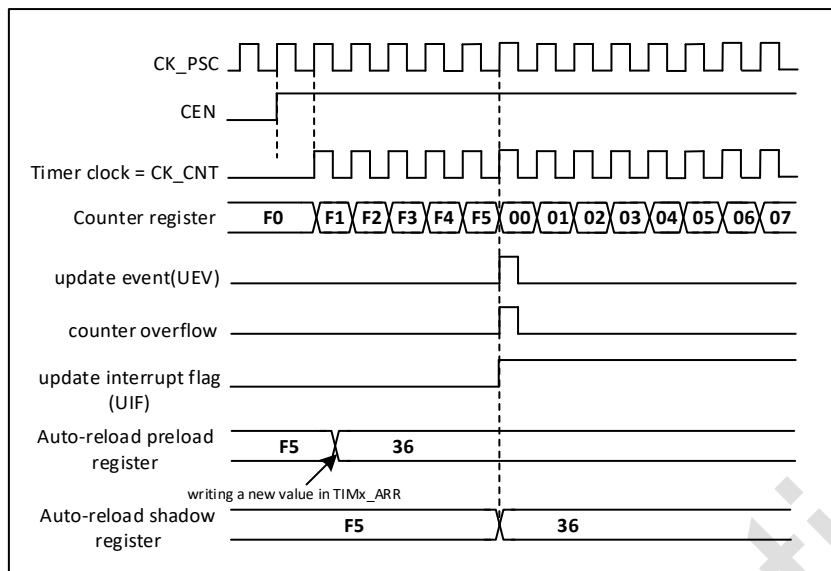


Figure 19-9 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

19.3.3.2. Downcounting mode

In down-counting mode, the counter counts down from the auto-reload value (content of the TIMx_ARR register) to 0, then restarts counting from the auto-reload value and generates a counter underflow event.

An update event is generated each time a counter underflow occurs, or by setting the UG bit in the TIMx_EGR register to 1 (via software or using the slave mode controller).

Setting the UDIS bit in the TIMx_CR1 register disables the update event (UEV); this prevents updating the shadow registers when writing new values to the preload registers. Then no update event occurs until the UDIS bit has been written to 0. Even so, the counter is still cleared to '0' when an update event should occur, and the prescaler count is also cleared (though the prescaler value remains unchanged).

Additionally, if the URS bit (update request selection) in the TIMx_CR1 register is set, setting the UG bit generates an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all registers are updated, and the update flag (UIF bit in the TIMx_SR register) is set to 1 (depending on the URS bit).

- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

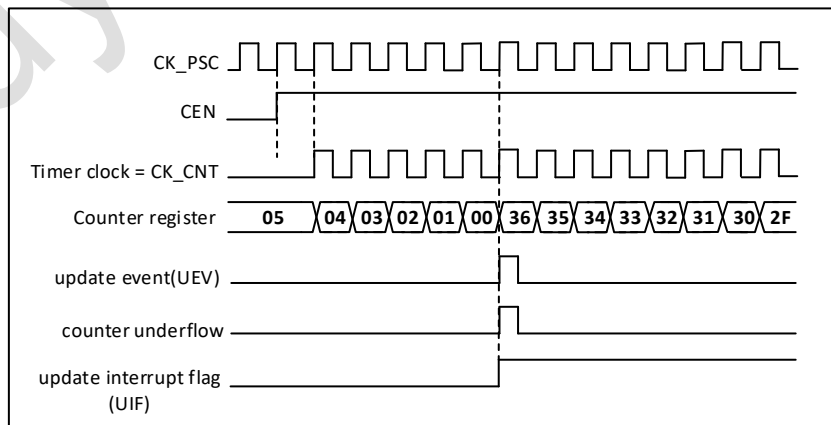


Figure 19-10 Counter timing diagram, internal clock divided by 1

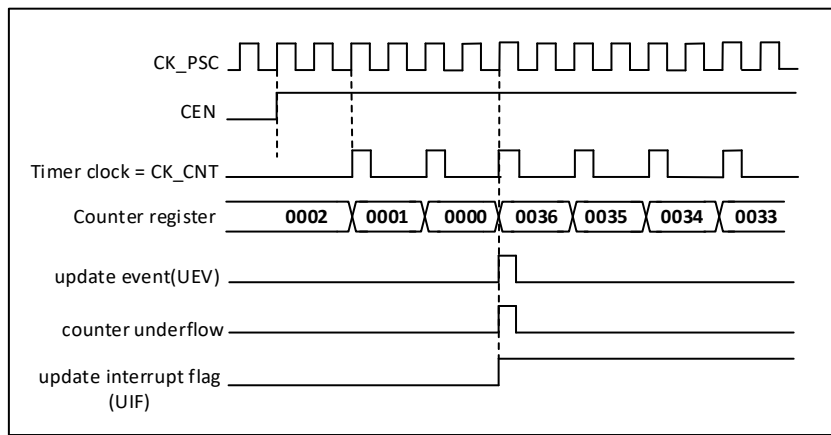


Figure 19-11 Counter timing diagram, internal clock divided by 2

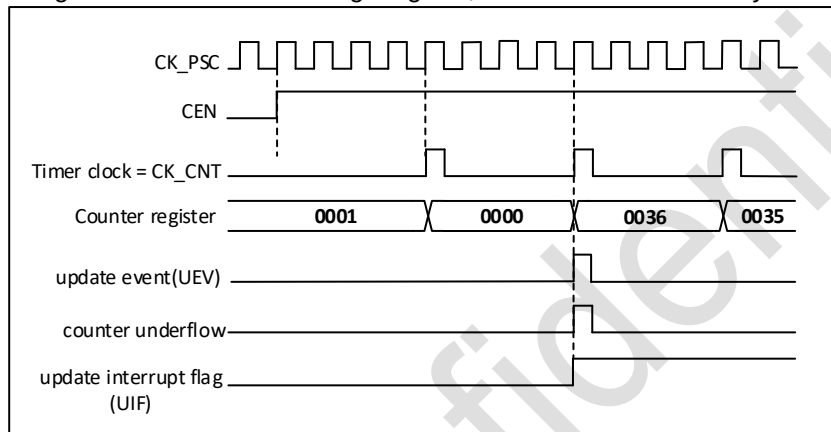


Figure 19-12 Counter timing diagram, internal clock divided by 4

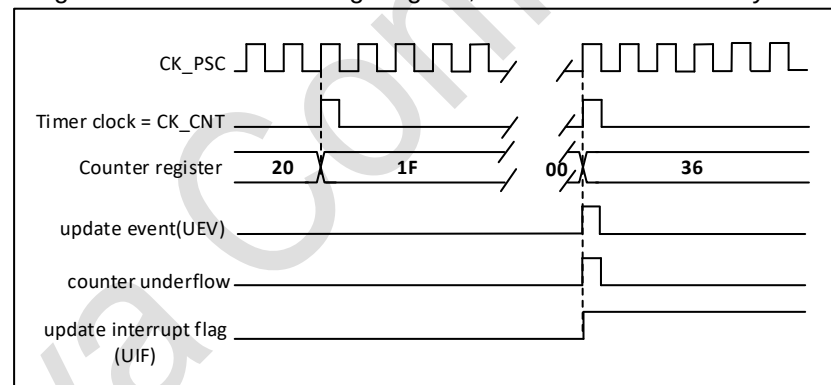


Figure 19-13 Counter timing diagram, internal clock divided by N

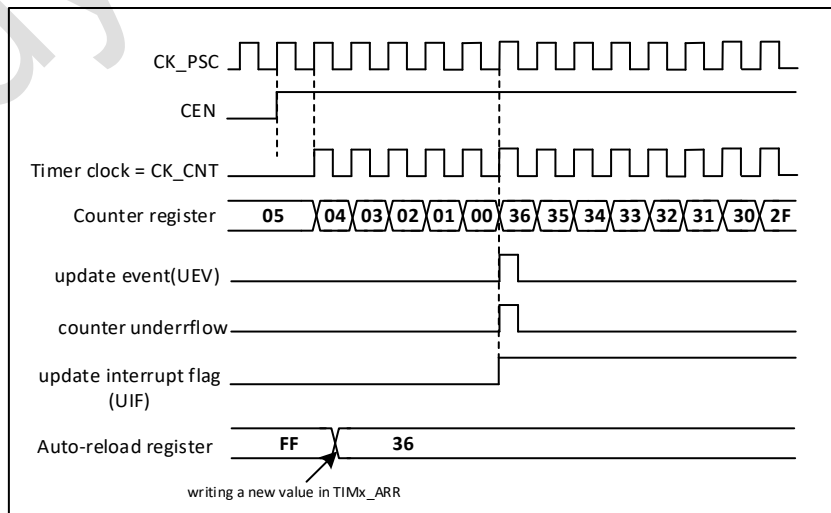


Figure 19-14 Counter timing diagram, update event when repeat counter is not used

19.3.3.3. Center-aligned mode

In center-aligned mode, the counter counts from 0 up to the auto-reload value (TIMx_ARR register) -1, generates a counter overflow event, then counts down to 1 and generates a counter underflow event, then restarts counting from 0.

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are not equal to '00'. When a channel is configured in output mode, its output compare interrupt flag will be set to 1 in the following modes: counter counting down (center-aligned mode 1, CMS = "01"), counter counting up (center-aligned mode 2, CMS = "10"), and counter counting up/down (center-aligned mode 3, CMS = "11").

In this mode, the DIR direction bit in the TIMx_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter.

The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event. In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter continues to count up or down based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag (thus no interrupt and DMA requests are sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all registers are updated and the update flag (UIF bit in TIM1_SR register) is set to 1 (depending on the URS bit).

- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The current auto-reload register is updated with the preload value (TIMx_ARR)

Note: If an update occurs due to counter overflow, the auto-reload will be updated before the counter is reloaded, so the next cycle will have the expected value (the counter is loaded with the new value).

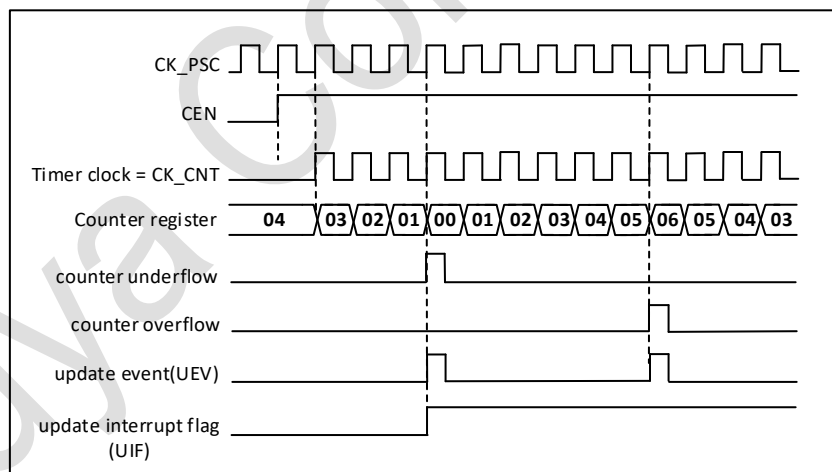


Figure 19-15 Counter timing diagram, internal clock divided by 1, TIM1_ARR = 0x6

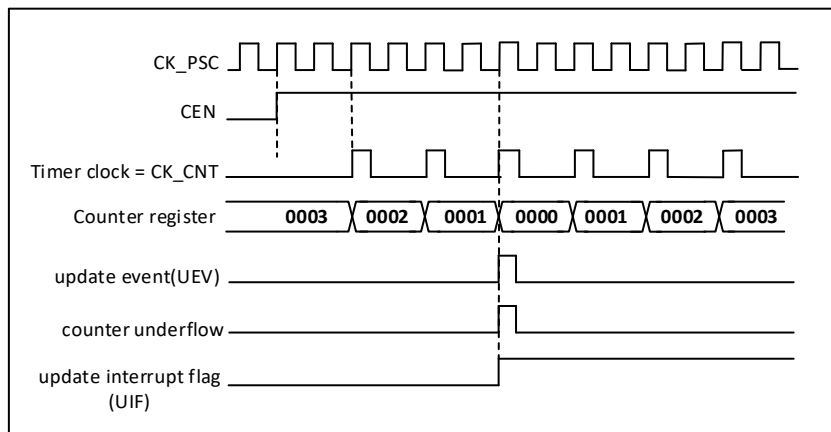


Figure 19-16 Counter timing diagram, internal clock divided by 2

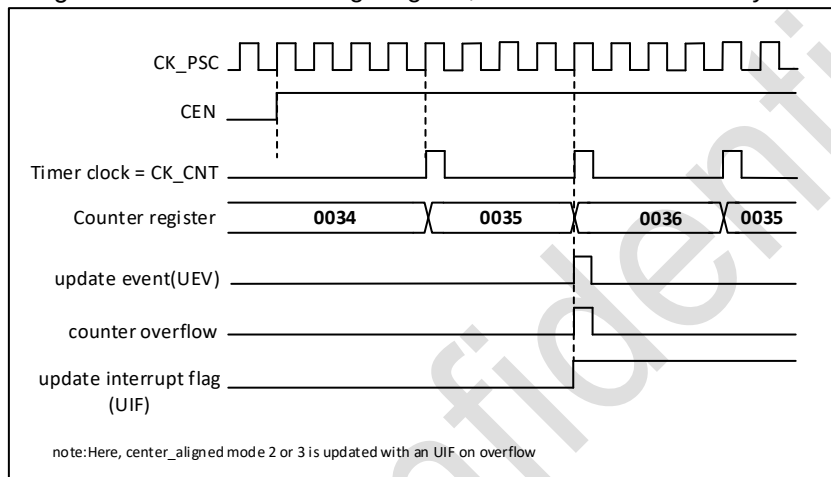


Figure 19-17 Counter timing diagram, internal clock divided by 4, TIM1_ARR=0x36

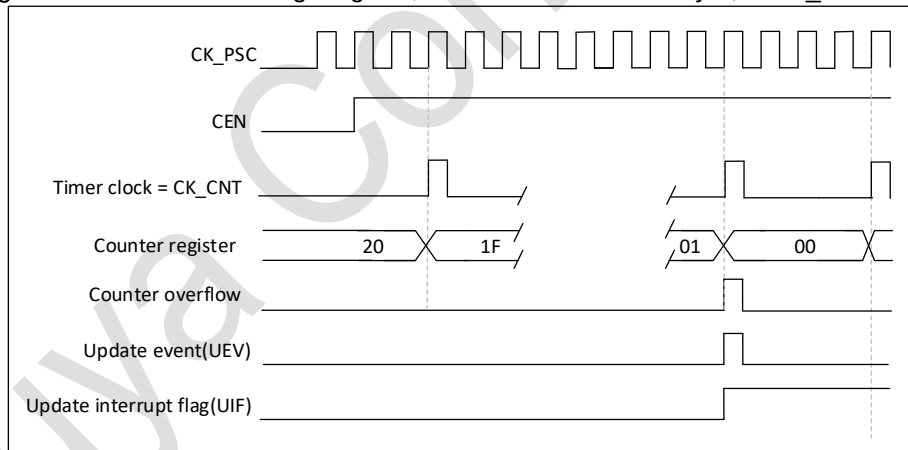


Figure 19-18 Counter timing diagram, internal clock divided by N

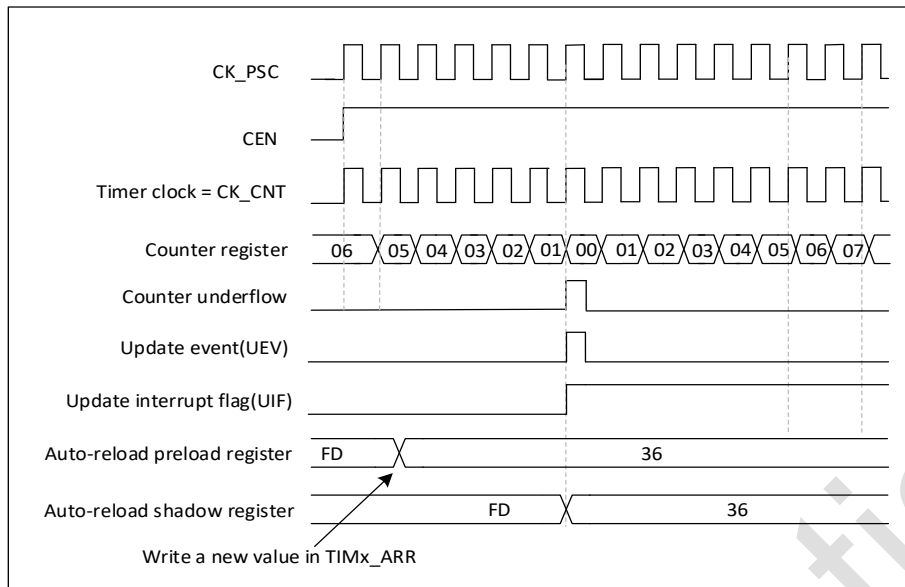


Figure 19-19 Counter timing diagram, update event with ARPE=1 (counter underflow)

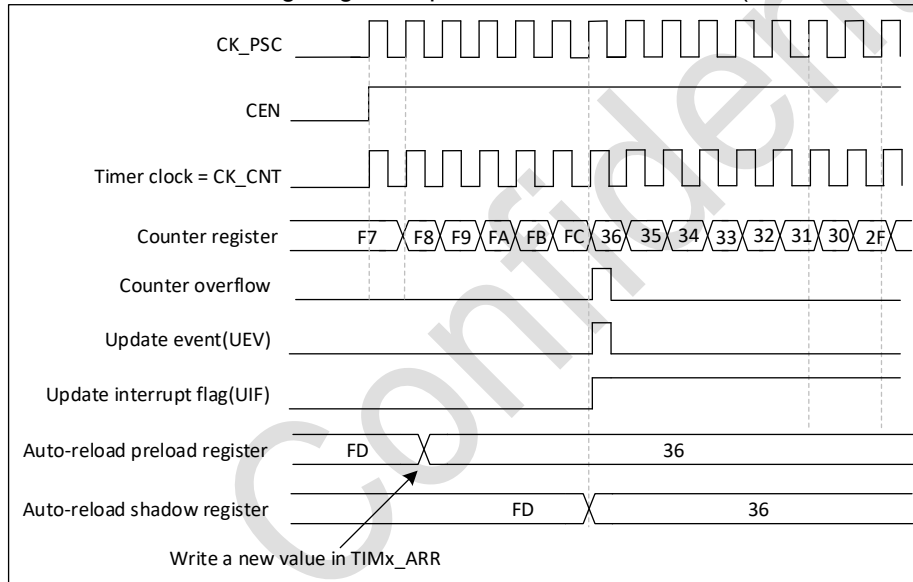


Figure 19-20 Counter timing diagram, update event when ARPE=1 (counter overflow)

19.3.4. Clock sources

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode 1: External input pin (TIx)
- External clock mode2: external trigger input ETR
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer. For example, the user can configure Timer 1 to act as a prescaler for Timer 2.

19.3.4.1. Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS=000), the CEN, DIR (TIMx_CR1 register) and UG bits (TIMx_EGR register) are the actual control bits and can only be modified by software (except UG, which remains auto-cleared). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

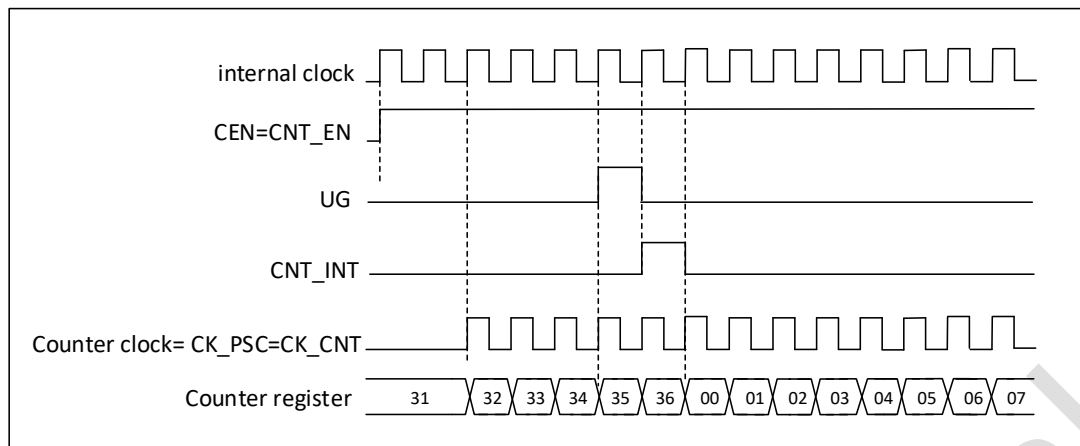


Figure 19-21 Control circuit in normal mode, internal clock divided by 1

19.3.4.2. External clock source mode 1

This mode is selected when SMS=111 in the TIMx_SMCR register. The counter can count at each rising or falling edge on a selected input.

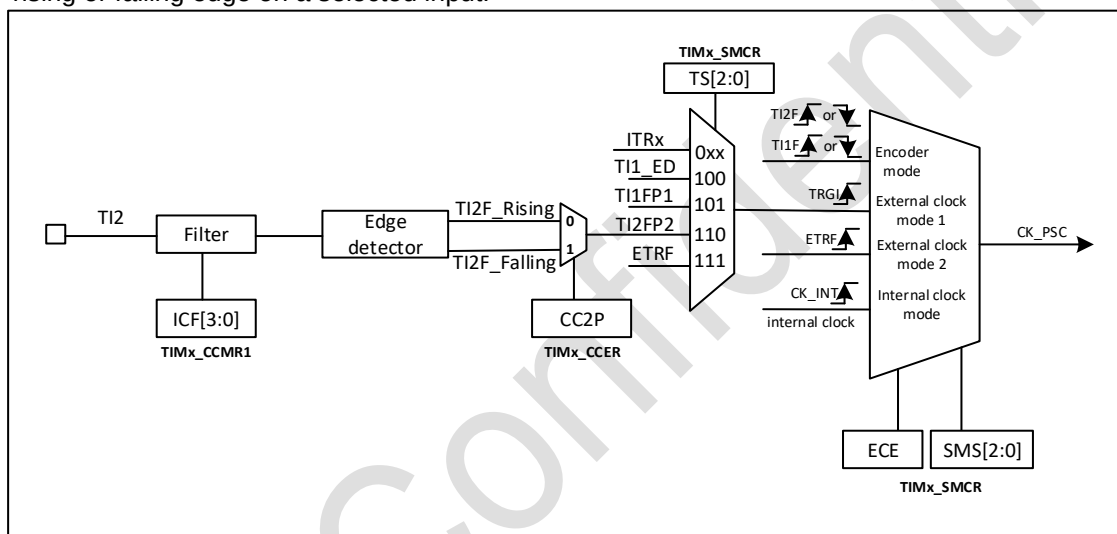


Figure 19-22 Example of TI2 External Clock Connection

For example, to configure the upcounter to count in response to a rising edge on the TI2 input, use the following procedure:

1. Select the correct TI2x source (internal or external) by configuring the TI2SEL[3:0] bits in the TIMx_TISEL register.
2. Configure TIMx_CCMR1 register CC2S=01 to set channel 2 to detect the rising edge of TI2 input
3. Configure IC2F[3:0] in TIMx_CCMR1 register to select input filter bandwidth (if no filter is needed, keep IC2F=0000)
4. Configure CC2P=0 in TIMx_CCER register to select rising edge polarity
5. Configure SMS=111 in TIMx_SMCR register to select timer external clock mode 1
6. Configure TS=00110 in TIMx_SMCR register to select TI2 as trigger input source
7. Set CEN=1 in TIMx_CR1 register to start the counter

Note: Since the capture prescaler is not used for trigger operation, users don't need to configure it.

When a rising edge occurs on TI2, the counter increments once and the TIF flag is set to 1.

The delay between the rising edge of TI2 and the actual counter clock is caused by the resynchronization circuit of the TI2 input.

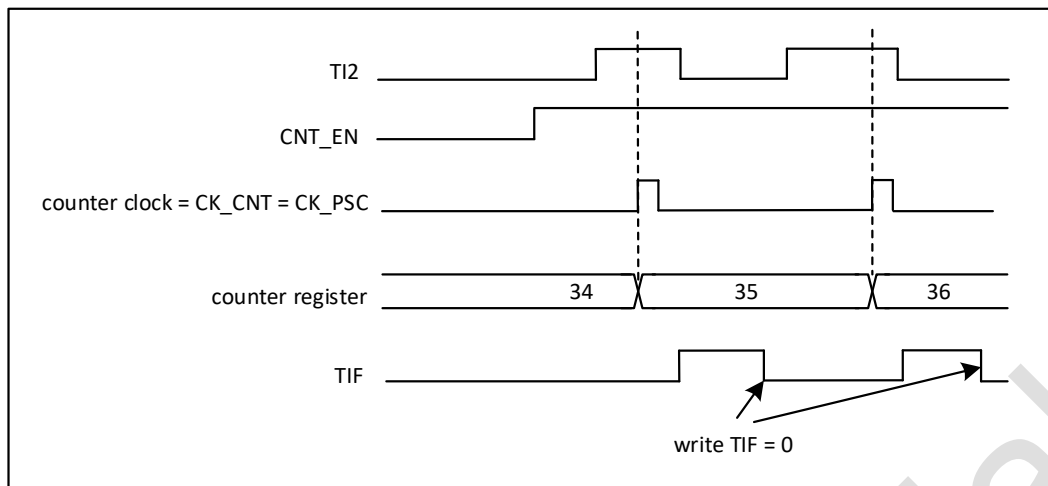


Figure 19-23 Timing of External Clock Mode 1 Control Circuit

19.3.4.3. External clock source mode 2

This mode is selected by writing ECE=1 in the TIM1_SMCR register. The counter can count at each rising or falling edge on the external trigger input ETR.

The following figure is a block diagram of the external trigger input:

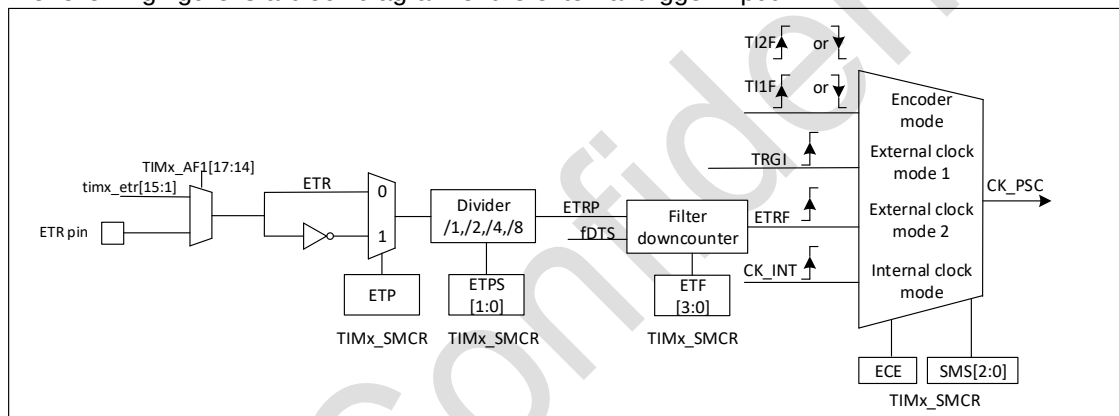


Figure 19-24 External Trigger Input Module

For example, to configure the upcounter to count each 2 rising edges on ETR, use the following procedure:

1. Since no filter is needed in this example, write ETF[3:0]=0000 in the TIMx_SMCR register.
2. Set the prescaler by writing ETSP[1:0]=01 in the TIMx_SMCR register.
3. Select rising edge detection for the ETR pin by writing ETP=0 in the TIMx_SMCR register.
4. Enable external clock mode 2 by writing ECE=1 in the TIMx_SMCR register.
5. Enable the counter by writing CEN=1 in the TIM1x_CR1 register.

The counter counts once each 2 ETR rising edges.

The delay between the rising edge on ETR and the actual clock of the counter is due to the resynchronization circuit on the ETRP signal.

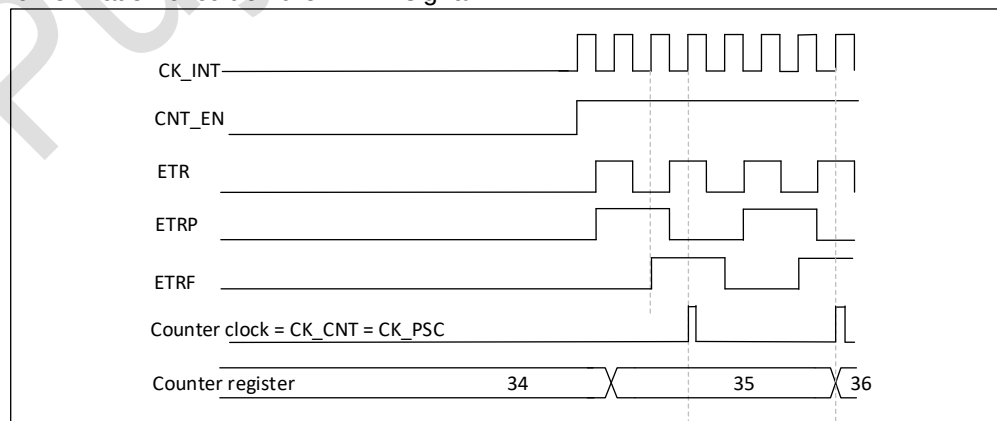


Figure 19-25 Timing diagram of external clock mode 2 control circuit

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

19.3.6. Input capture mode:

In Input capture mode, the capture/compare registers are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIMx_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when you write it to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the correct TI1x source (internal or external) using the TI1SEL[3:0] bits in the TIMx_TISEL register.
- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMx_CCR1 register becomes read-only.
- Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register)). Let's imagine that, when toggling, the input signal is not stable during at most 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at f_{DTS} frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the active transition edge for the TI1 channel by writing CC1P=0 (rising edge) (and CC1NP=0) in the TIMx_CCER register.
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PSC bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). When at least 2 consecutive captures occur, but CC1IF has never been cleared, CC1OF is also set to 1.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

19.3.7. PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input.
- These two ICx signals are edge-sensitive but have opposite polarities.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.

For example, you can measure the period (in TIMx_CCR1 register) and the duty cycle (in TIMx_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

- Select the proper TI1x source (internal or external) with the TI1SEL[3:0] bits in the TIMx_TISEL register.
- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).

- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2Pbit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to '1' in the TIMx_CCER register.

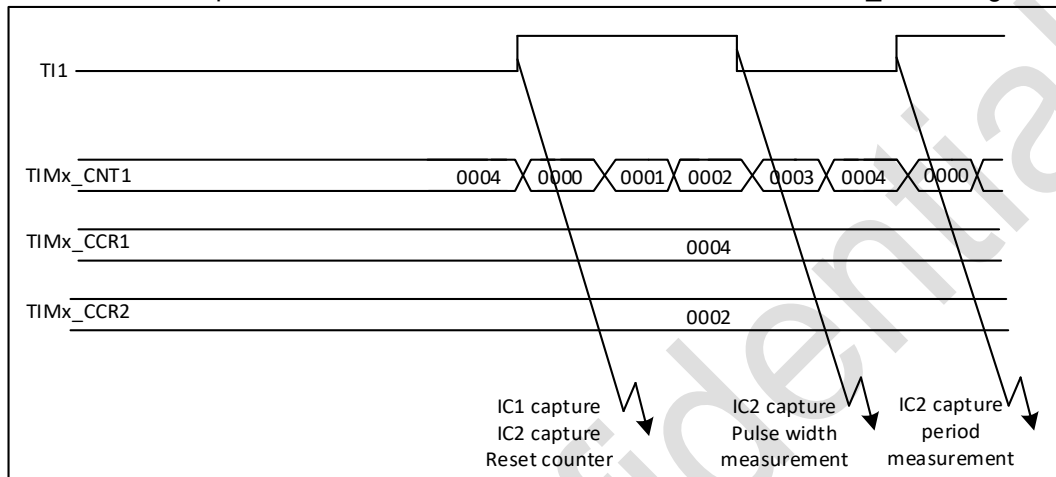


Figure 19-29 PWM input mode timing

19.3.8. Forced Output Mode

In output mode (CCxS=00 in the TIMx_CCMRx register), the output compare signals (OCxREF and the corresponding OCx) can be directly forced to active or inactive by software, independent of the comparison result between the output compare register and the counter. To force an output compare signal (OCxREF/OCx) to its active level, you just need to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCxREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level. The OCxREF signal can be forced low by writing the OCxM bits to 0100 in the TIMx_CCMRx register.

Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

19.3.9. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed. When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCxM=0000), be set active (OCxM=0001), be set inactive (OCxM=0010) or can toggle (OCxM=0011) on match.
- Sets a flag in the interrupt status register (CCXIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).
- If the corresponding DMA enable bit is set (CCxDE bit in the TIMx_DIER register, with the CCDS bit in the TIMx_CR2 register selecting the DMA request function), a DMA request is generated.

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing accuracy can reach one counting cycle of the counter. The output compare mode can also be used to output a single pulse (in one-pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:
 - Set OCxM=0011 to toggle the OCx output pin when the counter matches CCRx,
 - Write OCxPE = 0 to disable preload register
 - Write CCxP = 0 to select active high polarity
 - Write CCxE = 1 to enable the output
5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated at any time via software to control the output waveform, provided the preload register is not used (OCxPE=0'; otherwise, the shadow register of TIMx_CCRx can only be updated upon the next update event). An example is given in the figure below.

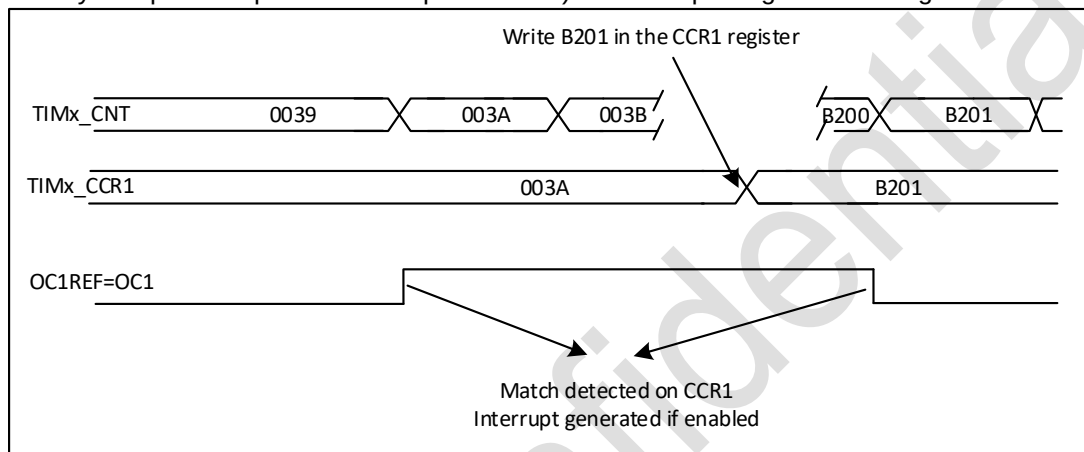


Figure 19-30 Output Compare Mode, Toggling OC1

19.3.10. PWM mode

PWM mode allows generation of a signal whose frequency is determined by the TIMx_ARR register and duty cycle by the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bits in the TIMx_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and finally, the ARPE bit in the TIMx_CR1 register must be set to enable the auto-reload preload register (in upcounting or center-aligned mode).

Since the preload register can only be transferred to the shadow register when an update event occurs, all registers must be initialized by setting the UG bit in the TIMx_EGR register before the counter starts counting.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. The output enable of OCx is controlled by a combination of CCxE, CCxNE, MOE, OSSI, and OSSR bits (TIMx_CCER and TIMx_BDTR registers). Refer to the TIMx_CCER register description for more details.

In PWM mode (mode 1 or mode 2), TIMx_CNT and TIMx_CCRx are continuously compared to determine whether $TIMx\_CCRx \leq TIMx\_CNT$ or $TIMx\_CNT \leq TIMx\_CCRx$ (depending on the counter's counting direction).

However, to satisfy the OCREF_CLR function (OCREF can be cleared by an external event generated by the ETR signal until the next PWM cycle), OCREF_CLR can only be provided in the following ways:

1. When the comparison result changes.
2. By setting OcxM in the TIMx_CCMRx register, the output compare mode switches from "frozen" configuration (OCxM=000) to any PWM mode (OCxM="110" or "111").

This is a software-forced switch to PWM mode while the timer is running.

Depending on the state of the CMS bit in the TIMx_CR1 register, the timer can generate edge-aligned PWM signals or center-aligned PWM signals.

19.3.10.1. PWM edge-aligned mode

Upcounting configuration

Upcounting is performed when the DIR bit in the TIMx_CR1 register is low. In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

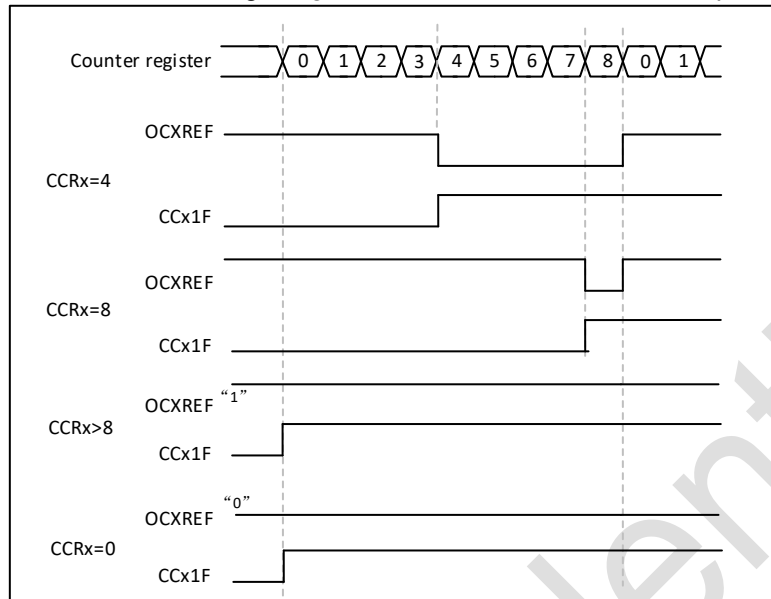


Figure 19-31 Edge-aligned PWM output waveform (ARR=8)

Downcounting configuration

Downcounting is performed when the DIR bit in the TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxREF is low when TIMx_CNT > TIMx_CCRx; otherwise, it is high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at '1'. 0% PWM is not possible in this mode.

19.3.10.2. PWM center-aligned mode

When the CMS bit in the TIMx_CR1 register is not '00', it is center-aligned mode (all other configurations have the same effect on the OCxREF/OCx signal). Depending on the CMS bit settings, the compare flag can be set to 1 during up-counting, down-counting, or both up and down-counting. The count direction bit (DIR) in the TIMx_CR1 register is updated by hardware and cannot be modified by software.

The figure below shows some examples of center-aligned PWM waveforms.

- TIMx_ARR = 8
- PWM mode is the PWM mode 1
- When CMS=01 in the TIMx_CR1 register, in center-aligned mode, the compare flag is set when the counter counts down.

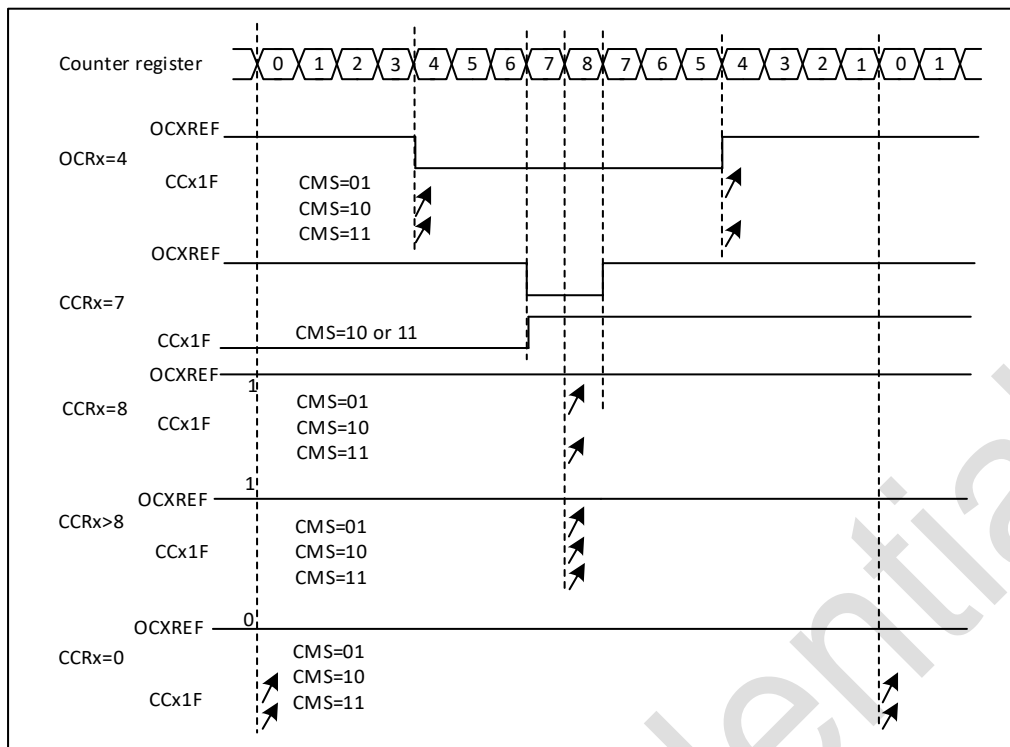


Figure 19-32 Center-aligned mode waveform (ARR=8)

Hints on using center-aligned mode:

- When entering center-aligned mode, the current up/down counting configuration is used; this means whether the counter counts up or down depends on the current value of the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- It is not recommended to rewrite the counter while running in center-aligned mode, as this may produce unpredictable results. In particular:
 - If the value written to the counter is greater than the auto-reload value (TIMx_CNT > TIMx_ARR), the direction is not updated. For example, if the counter is up-counting, it will continue to up-count.
 - If 0 or the TIMx_ARR value is written to the counter, the count direction is updated, but no update event (UEV) is generated.
- The safest way to use center-aligned mode is to generate a software update (set the UG bit in the TIMx_EGR register) before starting the counter and not to modify the counter value during counting.

19.3.11. Clearing the OCxREF signal on an external event

For a given channel, applying a high level to the ocref_clr_int input (with the OCxCE enable bit set to 1 in the corresponding TIMx_CCMRx register) can clear the OCxREF signal. The OCxREF signal will remain low until the next update event (UEV) occurs. This feature can only be used in output compare mode and PWM mode. It does not work in Forced mode.

By configuring the OCCS bit in the TIMx_SMCR register, ocref_clr_int can be selected between the OCREF_CLR input and ETRF (filtered ETR).

For a given channel, applying a high level to the ETRF input (with the OCxCE enable bit set to 1 in the corresponding TIMx_CCMRx register) can reset the OCxREF signal. The OCxREF signal will remain low until the next update event (UEV) occurs.

This feature can only be used in output compare mode and PWM mode. It does not work in Forced mode.

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, ETR must be configured as follows:

1. The external trigger prescaler must be turned off: ETPS[1:0]=00 in the TIMx_SMCR register.
2. The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to '0'.
3. The external trigger polarity (ETP) and the external trigger filter (ETF) can be configured as needed.

The figure below shows the behavior of the OCxREF signal when the ETRF Input becomes High, for both values of the enable bit OCxCE. In this example, the timer TIMx is programmed in PWM mode.

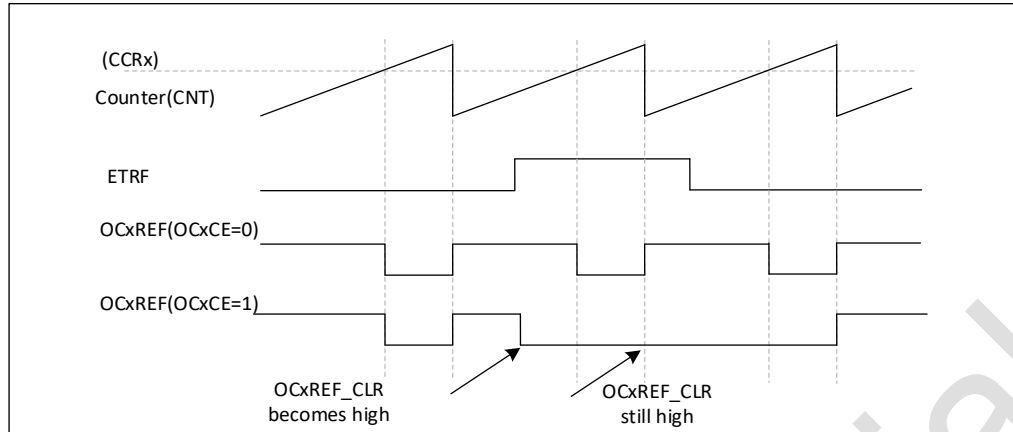


Figure 19-33 OCxREF for clearing TIMx

19.3.12. One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. One-pulse mode is selected by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV.

A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- Up-counting mode: Counter CNT < CCRx ≤ ARR (specifically, 0 < CCRx)
- Downcounting mode: Counter CNT > CCRx

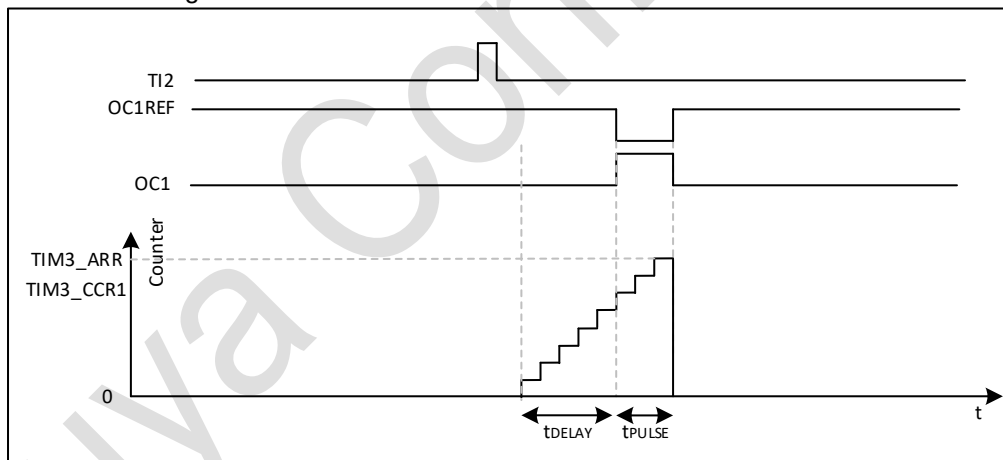


Figure 19-34 Example of One Pulse Mode

For example one may want to generate a positive pulse on OC1 with a length of tPULSE and after a delay of tDELAY as soon as a positive edge is detected on the TI2 input pin.

Let's use TI2FP2 as trigger 1:

- Use the TI2SEL[3:0] bits in the TIMx_TISEL register to select the correct TI2x source.
- Map TI2FP2 on TI2 by writing CC2S=01 in the TIMx_CCMR1 register.
- Set CC2P=0 and CC2NP=0 in the TIMx_CCER register to enable TI2FP2 to detect rising edges.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS=00110 in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The tDELAY is defined by the value written in the TIMx_CCR1 register.

- The tPULSE is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let's say one want to build a waveform with a transition from '0' to '1' when a compare match occurs and a transition from '1' to '0' when the counter reaches the auto-reload value. To do this PWM mode 2 must be enabled by writing OC1M=111 in the TIMx_CCMR1 register. Optionally the preload registers can be enabled by writing OC1PE='1' in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case one has to write the compare value in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '0' in this example.

In our example, the DIR and CMS bits in the TIMx_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0)

19.3.12.1. Particular case: OCx fast enable

In one-pulse mode, edge detection on the TIx input sets the CEN bit to 1, indicating the counter is enabled. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations and it limits the minimum delay tDELAY min we can get.

To output waveforms with minimal delay, set the OCxFE bit to 1 in the TIMx_CCMRx register. This forces OCxRef (and OCx) to respond to the excitation signal, regardless of the comparison result. Its new level is the same as if a compare match had occurred. OCxFE only takes effect when the channel is configured in PWM1 or PWM2 mode.

19.3.13. Encoder interface mode

To select Encoder Interface mode: write SMS='001' in the TIMx_SMCR register if the counter is counting on TI1 edges only, SMS=010 if it is counting on TI2 edges only and SMS=011 if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIMx_CCER register. When needed, you can program the input filter as well. CC1NP and CC2NP must be kept low.

The two inputs TI1 and TI2 are used as the interface for the quadrature encoder. Refer to the table below. Assuming the counter has been started (CEN=1 in the TIM1_CR1 register), the counter clock is driven by each valid signal transition on TI1FP1 or TI2FP2. TI1FP1=TI1 if not filtered and not inverted, TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIMx_CR1 register written to '1'). Based on the transition sequence of the two input signals, counting pulses and direction signals are generated. Depending on the transition sequence of the two input signals, the counter counts up or down, while the hardware sets the DIR bit in the TIM1_CR1 register accordingly. Whether the counter counts based on TI1, TI2, or both, the DIR bit is recalculated upon any signal transition at either input (TI1 or TI2).

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So the TIMx_ARR must be configured before starting. In the same way, the capture, compare, prescaler, trigger output features continue to work as normal. Encoder mode and External clock mode 2 are not compatible and must not be selected together.

In this mode, the counter is automatically updated based on the speed and direction of the quadrature encoder, so the counter content always indicates the encoder position. The count direction correspond to the rotation direction of the connected sensor. The table summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 19-1 Counting direction versus encoder signals

Active edge	Level of the opposite signal (TI1FP1 corresponds to TI2, TI2FP2 corresponds to TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Count only at TI1	High	Decreasing	Up	Not count	Not count
	Low	Up	Decreasing	Not count	Not count
Count only at TI2	High	Not count	Not count	Up	Decreasing
	Low	Not count	Not count	Decreasing	Up
Count at TI1 and TI2	High	Decreasing	Up	Up	Decreasing
	Low	Up	Decreasing	Decreasing	Up

A quadrature encoder can directly connect to the MCU without requiring external interface logic. However, comparators are typically used to convert the encoder's differential output into digital signals, significantly enhancing noise immunity. The third encoder output which indicates the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

The figure below gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- CC1S='01' (TIMx_CCMR1 register, IC1FP1 mapped on TI1).
- CC2S='01' (TIMx_CCMR1 register, IC2FP2 mapped on TI2).
- CC1P='0', CC1NP='0' (TIMx_CCER register, IC1FP1 non-inverted, IC1FP1=TI1).
- CC2P='0', CC2NP='0' (TIMx_CCER register, IC2FP2 non-inverted, IC2FP2=TI2).
- SMS='011' (TIMx_SMCR register, all inputs are active on both rising and falling edges).
- CEN='1' (TIMx_CR1 register, Counter enabled)

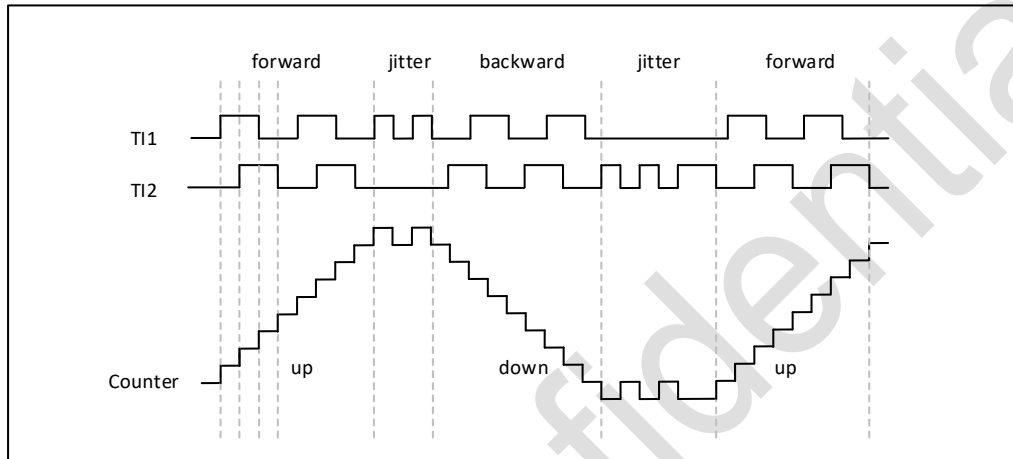


Figure 19-35 Example of Counter Operation in Encoder Interface Mode

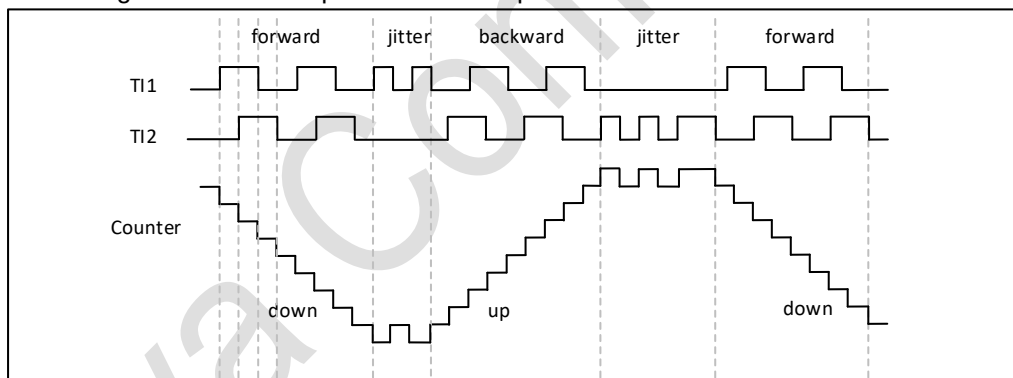


Figure 19-36 Encoder interface mode when TI1FP1 polarity is inverted

The timer, when configured in Encoder Interface mode provides information on the sensor's current position. Dynamic information can be obtained (speed, acceleration, deceleration) by measuring the period between two encoder events using a second timer configured in capture mode. The output of the encoder which indicates the mechanical zero can be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. This can be done by latching the counter value into a third input capture register if available (then the capture signal must be periodic and can be generated by another timer). When available, it is also possible to read its value through a DMA request generated by a real-time clock.

19.3.14. Timer input XOR mode

The TI1S bit in the TIMx_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIMx_CH1, TIMx_CH2 and TIMx_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture. This facilitates measuring the interval between the rising edges of two input signals.

19.3.15. Timer and External Trigger Synchronization

The TIMx timer can be synchronized with external triggers in the following modes: reset mode, gated mode, and trigger mode.

19.3.15.1. Slave mode: Reset mode

Upon a trigger input event, the counter and its prescaler can be reinitialized; simultaneously, if the URS bit in the TIMx_CR1 register is low, an update event UEV is also generated; then all the preload registers (TIMx_ARR, TIMx_CCRx) will be updated.

In the following example, the rising edge on the TI1 input causes the up-counter to be cleared:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). Since the capture prescaler is not used for trigger operations, no configuration is required. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Set CC1P=0 and CC1NP=0 in the TIMx_CCER register to determine the polarity (only rising edges are detected).
- Configure the timer in reset mode by writing SMS=100 in TIMx_SMCR register. Select TI1 as the input source by writing TS=00101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. At the same time, a trigger flag (TIF bit in the TIMx_SR register) is set, and an interrupt request or a DMA request is generated according to the setting of the TIE (Interrupt Enable) bit and the TDE (DMA Enable) bit in the TIMx_DIER register.

The following figure shows this behavior when the auto-reload register TIMx_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

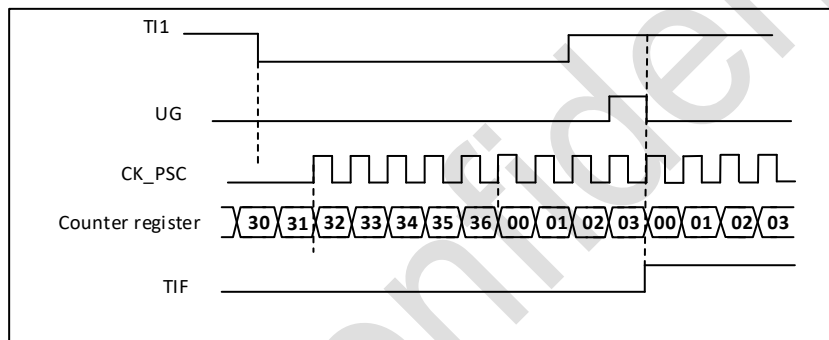


Figure 19-37 Control circuit in reset mode

19.3.15.2. Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the counter increments only when TI1 is low:

- Channel 1 is configured to detect the low level on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Set CC1P=1 (and CC1NP=0) in the TIMx_CCER register to determine polarity (only detects low level).
- Configure the timer in gated mode by writing SMS=101 in TIMx_SMCR register. Select TI1 as the input source by writing TS=00101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register. In gated mode, if CEN=0, the counter cannot start regardless of the trigger input level.

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

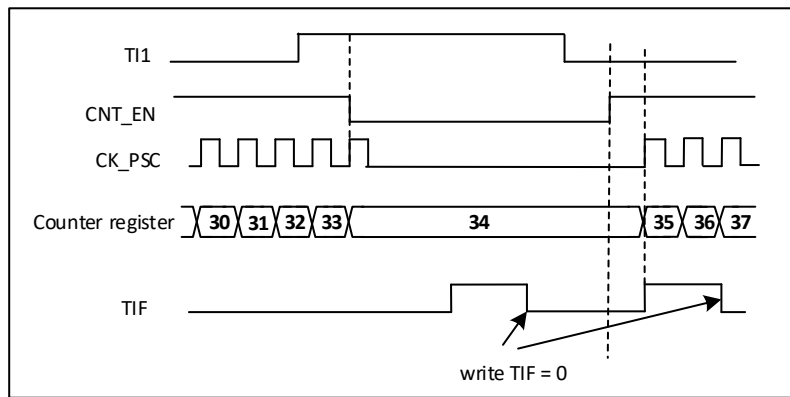


Figure 19-38 Control circuit in Gated mode

19.3.15.3. Slave mode: Trigger mode

The selected event on the input enables the counter.

In the following example, the counter starts incrementing on the rising edge of the TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we do not need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIM2x_CCMR1 register. Set CC2P=1 (and CC2NP=0) in the TIMx_CCER register to determine polarity (only detects low level).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI2 as the input source by writing TS=00110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set. The delay between the rising edge on TI2 and the actual start of the counter is due to the re-synchronization circuit on TI2 input.

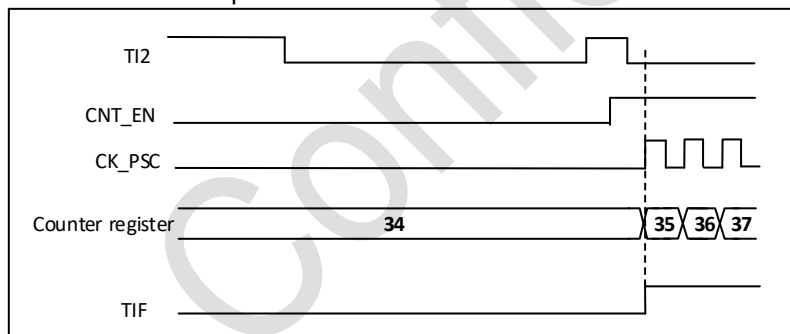


Figure 19-39 Control circuit in trigger mode

19.3.15.4. Slave mode: external clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input when operating in reset mode, gated mode or trigger mode. It is recommended not to select ETR as TRGI through the TS bits of TIMx_SMCR register.

In the following example, once a rising edge appears on TI1, the counter increments once at each rising edge of ETR:

1. Configure the external trigger input circuit by programming the TIMx_SMCR register as follows:
 - ETF=0000: No filter
 - ETPS=00: Prescaler disabled
 - ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.
2. Configure the channel 1 as follows, to detect rising edges on TI1:
 - IC1F=0000: No filter
 - The capture prescaler is not used for triggering, so it does not need to be configured.
 - CC1S=01 in TIMx_CCMR1 register to select only the input capture source
 - Set CC1P = 0 in the TIMx_CCER register to determine polarity (only rising edges are detected)
3. Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI1 as the input source by writing TS=00101 in TIMx_SMCR register.

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges. The delay between the rising edge of the ETR signal and the actual reset of the counter depends on the resynchronization circuit at the ETRP input.

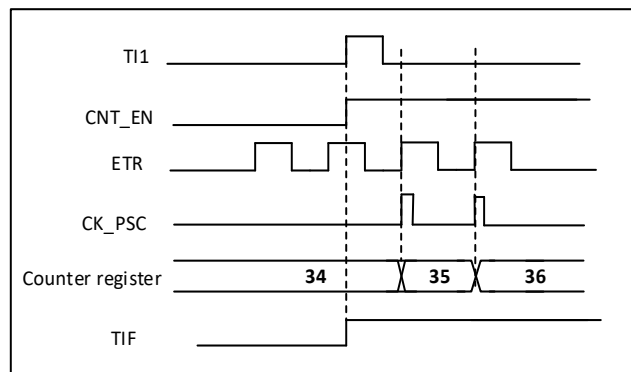


Figure 19-40 Control circuit in external clock mode 2 + trigger mode

19.3.16. Timer synchronization

TIMx timers are internally connected for timer synchronization or chaining functions. When one Timer is configured in Master mode, it can reset, start, stop or clock the counter of another Timer configured in Slave mode.

The figure below shows the trigger selection and master mode selection block diagram.

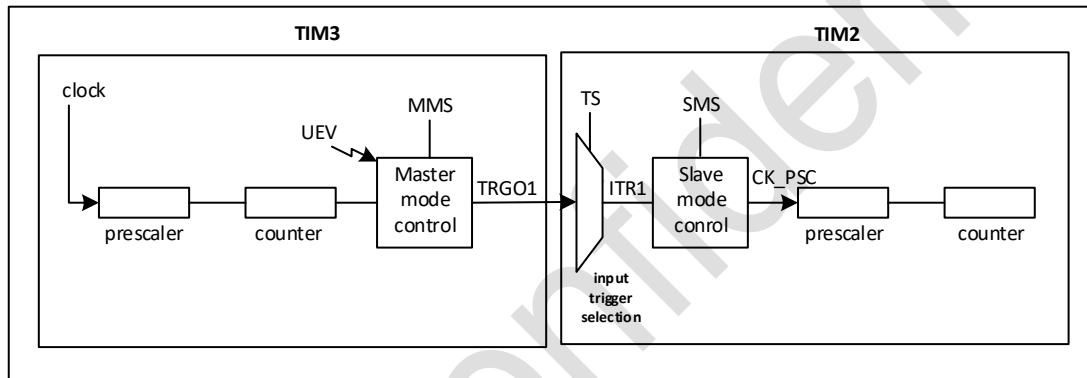


Figure 19-41 Master/slave timer example

19.3.16.1. One timer acts as another's prescaler

For example, the user can configure Timer 3 to act as a prescaler for Timer 2. To do this:

- Configure Timer 3 in master mode so that it outputs a periodic trigger signal on each update event UEV. When MMS='010' is configured in the TIM3_CR2 register, a rising edge signal is output on TRGO1 each time an update event occurs.
- To connect the TRGO output of Timer 3 to Timer 2, Timer 2 must be configured in slave mode using ITR0 as internal trigger. You select this through the TS bits in the TIM2_SMCR register (writing TS=00000).
- Then set the slave mode controller to external clock mode 1 (SMS=111 in TIM2_SMCR register); thus timer 2 can be driven by periodic rising edges (i.e., timer 3 counter overflow) from timer 3.
- Finally, the respective CEN bits (in TIM3_CR1/TIM2_CR1 registers) must be set to start both timers, ensuring timer 2 is started first, followed by timer 3.

Note: If OCx has been selected as timer 3's trigger output (MMS=01xx), its rising edge is used to drive timer 2's counter.

19.3.16.2. One timer enables another

In this example, we control the enable of Timer 2 with the output compare 3 of Timer 1. Refer to the diagram above for connections. Timer 2 counts on the divided internal clock only when OC1REF of Timer 3 is high. The clock frequencies of both timers are obtained by dividing CK_INT by 3 by the prescaler ($f_{CK\_CNT} = f_{CK\_INT}/3$).

- Configure timer 3 as master mode, sending its output compare reference signal (OC1REF) as trigger output (MMS=0100 in TIM3_CR2 register)
- Configure timer 3's OC1REF waveform (TIM3_CCMR1 register)
- Configure timer 2 to receive input trigger from timer 3 (TS=00000 in TIM2_SMCR register)
- Configure timer 2 in gated mode (SMS=101 in TIM2_SMCR register)
- Enable Timer 2 by writing '1' in the CEN bit (TIM2_CR1 register).
- Start Timer 1 by writing '3' in the CEN bit (TIM3_CR1 register).

Note: The clock of Timer 2 is not synchronized with Timer 3. This mode only affects the enable signal of Timer 2's counter.

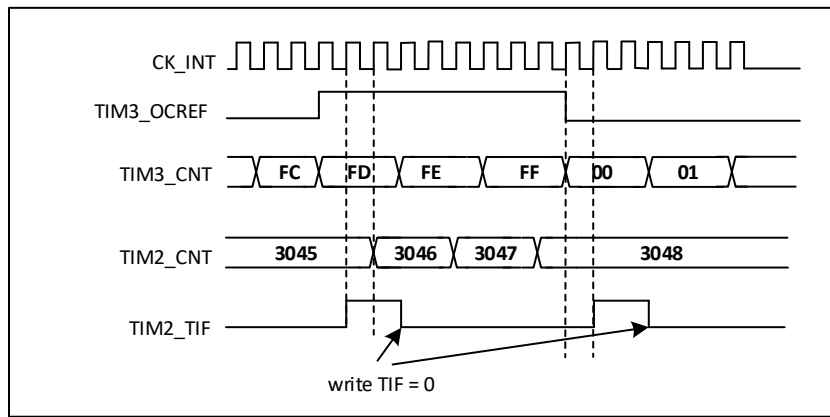


Figure 19-42 Using TIM3's OC1REF to gate-control TIM2

In the example of the figure above, the Timer 2 counter and prescaler are not initialized before being started. So they start counting from their current value. It is possible to start from a given value by resetting both timers before starting Timer 3. You can then write any value you want in the timer counters. Writing the UG bit of the TIMx_EGR register resets both timers.

In the next example, we synchronize Timer 2 and Timer 3. Timer 3 is in master mode and starts from 0, while Timer 2 is in slave mode and starts from 0xE7; both timers have the same prescaler coefficient. Timer 2 stops when Timer 3 is disabled by writing 0 to the CEN bit in the TIM3_CR1 register.

- Configure Timer 3 as master mode, sending the timer enable signal (CNT_EN) as a trigger output (MMS=0001 in TIM3_CR2 register).
- Configure the OC1REF waveform of Timer 3 (TIM1_CCMR1 register).
- Configure Timer 2 to receive input trigger from Timer 1 (TS=00000 in TIM2_SMCR register).
- Configure timer 2 in gated mode (SMS=101 in TIM2_SMCR register)
- Reset Timer 1 by writing 3 in UG bit (TIM3_EGR register).
- Reset Timer 2 by writing 1 in UG bit (TIM2_EGR register).
- Write '0xE7' to Timer 2's counter (TIM2_CNT) to initialize it to 0xE7.
- Enable Timer 2 by writing 1 in the CEN bit (TIM2_CR1 register).
- Start Timer 1 by writing 3 in the CEN bit (TIM3_CR1 register).
- Stop Timer 3 by writing 0 in the CEN bit (TIM3_CR1 register).

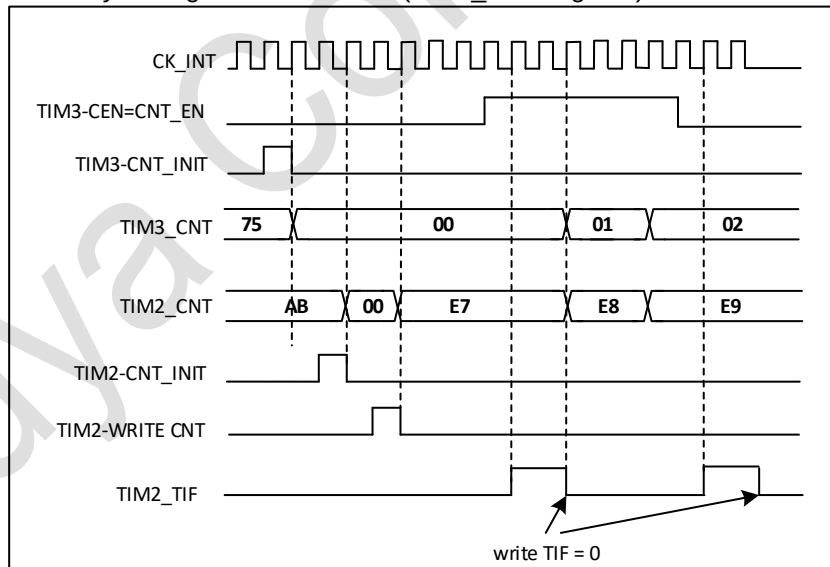


Figure 19-43 Using TIM3's enable signal to gate-control TIM2

19.3.16.3. One timer starts another

In this example, we set the enable of Timer 2 with the update event of Timer 3. Once Timer 3 generates an update event, Timer 2 starts counting from its current value (which may be non-zero) according to the divided internal clock. When Timer 2 receives the trigger signal its CEN bit is automatically set and the counter counts until we write 0 to the CEN bit in the TIM2_CR1 register. The clock frequency of both timers is divided by 3 by the prescaler pair of CK_INT ($f_{CK\_CNT} = f_{CK\_INT}/3$).

- Configure Timer 3 as master mode, sending its update event (UEV) as a trigger output (MMS=0010 in TIM3_CR2 register).
- Configure the period of Timer 3 (TIM3_ARR register).
- Configure Timer 2 to receive input trigger from Timer 1 (TS=00000 in TIM2_SMCR register).

- Configure Timer 2 in trigger mode (SMS=110 in TIM2_SMCR register).
- Start Timer 1 by writing '3' in the CEN bit (TIM3_CR1 register).

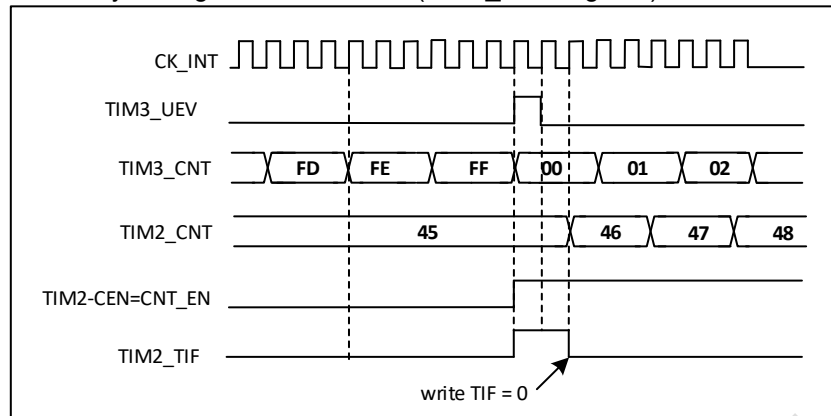


Figure 19-44 TIM3's update time enables TIM2

As in the previous example, the user can initialize both counters before starting counting. The figure below shows the operation when using trigger mode instead of gate mode (SMS=110 in TIM2_SMCR register) under the same configuration as above.

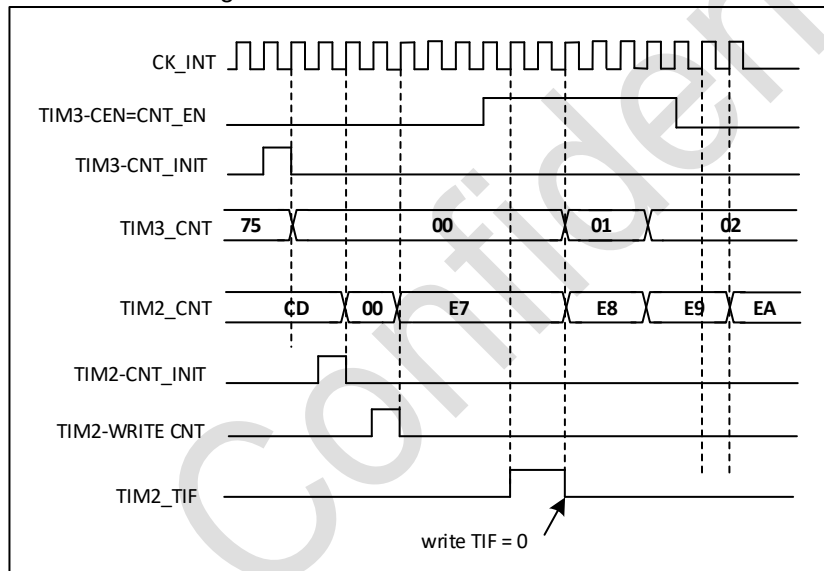


Figure 19-45 Enabling TIM3 starts TIM2

19.3.16.4. External trigger starts both timers simultaneously

In this example, Timer 2 is enabled when Timer 3's TI1 input rises, and Timer 2 is enabled simultaneously with Timer 3. To ensure counter alignment, Timer 3 must be configured in master/slave mode (slave for TI1, master for TIM2):

- Configure Timer 3 as master mode, sending its enable as a trigger output (MMS='0001' in TIM3_CR2 register).
- Configure Timer 3 as slave mode, receiving input trigger from TI1 (TS='00100' in TIM3_SMCR register).
- Configure Timer 3 in trigger mode (SMS='110' in TIM3_SMCR register).
- Configure the Timer 3 in Master/Slave mode by writing MSM=1 (TIM3_SMCR register).
- Configure Timer 2 to receive input trigger from Timer 3 (TS=00000 in TIM2_SMCR register)
- Configure Timer 2 in trigger mode (SMS='110' in TIM2_SMCR register).

When a rising edge occurs on TI1 (Timer 3), both counters start counting synchronously on the internal clock and both TIF flags are set.

Note: In this example, both timers are initialized (set the corresponding UG bits) before starting, and both counters start at 0, but an offset can be inserted between timers by writing to either counter register (TIMx_CNT). You can see that the master/slave mode insert a delay between CNT_EN and CK_PSC on timer 3.

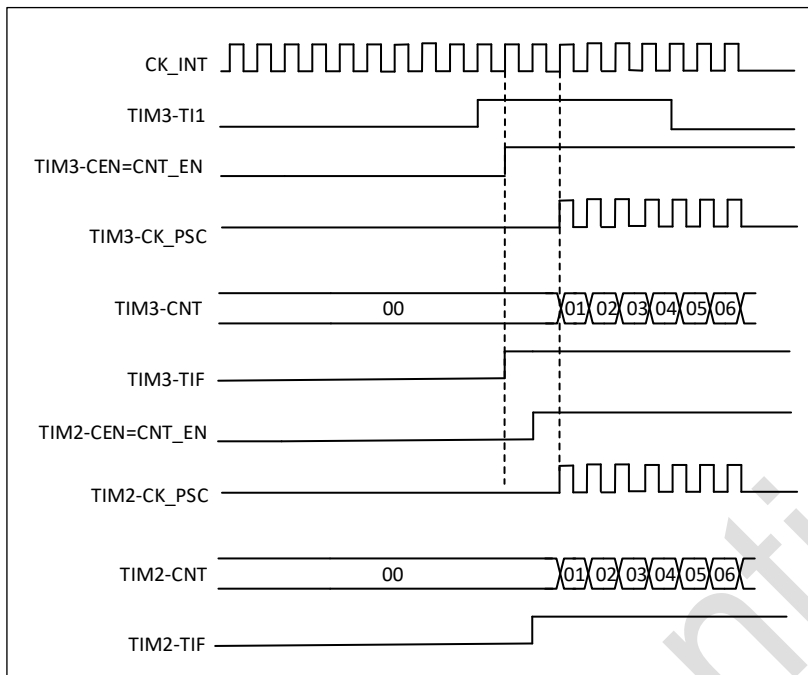


Figure 19-46 TIM3's TI1 triggering TIM3 and TIM2

19.4. TIM2/TIM3 interrupts

Table 19-2 TIM2/TIM3 interrupts

Interrupt event	Event flag	Interrupt enable control bit
update	UIF	UIE
Compare/Capture 1	CC1IF	CC1IE
Compare/Capture 2	CC2IF	CC2IE
Compare/Capture 3	CC3IF	CC3IE
Compare/Capture 4	CC4IF	CC4IE
Trigger	TIF	TIE

19.5. TIM2/TIM3 debug mode

When the chip enters debug mode, the TIM2/TIM3 counters can continue normal operation or stop working according to the DBG_TIMx_STOP setting in the DBG module.

19.6. TIM2/TIM3 registers

19.6.1. TIMx Control Register 1 (TIMx_CR1) (x=2,3)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	00	Clock division factor. These 2 bits define the division ratio between the timer clock (CK_INT) frequency and the sampling clock (t_{DTS}) used by the digital filter (ETR, TIX). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved, do not use this configuration
7	ARPE	RW	0	Auto-reload preload enable bit 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:5	CMS[1:0]	RW	00	Center-aligned mode selection. 00: Edge-aligned mode. The counter counts up or down based on the direction bit (DIR). 01: Center-aligned mode 1. The counter alternately increments and decrements. The output compare interrupt flag for channels configured as outputs The output compare interrupt flag is set only when the counter is incrementing for channels configured as output (CCxS=00 in TIMx_CCMRx register). 10: Center-aligned mode 2. The counter counts alternately up and down. The output compare interrupt flag is set only when the counter is incrementing for channels configured as output (CCxS=00 in TIMx_CCMRx register). 11: Center-aligned mode 3. The counter counts alternately up and down. The output compare interrupt flag is set for channels configured as output (CCxS=00 in TIMx_CCMRx register) when the counter is both incrementing and decrementing. Note: Switching from edge-aligned mode to center-aligned mode is not allowed when the counter is enabled (CEN=1).
4	DIR	RW	0	Counting direction. 0: Counter counts up 1: Counter counts down Note: This bit is read-only when the counter is configured in center-aligned mode or encoder mode.
3	OPM	RW	0	One-pulse mode. 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source. Software selects the source of the UEV event through this bit. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: – Counter overflow/underflow

				– Setting the UG bit – Update generation through the slave mode controller 1: If update interrupt or DMA request is enabled, only counter overflow/underflow generates an update interrupt or DMA request.
1	UDIS	RW	0	Update disable. This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller Buffered registers are then loaded with their preload values. 1: UEV disabled. The update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However, the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable. 0: Counter disabled 1: Enable counter Note: external clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However, trigger mode can set the CEN bit automatically by hardware.

19.6.2. TIMx control register 2 (TIMx_CR2) (x=2,3)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	MMS[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1S	MMS[2:0]		CCDS	Res.	Res.	Res.	Res.
								RW	RW	RW	RW	RW			

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	MMS[3]	RW	0	Refer to MMS[2:0] description.
24:8	Reserved	-	-	Reserved
7	TI1S	RW	0	TI1 selection. 0: The TIMx_CH1 pin is connected to TI1 input 1: The TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input.
6:4	MMS[2:0]	RW	000	Master mode selection. These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 0000: Reset – The UG bit of the TIM1_EGR register is used as the trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 0001: Enable – The counter enable signal CNT_EN is used as the trigger output (TRGO). Sometimes it is necessary to start multiple timers simultaneously or control a slave timer for a period of time. When configured in gated mode, the counter enable is generated by the logical AND of the CEN control bit and the trigger input signal. When the counter enable signal is controlled by the trigger input, there will be a

				delay on TRGO unless master/slave mode is selected (see the description of the MSM bit in the TIM1_SMCR register). 0010: Update – The update event is used as the trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 0011: Compare pulse – Once a capture or compare occurs successfully, when the CC1IF flag is set to 1, the trigger output sends a positive pulse (TRGO). 0100: Compare – The OC1REF signal is used as the trigger output (TRGO). 0101: Compare – The OC2REF signal is used as the trigger output (TRGO). 0110: Compare – The OC3REF signal is used as the trigger output (TRGO). 0111: Compare – The OC4REF signal is used as the trigger output (TRGO). 1000: Encoder clock output - The encoder clock signal is used as trigger output, available only when SMS[3:0] value is 0001, 0010, or 0011. Others: Reserved
3	CCDS	RW	0	Capture/compare DMA selection. 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
2:0	Reserved	-	-	Reserved

19.6.3. TIMx slave mode control register (TIMx_SMCR) (x=2,3)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS[4:3]	Res.	Res.	Res.	Res.	Res.
										RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]			ETF[3:0]			MSM		TS[2:0]		OCCS		SMS[2:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	Reserved
21:20	TS[4:3]	RW	000	See TS description for details
19:16	Reserved	-	-	Reserved
15	ETP	RW	0	External trigger polarity This bit selects whether ETR or the inversion of the ETR is used for trigger operations 0: ETR is non-inverted, active at high level or rising edge; 1: ETR inverted, active on low level or falling edge.
14	ECE	RW	0	External clock enable bit. This bit enables external clock mode 2 0: External clock mode 2 disabled; 1: External clock mode 2 enabled. The counter is driven by any active edge on the ETRF signal. Note 1: Setting ECE bit to 1 has the same effect as selecting external clock mode 1 and connecting TRGI to ETRF (SMS=111 and TS=00111). Note 2: The following slave modes can be used simultaneously with external clock mode 2: reset mode, gated mode, and trigger mode; however, in such cases, TRGI must not be connected to ETRF (TS bits must not be 00111). Note 3: If both external clock mode 1 and external clock mode 2 are enabled, the external clock input is ETRF.

13:12	ETPS[1:0]	RW	2'b0	External trigger prescaler The frequency of the external trigger signal ETR must not exceed 1/4 of the TIM1CLK frequency. The frequency of ETR can be reduced by enabling the prescaler. This method is very useful when inputting a fast external clock. 00: Prescaler OFF 01: ETR frequency divided by 2 10: ETR frequency divided by 4 11: ETR frequency divided by 8
11:8	ETF[3:0]	RW	4'b0	External trigger filter. This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. In practice, the digital filter is an event counter that records N events before considering a valid output edge. 0000: No filter, sampling is done at f_{DTS} 0001: $f_{SAMPLING} = f_{CK\_INT}$, $N = 2$ 0010: $f_{SAMPLING} = f_{CK\_INT}$, $N = 4$ 0011: $f_{SAMPLING} = f_{CK\_INT}$, $N = 8$ 0100: $f_{SAMPLING} = f_{DTS}/2$, $N = 6$ 0101: $f_{SAMPLING} = f_{DTS}/2$, $N = 8$ 0110: Sampling frequency $f_{SAMPLING} = f_{DTS}/4$, $N=6$ 0111: Sampling frequency $f_{SAMPLING} = f_{DTS}/4$, $N=8$ 1000: Sampling frequency $f_{SAMPLING} = f_{DTS}/8$, $N=6$ 1001: Sampling frequency $f_{SAMPLING} = f_{DTS}/8$, $N=8$ 1010: Sampling frequency $f_{SAMPLING} = f_{DTS}/16$, $N=5$ 1011: Sampling frequency $f_{SAMPLING} = f_{DTS}/16$, $N=6$ 1100: $f_{SAMPLING} = f_{DTS}/16$, $N = 8$ 1101: $f_{SAMPLING} = f_{DTS}/32$, $N = 5$ 1110: $f_{SAMPLING} = f_{DTS}/32$, $N = 6$ 1111: $f_{SAMPLING} = f_{DTS}/32$, $N = 8$
7	MSM	RW	0	Master/slave mode. 0: No action 1: The event on the trigger input (TRGI) is delayed to achieve perfect synchronization between the current timer and its slave timer (via TRGO). It is useful if we want to synchronize several timers on a single external event.
6:4	TS[2:0]	RW	3'b0	Trigger selection These 5 bits select the trigger input used to synchronize the counter. 00000: Internal Trigger 0 (ITR0) 00001: Internal Trigger 1 (ITR1) 00010: Internal Trigger 2 (ITR2) 00011: Internal Trigger 3 (ITR3) 00100: TI1 edge detection (TI1F_ED) 00101: Filtered timer input 1 (TI1FP1) 00110: Filtered timer input 2 (TI1FP2) 00111: external trigger input (ETRF) 01000: Internal Trigger 4 (ITR4) 01001: Internal Trigger 5 (ITR5) Others: Reserved Note: These bits can only be changed when not in use (e.g., SMS=0000) to avoid false edge detection during transitions. Note: For ITRx mapping, refer to the "Peripheral Interconnection" section.
3	OCCS	RW	0	OCREF clear selection This bit is used to select the OCREF clear source. 0: OCREF_CLR_INT clear source is connected to OCREF_CLR input 1: OCREF_CLR_INT is connected to ETRF

2:0	SMS[2:0]	RW	3'b0	Slave mode selection When an external signal is selected, the active edge of the trigger signal (TRGI) depends on the selected polarity of the external input (refer to the descriptions of the input control register and control register). 000: Slave mode disabled---If CEN=1, the prescaler is directly clocked by the internal clock. 001: Encoder mode 1---The counter counts up/down on edges of TI1FP1 depending on the level of TI2FP2. 010: Encoder mode 2---The counter counts up/down on edges of TI2FP2 depending on the level of TI1FP1. 011: Encoder mode 3---The counter counts up/down on edges of both TI1FP1 and TI2FP2 depending on the level of the other input. 100: Reset mode---The rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update register signal. 101: Gated mode---The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger mode---The counter starts on the rising edge of the trigger input TRGI (but is not reset), only controlling the counter's start. 111: External clock mode 1---The rising edge of the selected trigger input (TRGI) clocks the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS=00100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

19.6.4. TIMx DMA/interrupt enable register (TIMx_DIER) (x=2,3)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	RW		RW	RW	RW	RW	RW		RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	Reserved
14	TDE	RW	0	Trigger DMA request enable 0: Trigger DMA request disabled 1: Trigger DMA request enabled
13	Reserved	-	-	Reserved
12	CC4DE	RW	0	Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled 1: CC4 DMA request enabled
11	CC3DE	RW	0	Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled 1: CC3 DMA request enabled
10	CC2DE	RW	0	Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled 1: CC2 DMA request enabled
9	CC1DE	RW	0	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled 1: CC1 DMA request enabled
8	UDE	RW	0	Update DMA request enable

				0: Update DMA request disabled 1: Update DMA request enabled
7	Reserved	-	-	Reserved
6	TIE	RW	0	Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled
5	Reserved	-	-	Reserved
4	CC4IE	RW	0	Capture/Compare 4 interrupt enable 0: Capture/Compare 4 interrupt disabled 1: Capture/Compare 4 interrupt enabled
3	CC3IE	RW	0	Capture/Compare 3 interrupt enable 0: Capture/Compare 3 interrupt disabled 1: Capture/Compare 3 interrupt enabled
2	CC2IE	RW	0	Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled
1	CC1IE	RW	0	Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

19.6.5. TIMx status register (TIMx_SR) (x=2,3)

Offset Address: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Re s.	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	Re s.	Res.	Re s.	Res.	IC2IR	IC1IR	Re s.	Res.
		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0					RC_W0	RC_W0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	Re s.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	Re s.	TIF	Re s.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			RC_W0	RC_W0	RC_W0	RC_W0			RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29	IC4IF	RC_W0	0	Falling edge capture 4 flag Refer to IC1IF description
28	IC3IF	RC_W0	0	Falling edge capture 3 flag Refer to IC1IF description
27	IC2IF	RC_W0	0	Falling edge capture 2 flag Refer to IC1IF description
26	IC1IF	RC_W0	0	Falling edge capture 1 flag This flag can be set to 1 by hardware only when the corresponding channel is configured for input capture and triggered by a falling edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A falling edge capture event occurs.
25	IC4IR	RC_W0	0	Rising edge capture 4 flag Refer to IC1IR description
24	IC3IR	RC_W0	0	Rising edge capture 3 flag Refer to IC1IR description
23:20	Reserved	-	-	Reserved
19	IC2IR	RC_W0	0	Rising edge capture 2 flag Refer to IC1IR description
18	IC1IR	RC_W0	0	Rising edge capture 1 flag

				This flag can be set to 1 by hardware only when the corresponding channel is configured for input capture and triggered by a rising edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A rising edge capture event occurs.
17:13	Reserved	-	-	Reserved
12	CC4OF	RC_W0	0	Capture/Compare 4 Overcapture Flag Refer to CC1OF description
11	CC3OF	RC_W0	0	Capture/Compare 3 Overcapture Flag Refer to CC1OF description
10	CC2OF	RC_W0	0	Capture/Compare 2 Overcapture Flag Refer to CC1OF description
9	CC1F	RC_W0	0	Capture/Compare 1 Overcapture Flag This flag can be set to 1 by hardware only when the corresponding channel is configured as input capture. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in TIMx_CCR1 register while CC1F flag was already set.
8:7	Reserved	-	-	Reserved
6	TIF	RC_W0	0	Trigger Interrupt Flag In all modes except gated mode, this bit is set to 1 by hardware when a valid edge is detected at the TRGI input after enabling the slave mode controller. In gated mode, this flag will be set to 1 when the counter starts or stops, but it must be cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	Reserved	-	-	Reserved
4	CC4IF	RC_W0	0	Capture/Compare 4 interrupt flag Refer to CC1IF description
3	CC3IF	RC_W0	0	Capture/Compare 3 Interrupt Flag Refer to CC1IF description
2	CC2IF	RC_W0	0	Capture/Compare 2 interrupt flag Refer to CC1IF description
1	CC1IF	RC_W0	0	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This bit is set to 1 by hardware when the counter value matches the compare value, except in center-aligned mode (refer to the CMS bit in the TIMx_CR1 register). It is cleared by software. 0: No match; 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. Or when the value of TIM1_CCR1 is greater than TIM1_ARR (CC1IF is set to 1 during overflow in up-counting and center-aligned modes, or during underflow in down-counting). If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred

				1: The counter value has been captured in TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	RC_W0	0	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: – If UDIS=0 in the TIMx_CR1 register, an update event is generated when REP_CNT=0 (when the repeat counter overflows or underflows); – If UDIS=0 and URS=0 in the TIMx_CR1 register, an update event is generated when UG=1 in the TIMx_EGR register (software reinitializes CNT); – If UDIS = 0 and URS = 0 of the TIMx_CR1 register, an update event occurs when the CNT is reinitialized by the trigger event (Refer to the Slave Mode Control Register (TIMx_SMCR))

19.6.6. TIMx Event Generation Register (TIMx_EGR) (x=2,3)

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
									W		W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6	TG	W	0	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: When TIF=1 in the TIMx_SR register, the corresponding interrupt or DMA transfer event can occur if enabled.
5	Reserved	-	0	Reserved, must be kept at reset value.
4	CC4G	W	0	Capture/Compare 4 generation Refer to CC1G description
3	CC3G	W	0	Capture/Compare 3 generation Refer to CC1G description
2	CC2G	W	0	Capture/Compare 2 generation Refer to CC1G description
1	CC1G	W	0	Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as input: CC1IF is set, generating the corresponding interrupt and DMA if enabled. If channel CC1 is configured as input: The current counter value is captured into the TIMx_CCR1 register, CC1IF is set, generating the corresponding interrupt and DMA if enabled. The CC1OF flag is set if the CC1IF flag was already set.

0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). In center-aligned mode or when DIR=0 (upcounting), the counter is cleared to 0; when DIR=1 (downcounting), the counter is loaded with the TIMx_ARR value.

19.6.7. TIMx capture/compare mode register 1 (TIMx_CCMR1) (x=2,3)

Offset address: 0x18

Reset value: 0x0000 0000

The same register can be used for input capture mode or output compare mode. The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function for input capture and for output compare modes. For any given bit, OCxx indicates the function of the bit when the channel is configured as an output, while ICxx indicates the function when the channel is configured as an input.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE		OC2M [2:0]		OC2PE	CO2FE	CC2S [1:0]		OC1CE	OC1M [2:0]		OC1PE	OC1FE	CC1S [1:0]		
		IC2F [3:0]		IC2PSC [1:0]				IC1F [3:0]		IC1PSC [1:0]					
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

19.6.7.1. Input capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	IC2F	RW	4'b0	Input capture 2 filter. Refer to the description of IC1F[3:0].
11:10	IC2PSC [1:0]	RW	2'b0	Input/Capture 2 prescaler. Refer to the description of IC1PSC[1:0].
9:8	CC2S [1:0]	RW	0	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7:4	IC1F [3:0]	RW	4'b0	Input capture 1 filter. This bit field defines the sampling frequency of the TI1 input and the sampling length of the digital filter applied to TI1. The digital filter consists of an event counter, where only every N consecutive events are considered a valid output edge: 0000: No filter, sampling is done at f_{DTS} 0001: Sampling frequency f_{SAMPLING} = f_{CK_INT}, N = 2 0010: Sampling frequency f_{SAMPLING} = f_{CK_INT}, N = 4 0011: Sampling frequency f_{SAMPLING} = f_{CK_INT}, N = 8 0100: f_{SAMPLING} = f_{DTS}/2, N = 6 0101: f_{SAMPLING} = f_{DTS}/2, N = 8 0110: f_{SAMPLING} = f_{DTS}/4, N = 6 0111: f_{SAMPLING} = f_{DTS}/4, N = 8 1000: f_{SAMPLING} = f_{DTS}/8, N = 6

				1001: $f_{\text{SAMPLING}} = f_{\text{DTS}}/8$, $N = 8$ 1010: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 5$ 1011: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 6$ 1100: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 8$ 1101: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 5$ 1110: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 6$ 1111: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 8$
3:2	IC1PSC [1:0]	RW	2'b0	Input/capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). Once CC1E=0 (in the TIMx_CCER register), the prescaler is reset. 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S [1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

19.6.7.2. Output compare mode

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	OC2CE	RW	0	Output compare 2 clear enable. Refer to the description of OC1CE.
14:12	OC2M [2:0]	RW	3'b0	Output Compare 2 mode selection. Refer to the description of OC1M[2:0].
11	OC2PE	RW	0	Output Compare 2 preload enable. Refer to the description of OC1PE.
10	OC2FE	RW	0	Output Compare 2 fast enable. Refer to the description of OC1FE.
9:8	CC2S [1:0]	RW	2'b0	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7	OC1CE	RW	0	Output Compare 1 clear enable. 0: OC1REF is not affected by ETRF or OCREF_CLR input; 1: Once a high level is detected on ETRF or OCREF_CLR input, OC1REF is cleared.
6:4	OC1M [2:0]	RW	3'b0	Output Compare 1 mode. This bit defines the behavior of the output reference signal OC1REF for OC1 and OC1N. OC1REF is active

				high, while the active level of OC1 and OC1N depends on the CC1P and CC1NP bits. 000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 001: Set channel 1 to output active level upon match. When the value of counter TIMx_CNT matches Capture/Compare Register 1 (TIMx_CCR1), force OC1REF to high level. 010: Set channel 1 to output inactive level upon match. When the value of counter TIMx_CNT matches Capture/Compare Register 1 (TIMx_CCR1), force OC1REF to low level. 011: Toggle When TIMx_CCR1=TIMx_CNT, OC1REF toggles. 100: Force inactive level Force OC1REF to low level. 101: Force active level Force OC1REF to high level. 110: PWM mode 1—In up-counting, channel 1 is active when TIMx_CNT<TIMx_CCR1, otherwise inactive; in down-counting, channel 1 is inactive (OC1REF=0) when TIMx_CNT>TIMx_CCR1, otherwise active (OC1REF=1). 111: PWM mode 2—In up-counting, channel 1 is inactive when TIMx_CNT<TIMx_CCR1, otherwise active; in down-counting, channel 1 is active when TIMx_CNT>TIMx_CCR1, otherwise inactive. Note: In PWM mode 1 or 1, the OCREF level changes when the result of the comparison changes or when the output compare mode switches from frozen to PWM mode.
3	OC1PE	RW	0	Output Compare 1 preload enable. 0: Disable preload function of TIMx_CCR1 register. TIMx_CCR1 register can be written at any time, and the new value takes effect immediately. 1: Preload register on TIMx_CCR1 enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is loaded in the active register at each update event. Note 1: PWM mode can only be used in single-pulse mode without acknowledging the preload register; otherwise, the behavior is undefined.
2	OC1FE	RW	0	Output Compare 1 fast enable. This bit is used to accelerate the effect of trigger input events on CC outputs. 0: Even if the trigger is active, CC1 will operate normally based on the counter and CCR1 values. When a trigger input edge occurs, the minimum delay to activate CC1 output is 5 clock cycles. 1: A valid edge on the trigger input is equivalent to a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. The delay between sampling the trigger input and CC1 output is reduced to 3 clock cycles. OCxFE is effective only when the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	RW	2'b0	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2;

				11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM1_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

19.6.8. TIMx capture/compare mode register 2 (TIMx_CCMR2) (x=2,3)

Offset address: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M [2:0]			OC4PE	CO4FE	CC4S [1:0]		OC3CE	OC3M [2:0]			OC3PE	OC3FE	CC3S [1:0]	
IC4F [3:0]				IC4PSC [1:0]				IC3F [3:0]			IC3PSC [1:0]				
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

19.6.8.1. Input capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	IF4F	RW	0000	Input capture 4 filter. Refer to the description of IC1F.
11:10	IC4PSC [1:0]	RW	2'b0	Input/Capture 4 prescaler. Refer to the description of IC1PSC.
9:8	CC4S [1:0]	RW	2'b0	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3. 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIMx_SMCR register). Note: CC4S is writable only when the channel is closed (CC4E = 0 in the TIMx_CCER register).
7:4	IC3F [3:0]	RW	4'b0	Input capture 1 filter. Refer to the description of IC1F.
3:2	IC3PSC [1:0]	RW	2'b0	Input/Capture 3 prescaler. Refer to the description of IC1PSC.
1:0	CC3S [1:0]	RW	2'b0	CC3S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3. 10: CC3 channel is configured as input, IC3 is mapped on TI3; 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIMx_SMCR register). Note: CC3S is writable only when the channel is closed (CC3E = 0 in the TIMx_CCER register).

19.6.8.2. Output compare mode

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	OC4CE	RW	0	Output Compare 4 clear enable. Refer to the description of OC1CE.
14:12	OC4M [2:0]	RW	3'b0	Output compare 4 mode Refer to the description of OC4M.

11	OC4PE	RW	0	Output Compare 4 preload enable. Refer to the description of OC1PE.
10	OC4FE	RW	0	Output Compare 4 fast enable. Refer to the description of OC1FE.
9:8	CC4S [1:0]	RW	2'b0	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3. 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIMx_SMCR register). Note: CC4S is writable only when the channel is closed (CC4E = 0 in the TIMx_CCER register).
7	OC3CE	RW	0	Output Compare 3 clear enable. Refer to the description of OC1CE.
6:4	OC3M [2:0]	RW	3'b0	Output Compare 3 mode. Refer to the description of OC1M[3:0].
3	OC3PE	RW	0	Output Compare 3 preload enable. Refer to the description of OC1PE.
2	OC3FE	RW	0	Output Compare 3 fast enable. Refer to the description of OC1FE.
1:0	CC3S [1:0]	RW	2'b0	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3. 10: CC3 channel is configured as input, IC3 is mapped on TI4. 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIMx_SMCR register). Note: CC3S is writable only when the channel is closed (CC3E = 0 in the TIMx_CCER register).

19.6.9. TIMx capture/compare enable register (TIMx_CCER) (x=2,3)

Offset address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Res.	CC4P	CC4E	CC3NP	Res	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
RW		RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	CC4NP	RW	0	Input/Capture 4 complementary output polarity. Refer to CC1NP description.
14	Reserved	-	-	Reserved
13	CC4P	RW	0	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	RW	0	Input/Capture 4 output enable Refer to CC1E description.
11	CC3NP	RW	0	Input/Capture 3 complementary output polarity. Refer to CC1NP description.
10	Reserved	-	-	Reserved

9	CC3P	RW	0	Input/Capture 3 output polarity Refer to CC1P description.
8	CC3E	RW	0	Input/Capture 3 output enable Refer to CC1E description.
7	CC2NP	RW	0	Input/Capture 2 complementary output polarity. Refer to CC1NP description.
6	Reserved	-	-	Reserved
5	CC2P	RW	0	Input/Capture 2 output polarity Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 output enable Refer to CC1E description.
3	CC1NP	RW	0	This bit, together with CC1P, defines the polarity of TI1FP1/TI2FP1. Refer to the description of CC1P.
2	Reserved	-	-	Reserved
1	CC1P	RW	0	Capture/Compare 1 output polarity CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: The [CC1NP, CC1P] bits select the polarity of TI1FP1 and TI2FP1 as trigger or capture signals: 00: Non-inverted/rising edge. TIxFP1 rising edge active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 non-inverted (gated mode, encoder mode). 01: Inverted/falling edge. TIxFP1 falling edge active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 inverted (gated mode, encoder mode). 10: reserved, do not use this configuration. 11: Non-inverted/both edges. TIxFP1 is active on both rising and falling edges (capture or trigger operations performed in reset mode, external clock mode or trigger mode); TIxFP1 non-inverted (gated mode). This configuration must not be used in encoder mode.
0	CC1E	RW	0	Input/Capture 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active 1: On - OC1 signal output to the corresponding output pin CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled.

Table19-3 Output control bits for standard OCx channels

CCxE bit	OCx output state
0	Output Disabled (OCx=0, OCx_EN=0)
1	OCx=OCxREF+polarity, OCx_EN=1

19.6.10. TIMx counter register (TIMx_CNT) (x=2,3)

Offset address: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function

31:0	CNT[31:0]	RW	0	Counter value Note: TIM3 valid bits are [15:0]

19.6.11. TIMx prescaler register (TIMx_PSC) (x=2,3)

Offset address: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	16'b0	Prescaler value. The counter clock frequency (CK_CNT) is equal to $f_{CK\_PSC} / (PSC[15:0] + 1)$. PSC contains the value to be loaded into the active prescaler register each time an update event occurs (including when the counter is cleared via the UG bit in the TIM1_EGR register, or when cleared by the trigger controller when configured in "reset mode").

19.6.12. TIMx auto-reload register (TIMx_ARR) (x=2,3)

Offset address: 0x2C

Reset value: 0xFFFF FFFF

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

Bit	Name	R/W	Reset Value	Function
31:0	ARR[31:0]	RW	32'hFFFFFFFF	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null. Note: TIM3 valid bits are [15:0]

19.6.13. TIMx capture/compare register 1 (TIMx_CCR1) (x=2,3)

Offset address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1 [31:16]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1 [15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:0	CCR1 [31:0]	RW	32'b0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 contains the value to be loaded into the current capture/compare register 1 (preload value). If the preload feature is not selected in the TIMx_CCMR1 register (OC1PE bit), the value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the counter TIMx_CNT, and outputs the signal on the OC1 port. If channel CC1 is configured as input:

				CCR1 is the counter value at the time of the last input capture 1 event (IC1). At this time, the TIMx_CCR1 register can only be read and cannot be programmed. Note: TIM3 valid bits are [15:0]

19.6.14. TIMx capture/compare register 2 (TIMx_CCR2) (x=2,3)

Offset Address: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2 [31:16]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2 [15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:0	CCR2 [31:0]	RW	32'b0	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 contains the value to be loaded into the current capture/compare register 2 (preload value). If the preload feature is not selected in the TIMx_CCMR1 register (OC2PE bit), the value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the counter TIMx_CNT, and outputs the signal on the OC2 port. If channel CC2 is configured as input: CCR2 is the counter value at the time of the last input capture 2 event (IC2). At this time, only the TIMx_CCR2 register can be read, and it cannot be programmed. Note: TIM3 valid bits are [15:0]

19.6.15. TIMx capture/compare register 3 (TIMx_CCR3) (x=2,3)

Offset Address: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3 [31:16]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3 [15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:0	CCR3 [31:0]	RW	32'b0	Capture/Compare 3 value If channel CC3 is configured as output: CCR3 contains the value to be loaded into the current capture/compare register 3 (preload value). If the preload feature is not selected in the TIMx_CCMR2 register (OC3PE bit), the value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the counter TIMx_CNT, and outputs the signal on the OC3 port. If channel CC3 is configured as input: CCR3 is the counter value at the time of the last input capture 3 event (IC3). At this time, the TIMx_CCR3 register can only be read and cannot be programmed. Note: TIM3 valid bits are [15:0]

19.6.16. TIMx capture/compare register 4 (TIMx_CCR4) (x=2,3)

Offset address: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4 [31:16]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4 [15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:0	CCR4 [31:0]	RW	32'b0	Capture/Compare 4 value If channel CC4 is configured as output: CCR4 contains the value to be loaded into the current capture/compare register 4 (preload value). If the preload feature is not selected in the TIMx_CCMR2 register (OC4PE bit), the value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the counter TIMx_CNT, and outputs the signal on the OC4 port. If channel CC4 is configured as input: CCR4 is the counter value at the time of the last input capture 4 event (IC4). At this time, only the TIMx_CCR4 register can be read, and it cannot be programmed. Note: TIM3 valid bits are [15:0]

19.6.17. TIMx input selection register (TIMx_TISEL) (x=2,3)

Offset address: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TI4SEL [3:0]				Res.	Res.	Res.	Res.	TI3SEL [3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	TI2SEL [3:0]				Res.	Res.	Res.	Res.	TI1SEL [3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:24	TI4SEL [3:0]	RW	4'b0	TI4 input selection. 0000: TIMx_CH4 0001: COMP1_OUT 0010 COMP2_OUT Other: Reserved
23:20	Reserved	-	-	Reserved
19:16	TI3SEL [3:0]	RW	4'b0	TI3 input selection. 0000: TIMx_CH3 0001: COMP1_OUT 0010 COMP2_OUT Other: Reserved
15:12	Reserved	-	-	Reserved
11:8	TI2SEL [3:0]	RW	4'b0	TI2 input selection. 0000: TIMx_CH2 0001: COMP1_OUT 0010 COMP2_OUT Other: Reserved
7:4	Reserved	-	-	Reserved
3:0	TI1SEL [3:0]	RW	4'b0	TI1 input selection. 0000: TIMx_CH1 0001: COMP1_OUT

				0010 COMP2_OUT Other: Reserved

19.6.18. TIMx alternate function register 1 (TIMx_AF1) (x=2,3)

Offset address: 0x60

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	Reserved
17:14	ETRSEL[3:0]	RW	4'b0	External trigger source selection (etr_in source selection) This bit segment is used to select an ETR input source 0000: TIM2_ETR for TIM2, TIM3_ETR for TIM3 0001: COMP1_OUT 0010 COMP2_OUT 0011: Reserved 0100: Reserved 0101: TIM3_ETR for TIM2, TIM2_ETR for TIM3 Others: Reserved
13:0	Reserved	-	-	Reserved

19.6.19. TIMx alternate function register 2 (TIMx_AF2) (x=2,3)

Offset address: 0x64

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18:16	OCRSEL[2:0]	RW	3'b0	OCREF reset source selection. This bit segment is used to select that ocref_clr input source 0000: COMP1_OUT 0001: COMP2_OUT Others: Reserved
15:0	Reserved	-	-	Reserved

19.6.20. TIMx DMA control register (TIMx_DCR) (x=2,3)

Offset address: 0x3DC

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12:8	DBL[4:0]	RW	5'b0	DMA burst length These bits define the transfer length in DMA burst mode (when reading or writing to the TIMx_DMAR

				register address, the timer performs one burst transfer). 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers 10001: 18 transfers
7:5	Reserved	-	-	Reserved
4:0	DBA[4:0]	RW	5'b0	DBA[4:0]: DMA base address These bits define the base address of the DMA in continuous mode (when read/write access are done through the TIMx_DMAR address). DBA is defined as an offset starting from the address of the TIMx_CR1 register. 00000: TIMx_CR1 00001: TIMx_CR2 00010: TIMx_SMCR

19.6.21. TIMx DMA burst transfer address register (TIMx_DMAR) (x=2,3)

Offset Address: 0x3E0

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[31:0]	RW	32'b0	DMA register for burst accesses A read or write operation to the DMAR register accesses the register located at the address: TIMx_CR1 address + (DBA + DMA index)*4, where: TIMx_CR1 address is the address of the control register 1; DBA is the DMA baseaddress configured in TIMx_DCR register; DMA index is automatically controlled by the DMA transfer, and ranges from 0 to DBL (DBL configured in TIMx_DCR).

20. General-purpose timers (TIM15)

20.1. TIM15 main features

- 16-bit auto-reload upcounter
- 16-bit programmable prescaler used to divide the counter clock frequency by any factor between 1 and 65536 (can be changed “on the fly”).
- Two channels can be used as:
 - Input capture
 - Output compare
 - PWM generation (edge-aligned mode)
 - One-pulse mode output
- Complementary outputs with programmable dead-time (only for channel 1)
- Synchronization circuit for controlling the timer with external signals and enabling interconnection of multiple timers
- Repetition counter to update the timer registers only after a given number of cycles of the counter.
- The break input can place the timer's output signals into a user-selectable safe configuration
- Interrupt/DMA is generated when the following events occur:
 - Update: counter overflow, counter initialization (by software or internal/external trigger)
 - Trigger event (counter starts, stops, initializes or counts by internal/external triggers)
 - Input capture
 - Output compare
 - Break input

20.2. TIM15 functional description

20.2.1. Functional block diagram

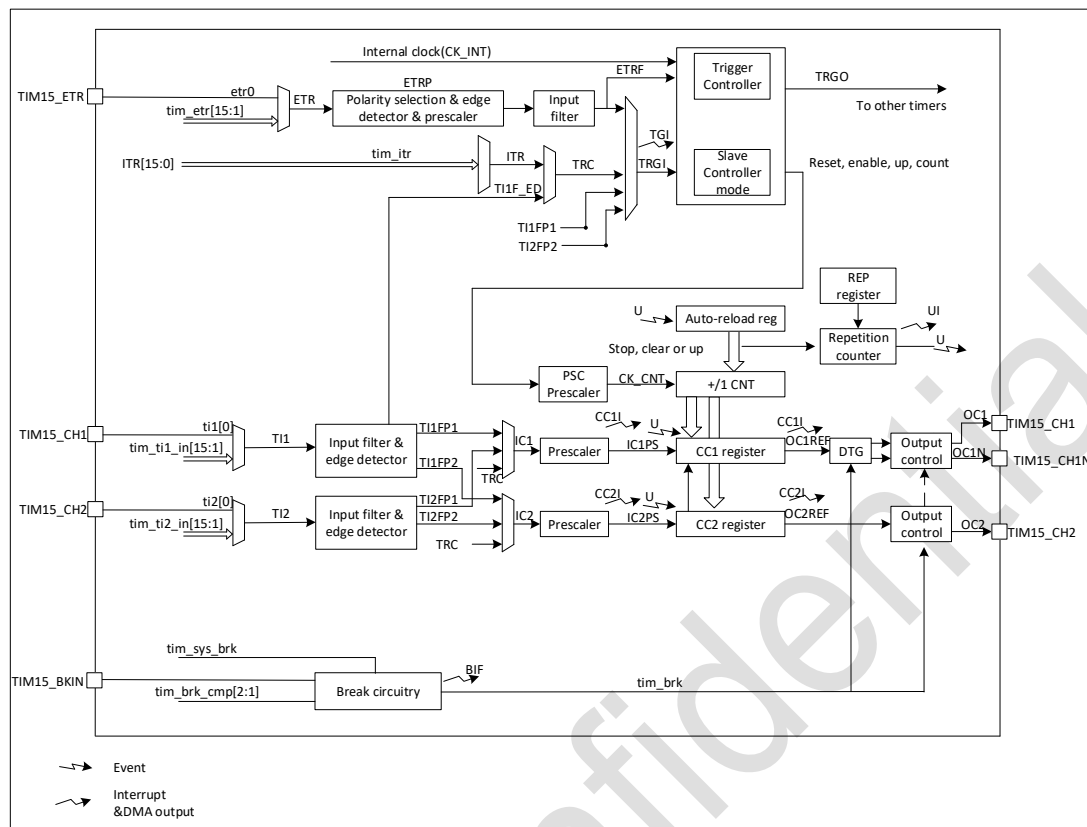


Figure 20-1 TIM15 block diagram

20.2.2. Time-base unit

The main part of this programmable timer is a 16-bit upcounter with auto-reload capability. The counter clock can be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIM15_CNT)
- Prescaler register (TIM15_PSC)
- Auto-reload register (TIM15_ARR)
- Repetition counter register (TIM15_RCR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM15_CR1 register. An update event is generated when the counter reaches the overflow value and the UDIS bit in the TIM15_CR1 register is 0. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM15_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIM15_CR register.

20.2.2.1. Prescaler description

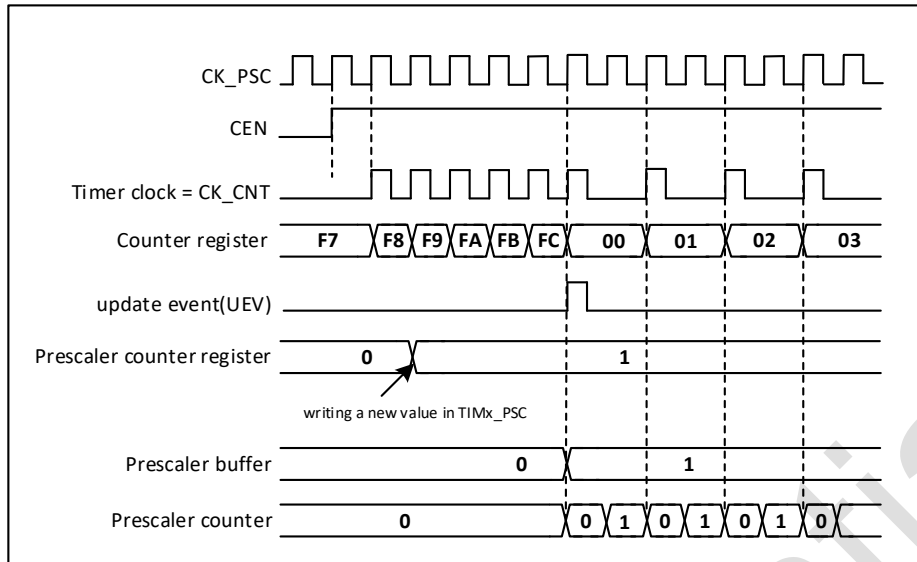


Figure 20-2 Counter timing diagram when prescaler division changes from 1 to 2

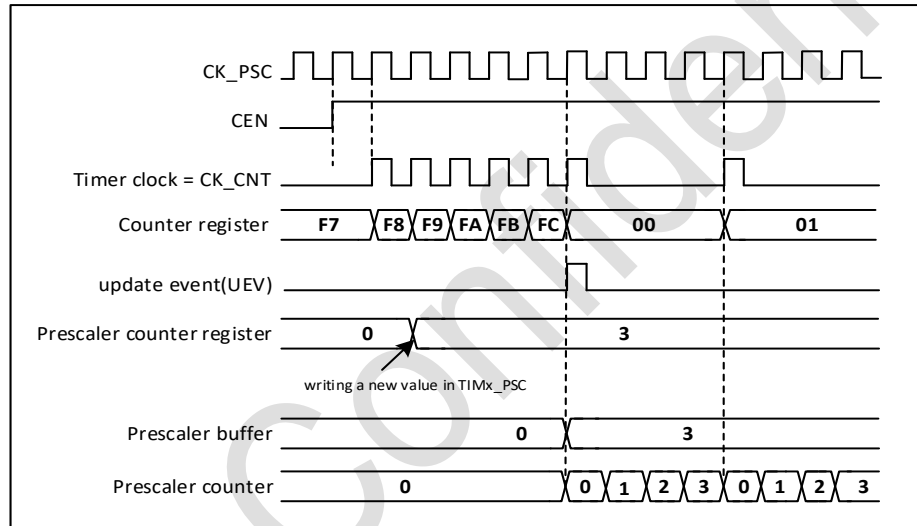


Figure 20-3 Counter timing diagram when prescaler division changes from 1 to 4

20.2.3. Counting mode (upcounting)

The counter counts from 0 to the auto-reload value (contents of the TIM15_ARR register), then re-starts from 0 and generates a counter overflow event.

If the repetition counter is used, an update event (UEV) is generated when the upcounting repetition reaches the value set in the repetition counter register plus one ((TIM15_RCR) + 1). Else the update event is generated at each counter overflow.

Setting the UG bit in the TIM15_EGR register (via software or using the slave mode controller) can also generate an update event.

Setting the UDIS bit in the TIM15_CR1 register disables the update event (UEV); this also prevents updating the shadow registers when writing new values to the preload registers. Then no update event occurs until the UDIS bit has been written to 0. Even so, the counter is still cleared to '0' when an update event should occur, and the prescaler count is also cleared (though the prescaler value remains unchanged). Additionally, if the URS bit in the TIM15_CR1 register is set (update request selection), setting the UG bit generates an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all registers are updated, and the update flag (UIF bit in the TIM15_SR register) is set to 1 (depending on the URS bit).

- The repetition counter is reloaded with the value in the TIM15_RCR register.
- The auto-reload shadow register is updated with the preload value (TIM15_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIM15_PSC register).

The following figure shows several counter behaviors at different frequencies when TIM15_ARR=0x36.

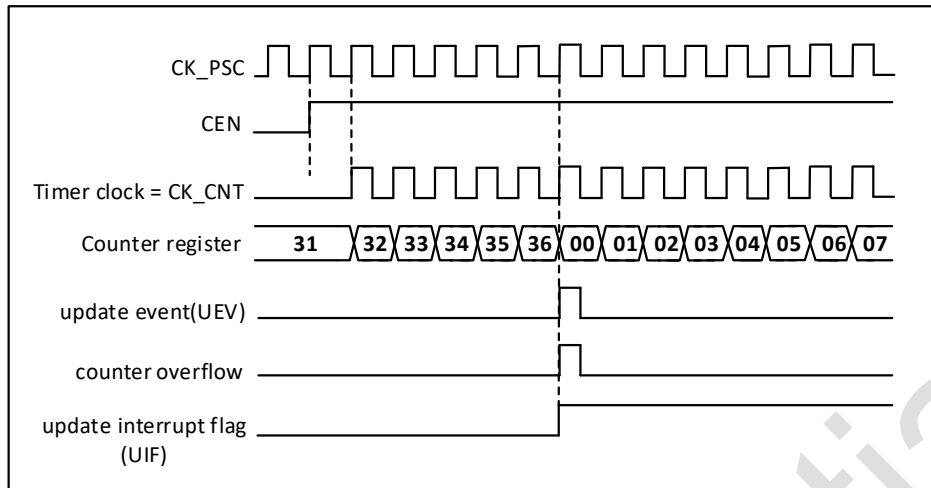


Figure 20-4 Counter timing diagram, internal clock divided by 1

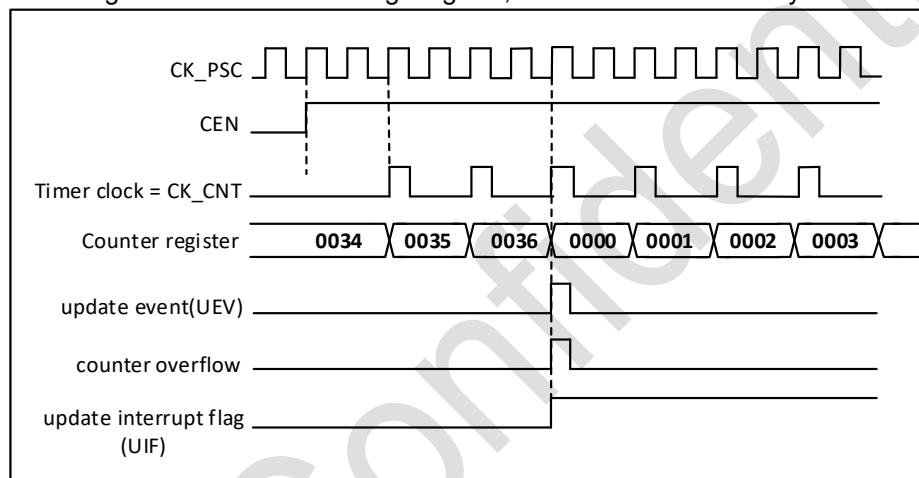


Figure 20-5 Counter timing diagram, internal clock divided by 2

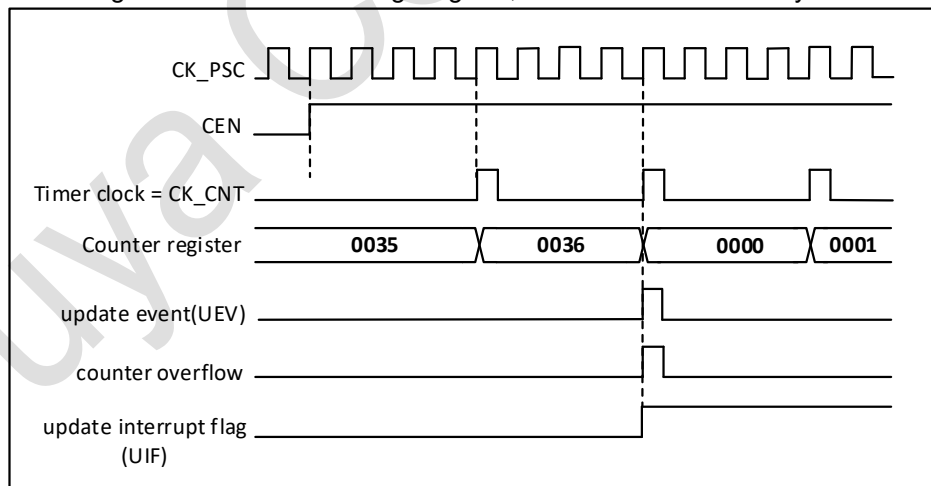


Figure 20-6 Counter timing diagram, internal clock divided by 4

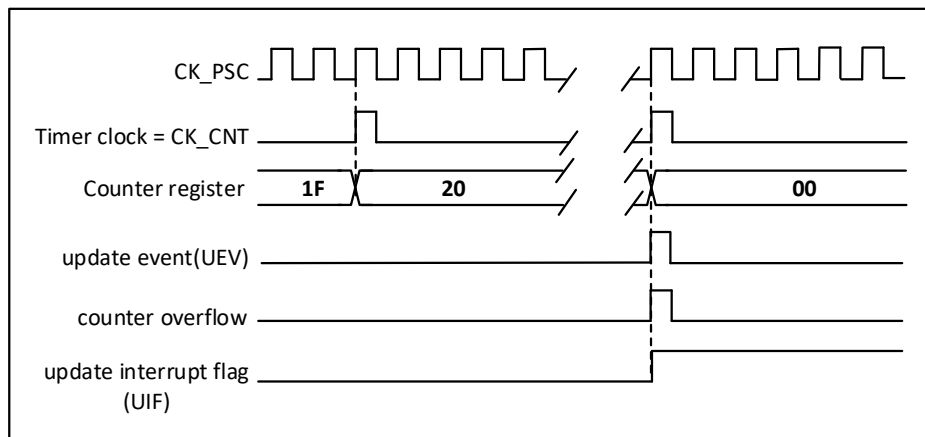


Figure 20-7 Counter timing diagram, internal clock divided by N

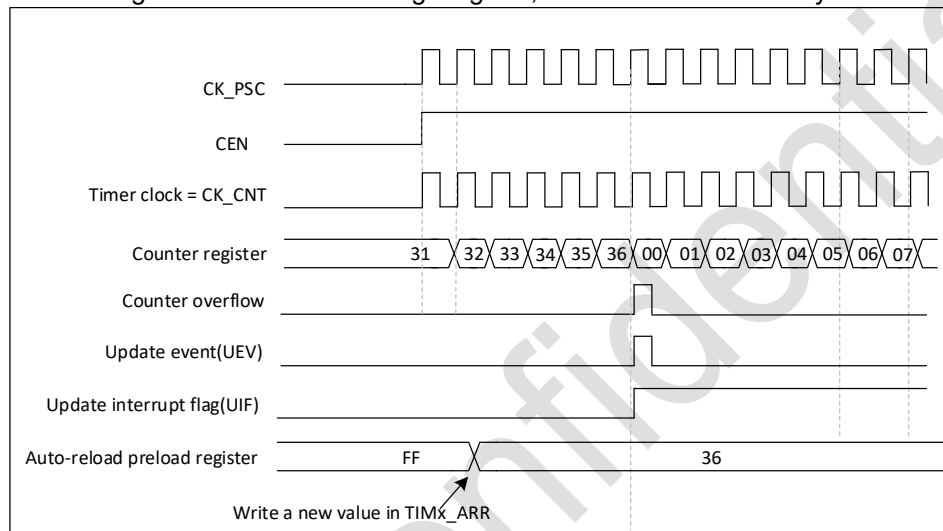


Figure 20-8 Counter timing diagram, update event when ARPE=0 (TIM15_ARR not preloaded)

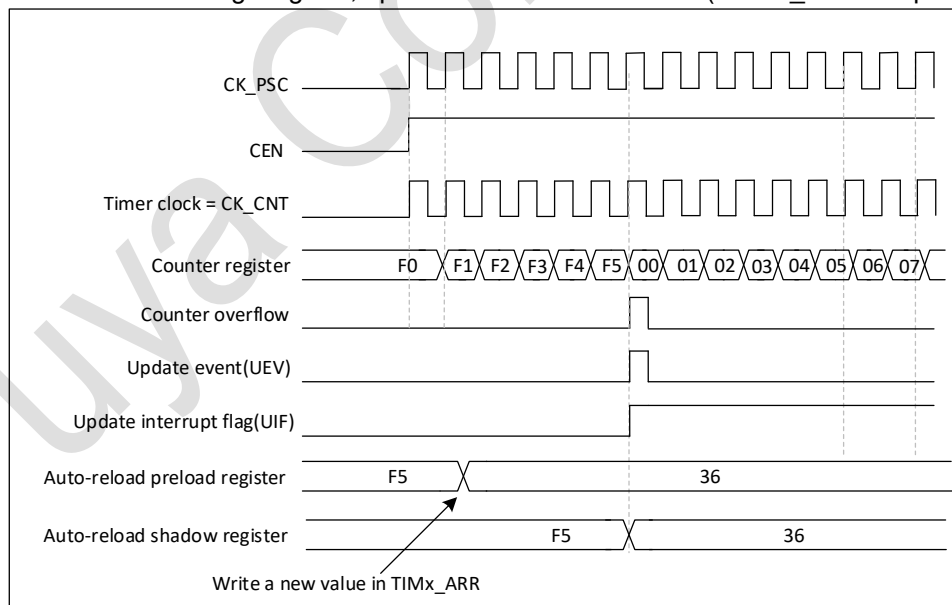


Figure 20-9 Counter timing diagram, update event when ARPE=1 (TIM1_ARR preloaded)

20.2.4. Repetition counter

The time base unit describes how the update event (UEV) is generated when the counter overflows, but in reality, it can only be generated when the repetition counter reaches 0. This feature is useful for generating PWM signals.

This means that on every Nth counter overflow (where N is the value in the TIM15_RCR repetition counter register), data is transferred from the preload register to the shadow register (TIM15_ARR

auto-reload register, TIM15_PSC prescaler register, and the capture/compare register TIM15_CCRx in compare mode).

The repetition counter decrements on each counter overflow in increment counter mode.

The repetition counter is auto-reloaded, and the repetition rate is defined by the value in the TIM15_RCR register. When an update event is generated by software (by setting the UG bit in TIM15_EGR) or by the hardware slave mode controller, the update event occurs immediately regardless of the repetition counter's value, and the content of the TIM15_RCR register is reloaded into the repetition counter.

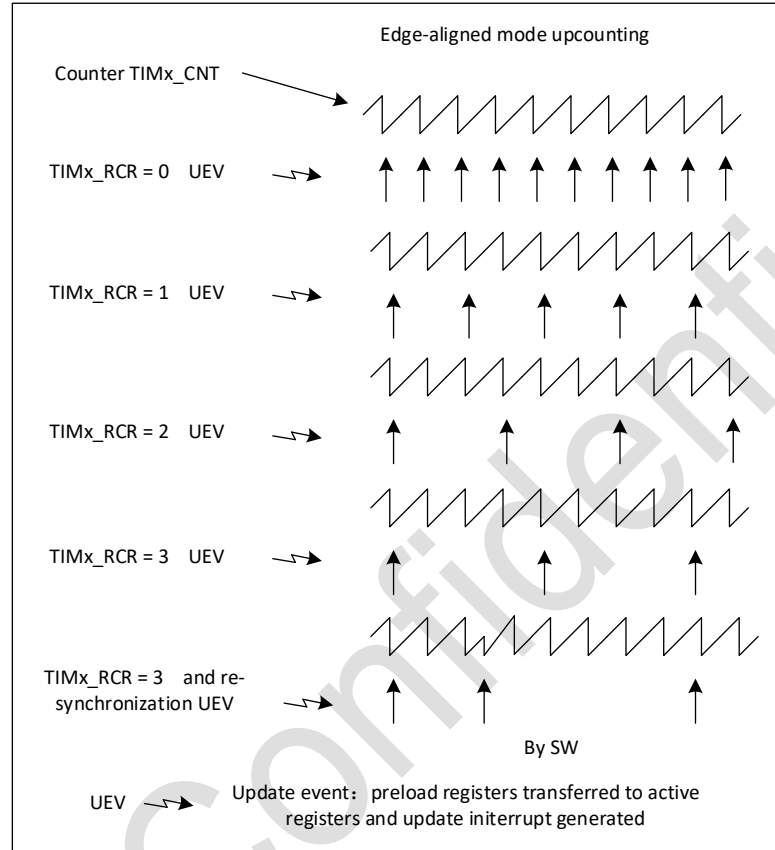


Figure 20-10 Example of update frequency under different modes and TIM15_RCR register settings

20.2.5. Clock sources

20.2.5.1. Internal clock source (CK_INT)

The counter clock is provided by the Internal clock (CK_INT) source. The CEN bit in the TIM15_CR1 register and the UG bit in the TIM15_EGR register are the actual control bits and can only be modified by software (except for the UG bit, which is automatically cleared). Once the CEN bit is set to 1, the prescaler's clock is provided by the internal clock CK_INT.

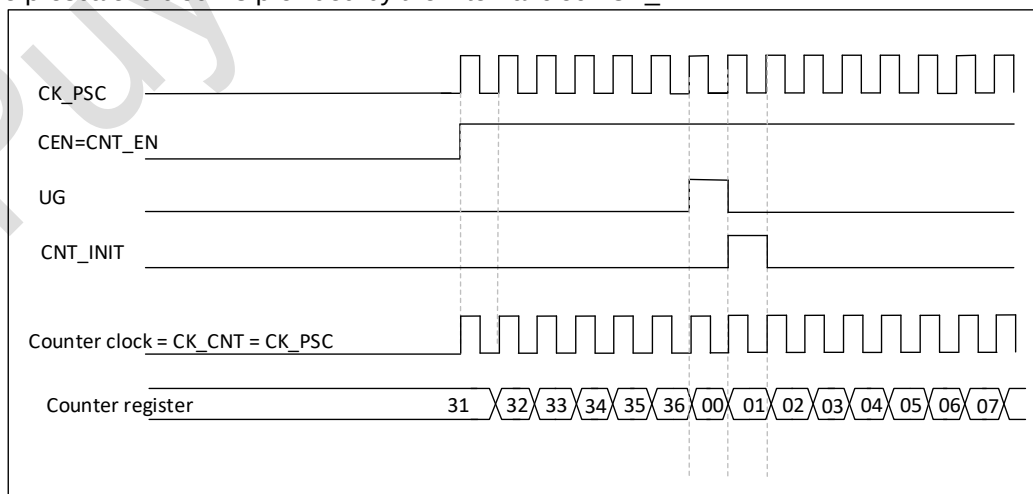


Figure 20-11 Control circuit in normal mode, internal clock divided by 1

20.2.5.2. External clock mode 1

This mode is selected when SMS=111 in the TIM15_SMCR register. The counter can count at each rising or falling edge on a selected input.

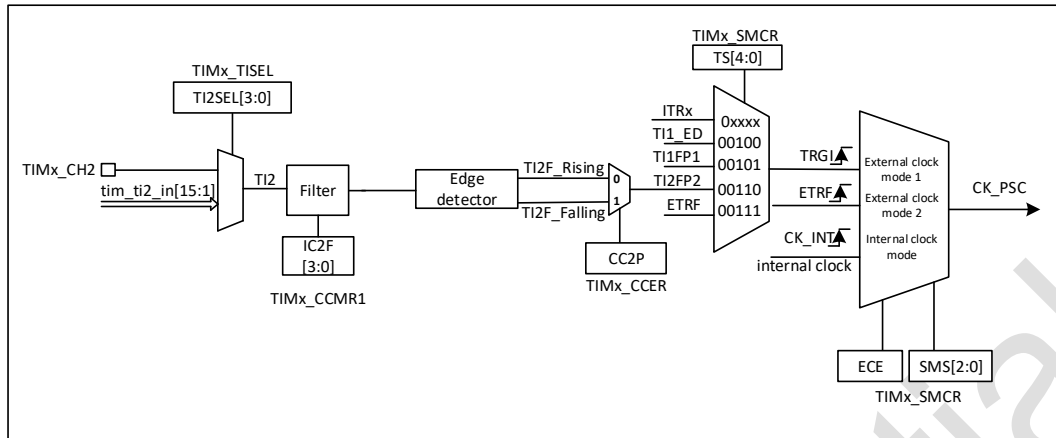


Figure 20-12 TI2 external clock connection example

For example, to configure the upcounter to count in response to a rising edge on the TI2 input, use the following procedure:

1. Select the correct TI2x source (internal or external) by configuring the TI2SEL[3:0] bits in the TIM15_TISEL register.
2. Configure TIM15_CCMR1 register CC2S=01 to detect the rising edge of the TI2 input on channel 2.
3. Configure IC2F[3:0] of the TIM15_CCMR1 register to select the input filter bandwidth (if no filter is needed, keep IC2F=0000)
4. Configure CC2P=0 and CC2NP=0 in the TIM15_CCER register to select the rising edge polarity
5. Configure SMS=111 in the TIM15_SMCR register to select timer external clock mode 1
6. Configure TS=00110 in the TIM15_SMCR register to select TI2 as the trigger input source
7. Set CEN=1 in the TIM15_CR1 register to start the counter

Note: Since the capture prescaler is not used as a trigger, it does not need to be configured.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge of TI2 and the actual counter clock is caused by the resynchronization circuit of the TI2 input.

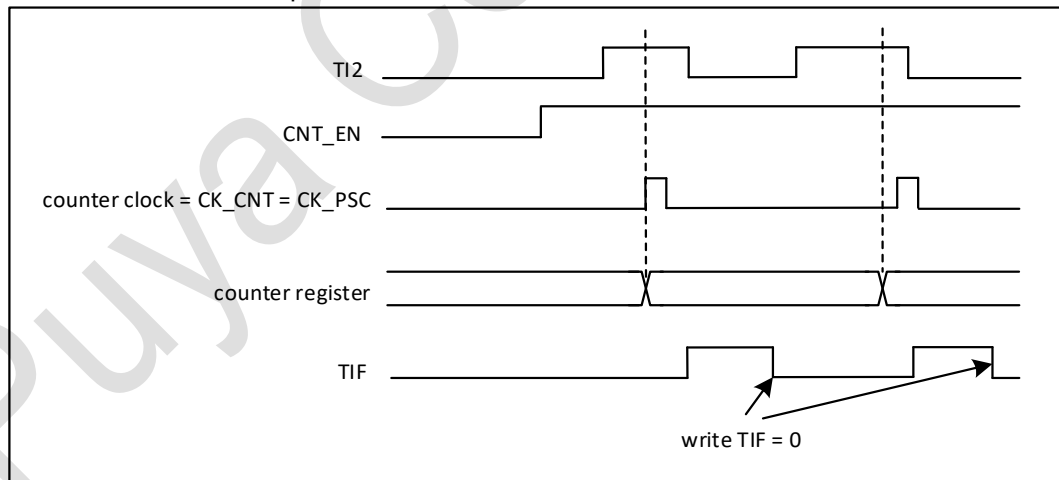


Figure 20-13 Timing diagram of external clock mode 1 control circuit

20.2.5.3. External Clock Mode 2

Select this mode by writing ECE=1 in the TIM15_SMCR register. The counter can count at each rising or falling edge on the external trigger input ETR.

The following figure is a block diagram of the external trigger input:

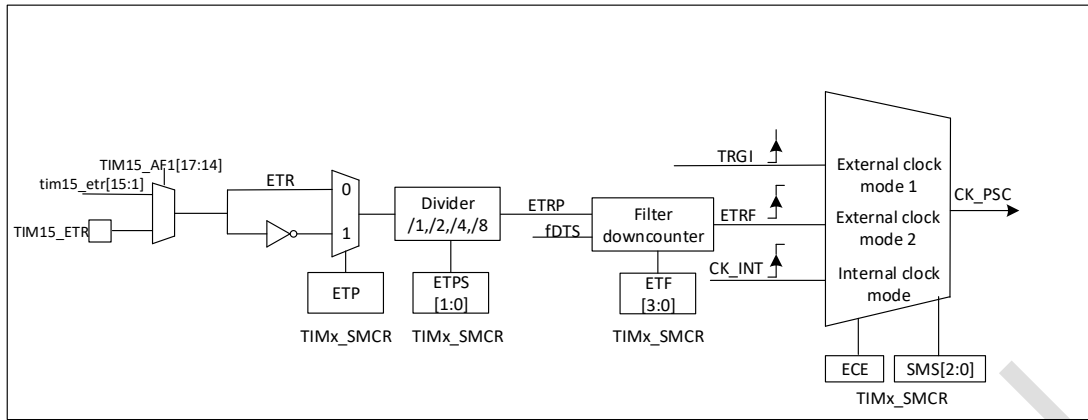


Figure 20-14 External Trigger Input ETR Connection

For example, to configure the upcounter to count each 2 rising edges on ETR, use the following procedure:

1. Since no filter is needed in this example, write ETF[3:0]=0000 in the TIM15_SMCR register.
2. Set the prescaler by writing ETPS[1:0]=01 in the TIM15_SMCR register.
3. Select rising edge detection on the ETR pin by writing ETP=0 in the TIM15_SMCR register.
4. Enable external clock mode 2 by writing ECE=1 in the TIM15_SMCR register.
5. Enable the counter by writing CEN=1 in the TIM15_CR1 register.

The counter counts once each 2 ETR rising edges.

The delay between the rising edge on ETR and the actual clock of the counter is due to the resynchronization circuit on the ETRP signal.

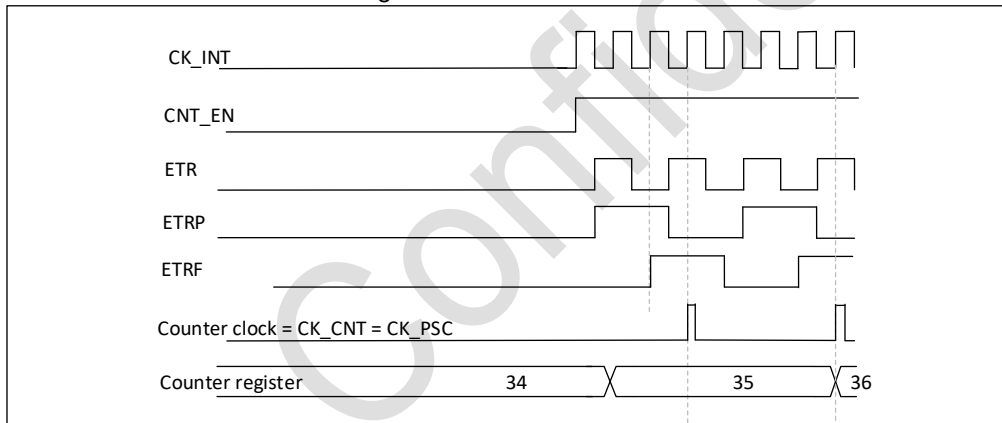


Figure 20-15 External Clock Mode 2 Timing

20.2.6. Capture/Compare channels

Each capture/compare channel is built around a capture/compare register (including a shadow register), a capture input section (digital filter, multiplexer, and prescaler), and an output section (comparator and output control).

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

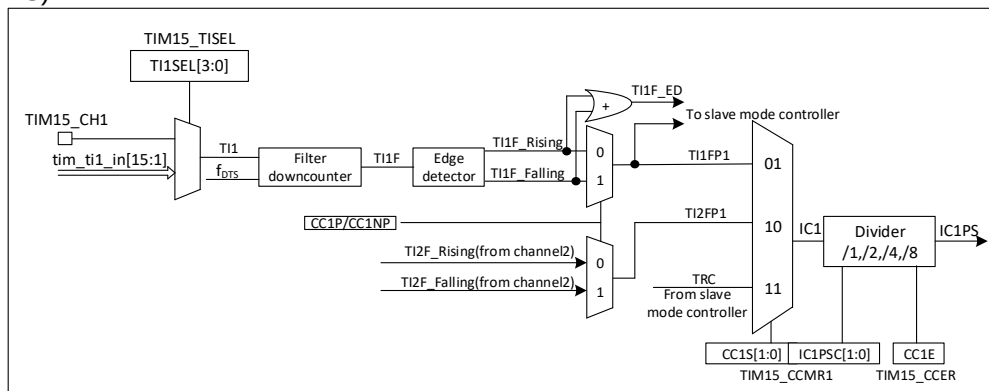
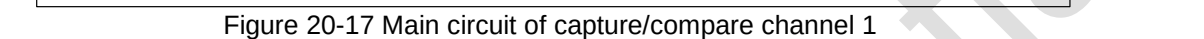
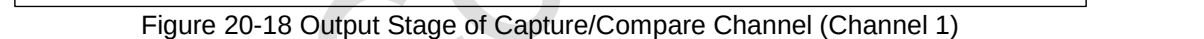


Figure 20-16 Capture/compare channel (example: channel 1 input stage)



0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99



0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

Figure 1. The effect of the number of trials on the number of correct responses.

Figure 1. The effect of the number of trials on the number of correct responses. The number of correct responses was plotted against the number of trials for each condition. The number of correct responses increased with the number of trials for all conditions. The number of correct responses was highest for the condition with the highest number of trials (10 trials) and lowest for the condition with the lowest number of trials (2 trials).

- Figure 1. The model for the effect of the number of children on the number of children in the next generation. The model is based on the idea that the number of children in the next generation is determined by the number of children in the current generation, the number of children in the previous generation, and the number of children in the previous generation's previous generation. The model is based on the idea that the number of children in the next generation is determined by the number of children in the current generation, the number of children in the previous generation, and the number of children in the previous generation's previous generation.

consecutive samples with a new level (at the f_{DTS} frequency), a transition edge on TI1 can be confirmed, i.e., write IC1F=0011 in the TIM15_CCMR1 register.

- Select the active transition edge for the TI1 channel by writing CC1P=0 and CC1NP=0 (rising edge) in the TIM15_CCER register.
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PSC bits to '00' in the TIM15_CCMR1 register).
- Set CC1E=1 in the TIM15_CCER register to allow capturing the counter value into the capture register.
- If needed, enable the corresponding interrupt request by setting the CC1IE bit in the TIM15_DIER register, and enable the DMA request by setting the CC1DE bit in the TIM15_DIER register.

When an input capture occurs:

- The TIM15_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- If the CC1DE bit is set, a DMA request will be generated.

To handle interrupt overflow, it is recommended to read the data before reading the capture overflow flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: Setting the corresponding CCxG bit in the TIM15_EGR register can generate an input capture interrupt and/or DMA request via software.

20.2.8. PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.

For example, you can measure the period (in TIM15_CCR1 register) and the duty cycle (in TIM15_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

- Use the TI1SEL[3:0] bits in the TIM15_TISEL register to select the correct TI1x source (internal or external).
- Select the active input for TIM15_CCR1: write the CC1S bits to 01 in the TIM15_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIM15_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIM15_CCR2: write the CC2S bits to 10 in the TIM15_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FPx (capturing data to TIM15_CCR2): set CC2P=1, CC2NP=0 (falling edge active).
- Select the valid trigger input signal: set TS=00101 in the TIM15_SMCR register (selecting TI1FP1).
- Configure the slave mode controller to reset mode: set SMS=100 in the TIM15_SMCR register.
- Enable capture: set CC1E=1 and CC2E=1 in the TIM15_CCER register.

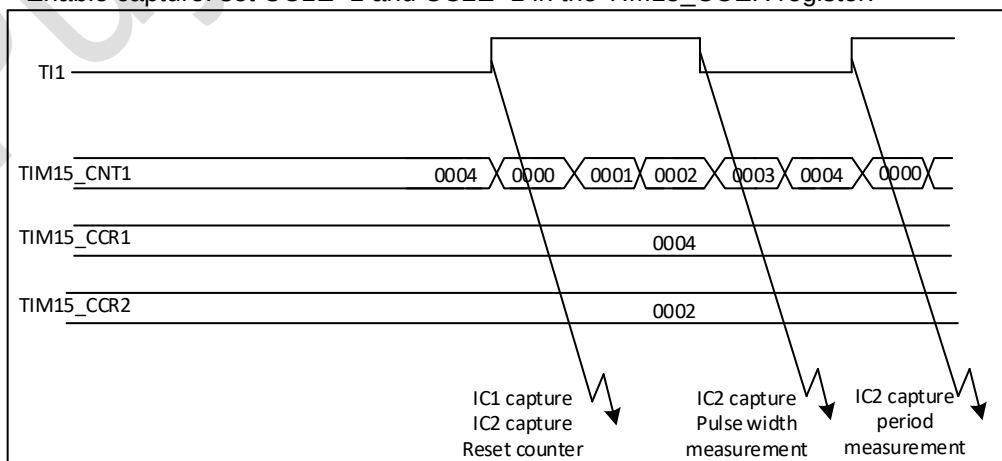


Figure 20-19 PWM input mode timing

20.2.9. Forced Output Mode

In output mode (CCxS=00 in the TIM15_CCMRx register), the output compare signal (OCxREF and the corresponding OCx/OCxN) can be forcibly set to active or inactive by software, independent of the comparison result between the output compare register and the counter. Set the corresponding OCxM=0101 in the TIM15_CCMRx register to forcibly set the output compare signal (OCxREF/OCx) to the active state. Thus OCxREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level. The OCxREF signal can be forced low by writing the OCxM bits to 0100 in the TIM15_CCMRx register.

Anyway, the comparison between the TIM15_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

20.2.10. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed. When a match is found between the capture/compare register and the counter, the output compare function:

- Output the value defined by the output compare mode (OCxM bits in the TIM15_CCMRx register) and output polarity (CCxP bits in the TIM15_CCER register) to the corresponding pin. The output pin can keep its level (OCxM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Set the flag bit in the interrupt status register (CCxIF bit in the TIM15_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIM15_DIER register).
- If the corresponding DMA enable bit is set (CCxDE bit in the TIM15_DIER register, with the CCDS bit in the TIM1_CR2 register selecting the DMA request function), a DMA request will be generated.

The TIM15_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM15_CCMRx register. In output compare mode, the update event UEV has no effect on OCxREF and OCx output.

The timing accuracy can reach one counting cycle of the counter. The output compare mode can also be used to output a single pulse (in one-pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the corresponding data into the TIM15_ARR and TIM15_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:
 - Set OCxM=0011, then the output pin OCx will toggle when the counter matches CCRx
 - Write OCxPE = 0 to disable preload register
 - Write CCxP = 0 to select active high polarity
 - Write CCxE = 1 to enable the output
5. Enable the counter by setting the CEN bit in the TIM15_CR1 register.

The TIM15_CCRx register can be updated via software at any time to control the output waveform, provided the preload register is not used (OCxPE='0'; otherwise, the shadow register of TIM15_CCRx can only be updated upon the next update event). An example is given in the figure below.

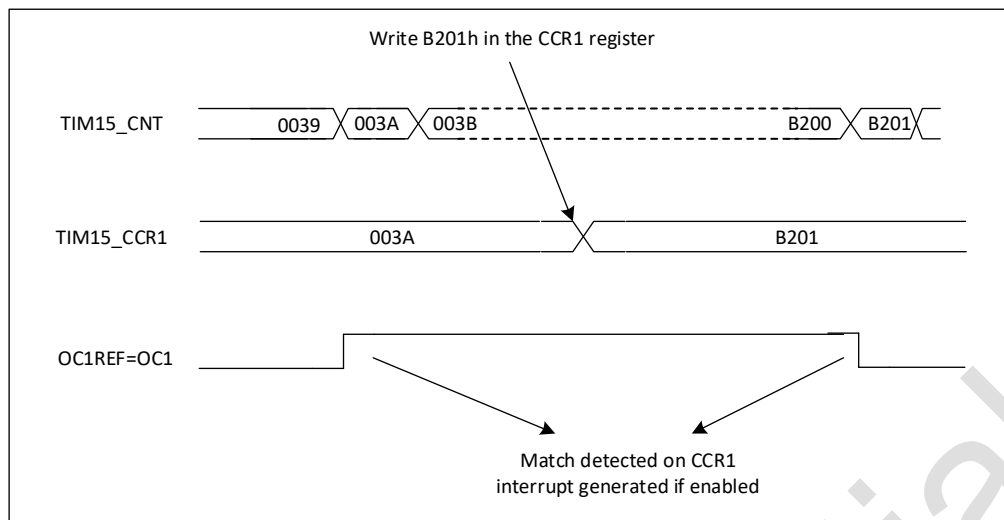


Figure 20-20 Output Compare Mode, Toggling OC1

20.2.11. PWM mode

The pulse width modulation mode can generate a signal with a frequency determined by the TIM15_ARR register and a duty cycle determined by the TIM15_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bits in the TIM15_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM15_CCMRx register, and finally, the ARPE bit in the TIM15_CR1 register must be set to 1 to enable the auto-reload preload register (in up-counting or center-aligned mode).

Since the preload register can only be transferred to the shadow register upon an update event, all registers must be initialized by setting the UG bit in the TIM15_EGR register before the counter starts counting.

OCx polarity is software programmable using the CCxP bit in the TIM15_CCER register. It can be programmed as active high or active low. The OCx output is enabled by the CCxE bit in the TIM15_CCER register.

In PWM mode (mode 1 or mode 2), TIM15_CNT and TIM15_CCRx are continuously compared to determine whether $TIM15\_CNT \leq TIM15\_CCRx$.

The timer can generate edge-aligned PWM when the counter is counting up.

20.2.11.1. PWM edge-aligned mode

In the following example, we consider PWM mode 1. When $TIM15\_CNT < TIM15\_CCRx$, the PWM reference signal OCxREF is high; otherwise, it is low. If the compare value in TIM15_CCRx is greater than the auto-reload value (in TIM15_ARR) then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'. The figure below shows an example of edge-aligned PWM waveform when $TIM15\_ARR=8$.

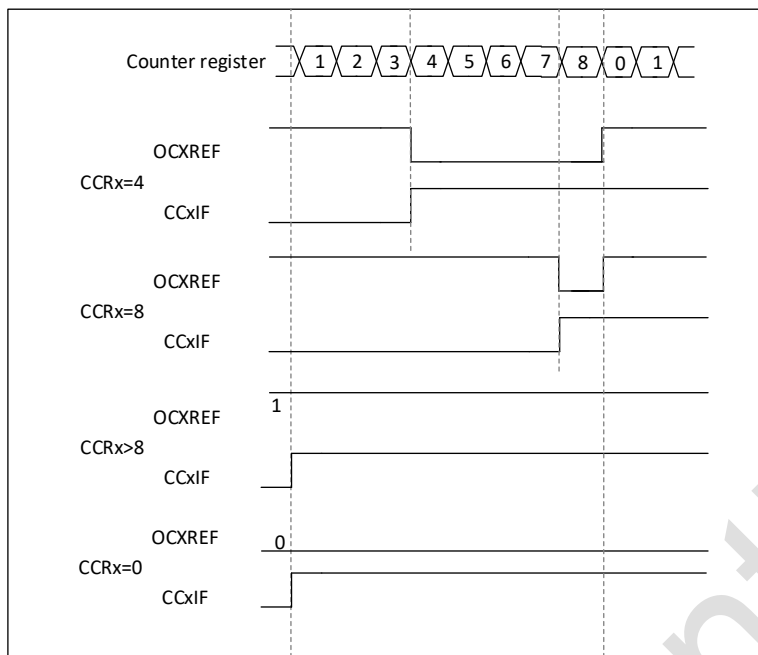


Figure 20-21 Edge-aligned PWM output waveform (ARR=8)

20.2.12. Complementary outputs and dead-time insertion

TIM15 can output a pair of complementary signals and manage the instantaneous turn-off and turn-on of the outputs. This period is typically called the dead-time, and users must adjust the dead-time based on the connected output devices and their characteristics (level transition delay, power switch delay, etc.).

By configuring the CCxP and CCxNP bits in the TIM15_CCER register, the polarity can be independently selected for each output (main output OCx or complementary output OCxN).

The complementary signals OCx and OCxN are controlled by the combination of the following control bits: CCxE and CCxNE bits in the TIM15_CCER register, and MOE, OISx, OISxN, OSS1, and OSSR bits in the TIM15_BDTR and TIM15_CR2 registers. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. There is one 8-bit dead-time generator for each channel. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal except for the rising edge, which is delayed relative to the reference rising edge.
- The OCxN output signal is the opposite of the reference signal except for the rising edge, which is delayed relative to the reference falling edge.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose CCxP=0, CCxNP=0, MOE=1, CCxE=1 and CCxNE=1 in these examples)

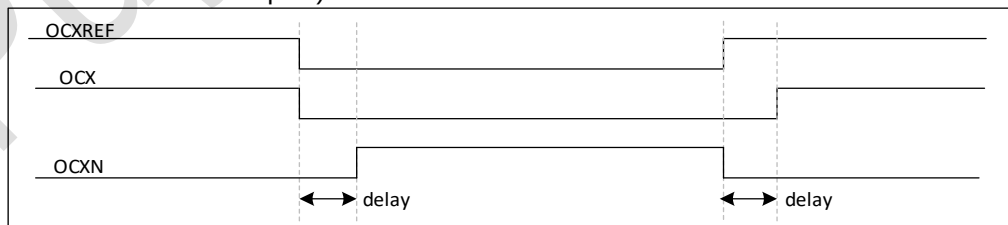


Figure 20-22 Complementary output with dead-time insertion

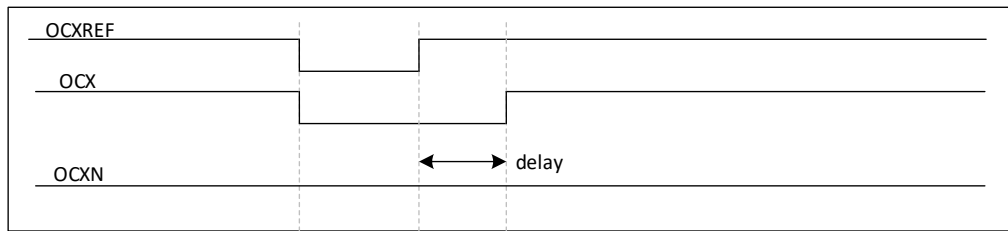


Figure 20-23 Dead-time waveform where the delay time is greater than the negative pulse width

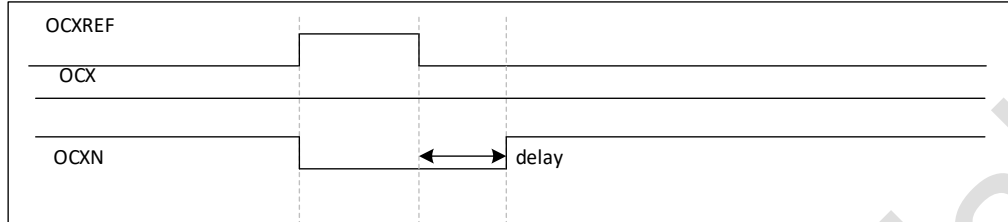


Figure 20-24 Dead-time waveform where the delay time is greater than the positive pulse width

The dead-time duration is identical for each channel and is programmed through the DTG bits in the TIM15_BDTR register.

20.2.12.1. Redirect OCxREF to OCx or OCxN

In output modes (forced mode, output compare mode, or PWM mode), OCxREF can be redirected to the OCx or OCxN output by configuring the CCxE and CCxNE bits in the TIM15_CCER register.

This feature allows sending a special waveform (e.g., PWM or static active level) on one output while keeping the complementary output at an inactive level when the complementary output is inactive. Alternatively, both outputs can be kept at inactive levels simultaneously, or both can be kept at active levels with complementary outputs and dead-time.

Note: When only OCxN is enabled (CCxE=0, CCxNE=1), the two are not complementary, and OCxN immediately goes high when OCxREF is valid. For example, if CCxNP=0 then OCxN=OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE=CCxNE=1), OCx is active when OCxREF is high, while OCxN is active when OCxREF is low (inverted behavior).

20.2.13. Using the break function

The purpose of the break function is to protect power switches driven by PWM signals generated with the TIM15 timer. The two break inputs are usually connected to fault outputs of power stages and 3-phase inverters. When activated, the break circuitry shuts down the PWM outputs and forces them to a predefined safe state. A number of internal MCU events can also be selected to trigger an output shut-down.

A break channel which gathers both system-level fault (clock failure, parity error,...) and application fault (from input pins and built-in comparator), and can force the outputs to a predefined level (either active or inactive) after a deadtime duration.

The output enable signal and output levels during break are depending on several control bits:

- The MOE bit in the TIM15_BDTR register allows the outputs to be enabled/disabled by software and is reset upon a break event.
- The OSSI bit in the TIM15_BDTR register defines whether the timer controls the outputs in an inactive state or releases control to the GPIO controller (typically placing it in high-impedance mode).
- The OISx and OISxN bits in the TIM15_CR2 register set the output to the shutdown level (active or inactive). Regardless of the values of OISx and OISxN, it is not possible to set both OCx and OCxN outputs to an active level simultaneously at a given time.

When exiting from reset, the break circuit is disabled and the MOE bit is low. The break function is enabled by setting the BKE bit in the TIM15_BDTR register. The polarity of the brake input can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time.

Because the falling edge of MOE can be asynchronous, a resynchronization circuit is implemented between the actual signal (acting on the output) and the synchronous control bit (in the TIM1_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if you write MOE to 1 whereas it was low, you must insert a delay (dummy instruction) before reading it correctly. This is because the write acts on the asynchronous signal whereas the read reflects the synchronous signal.

The source for break (BRK) channel can be an external source connected to the BKIN pin or one of the following internal sources:

- the CPU LOCKUP output
- the PVD output
- a clock failure event generated by the CSS detector
- the output from a comparator
- SRAM byte parity error
- FLASH ECC error

A break event can also be generated by software through the BG bit in the TIM15_EGR register. Regardless of the value of the BKE enable bit, a brake can be generated via software using BG.

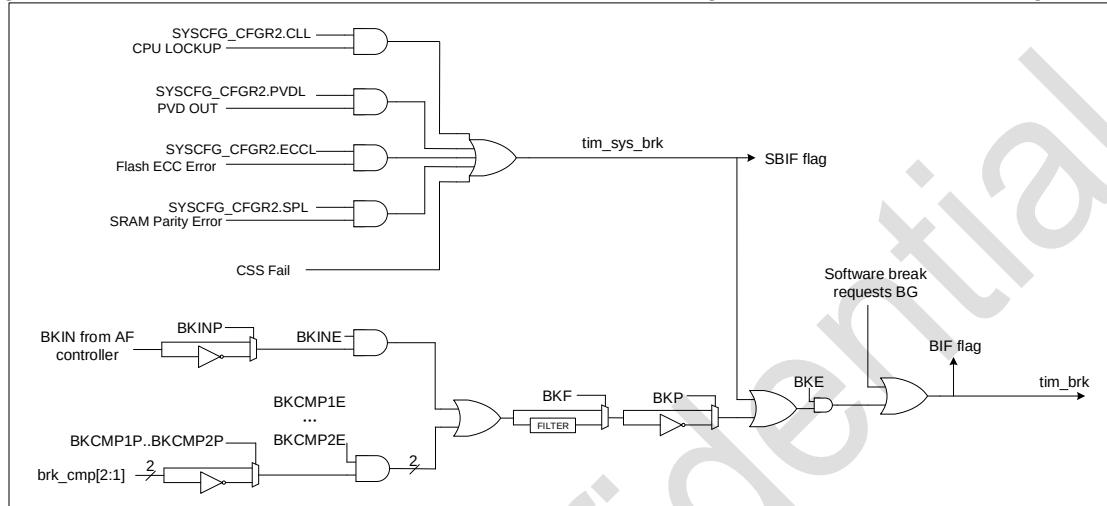


Figure 20-25 Brake circuit diagram

When a break occurs (selected level on the break input):

- The MOE bit is asynchronously cleared, placing the output in an inactive state, idle state, or releasing control to the GPIO (selected by the OSS1 bit). This feature remains effective even when the MCU's clock oscillator is turned off.
- Once MOE=0, each output channel will be driven to the level set by the OISx bits in the TIM15_CR2 register. If OSS1=0, the timer releases output control (taken over by the GPIO controller); otherwise, the output is enabled to remain high.
- When complementary outputs are used:
 - The outputs are first put in reset state inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
 - If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note: Due to MOE resynchronization, the dead time is slightly longer than usual (approximately 2 timer clock cycles).
 - If OSS1=0, the timer releases the enabled output (taken over by the GPIO controller forcing a high-impedance state); otherwise, the enabled output remains high or immediately goes high if either the CCxE or CCxNE bit is high.
- If the BIE bit in the TIM15_DIER register is set, an interrupt will be generated when the brake status flags (SBIF and BIF bits in the TIM15_SR register) are '1'.
- If the AOE bit in the TIM15_BDTR register is set, the MOE bit will be automatically set again at the next update event (UEV); for example, this can be used for shaping. Otherwise, MOE remains low until the application sets it to '1' again. In this case, this feature ensures safety by allowing the brake input to be connected to a power-driven alarm output, thermal sensor, or other safety device.

Note: The break inputs are active on level. Therefore, when the brake input is active, the MOE bit cannot be set to 1 (automatically or via software). In the meantime, the status flag BIF can not be cleared.

The brake can be generated by the BRK input, whose active polarity is programmable and enabled by the BKE bit in the TIM15_BDTR register.

There are two ways to generate a brake:

- Through the BKR input with programmable polarity, while enabling BKE in the TIM15_BDTR register
- By software setting the BG bit in TIM15_EGR.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). Users can select one of three protection levels via the LOCK bit in the TIM15_BDTR register. The LOCK bits can be written only once after an MCU reset.

The figure below shows an example of behavior of the outputs in response to a break.

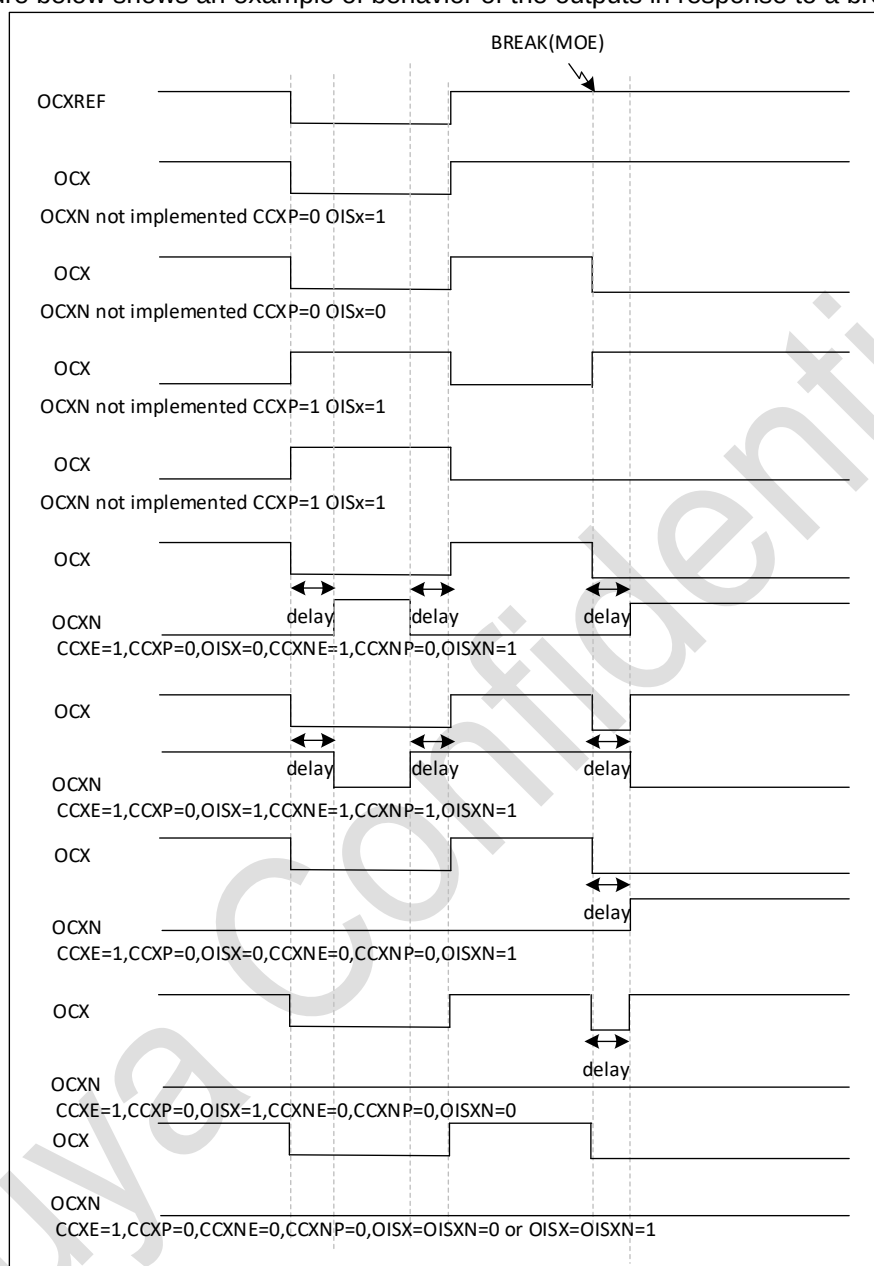


Figure 20-26 Different output behaviors in response to a brake event

20.2.14. One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. One-pulse mode is selected by setting the OPM bit in the TIM15_CR1 register. This makes the counter stop automatically at the next update event UEV.

A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- Up-counting mode: Counter $CNT < CCRx \leq ARR$ (specifically, $0 < CCRx$)

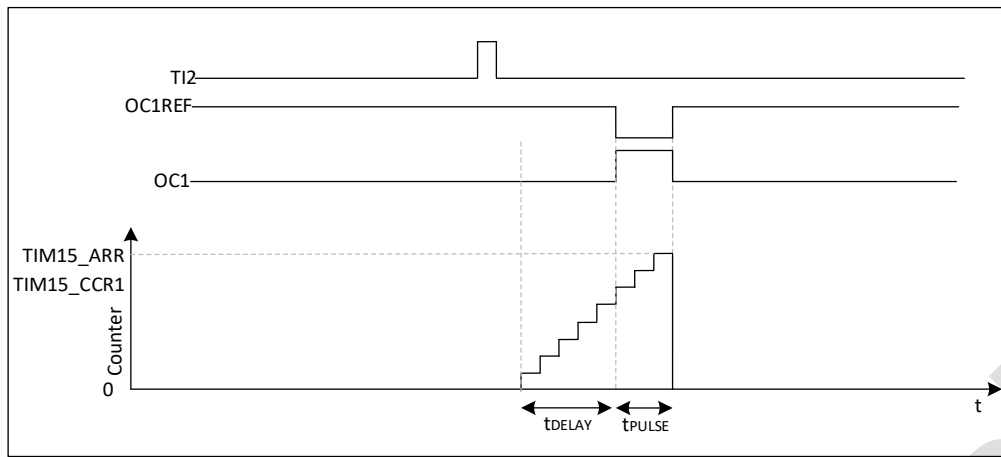


Figure 20-27 Example of One Pulse Mode

For example one may want to generate a positive pulse on OC1 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Using TI2FP2 as the trigger source:

- Select the correct TI2x source using the TI2SEL[3:0] bits in the TIM15_TISEL register
- Map TI2FP2 on TI2 by writing CC2S=01 in the TIM15_CCMR1 register.
- Setting CC2P=0 and CC2NP=0 in the TIM15_CCER register enables TI2FP2 to detect rising edges.
- Set TS=00110 in the TIM15_SMCR register to use TI2FP2 as the trigger (TRGI) for the slave mode controller.
- Set SMS=110 (trigger mode) in the TIM15_SMCR register to use TI2FP2 to start the counter.

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The t_{DELAY} is defined by the value written in the TIM15_CCR1 register.
- t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIM15_ARR – TIM15_CCR1).
- Assuming a waveform transition from 0 to 1 is to be generated upon compare match and a transition from 1 to 0 when the counter reaches the preload value; first, set OC1M=0111 in the TIM15_CCMR1 register to enter PWM mode 2; optionally enable the preload register as needed: set OC1PE=1 in TIM15_CCMR1 and ARPE in the TIM15_CR1 register; then fill the compare value in the TIM15_CCR1 register and the auto-reload value in the TIM15_ARR register, set the UG bit to generate an update event, and wait for an external trigger event on TI2. CC1P is written to '0' in this example.

In our example, the DIR and CMS bits in the TIM15_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIM15_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0) When OPM=0 in the TIM15_CR1 register, the repetition mode is selected.

20.2.15. Timer and External Trigger Synchronization

The TIM15 timer can synchronize with an external trigger in various modes: reset mode, gated mode, and trigger mode.

20.2.15.1. Slave mode: Reset mode

Upon a trigger input event, the counter and its prescaler can be reinitialized; at the same time, if the URS bit in the TIM15_CR1 register is low, an update event UEV is also generated; then all preload registers (TIM15_ARR, TIM15_CCRx) will be updated.

In the following example, the rising edge on the TI1 input causes the up-counter to be cleared:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). Since the capture prescaler is not used for trigger operations, no configuration is required. The CC1S bits select the input capture source only, CC1S=01 in TIM15_CCMR1 register. Set CC1P=0 (and CC1NP=0) in the TIM15_CCER register to determine the polarity (only detect rising edges).
- Set SMS=100 in the TIM15_SMCR register to configure the timer in reset mode; set TS=00101 in the TIM15_SMCR register to select TI1 as the input source.
- Enable the counter by writing CEN=1 in the TIM15_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. At the same time, the trigger flag (TIF bit in the

TIM15_SR register) is set, and depending on the settings of the TIE (interrupt enable) bit and TDE (DMA enable) bit in the TIM15_DIER register, an interrupt request or a DMA request is generated. The figure below shows the operation when the auto-reload register TIM15_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

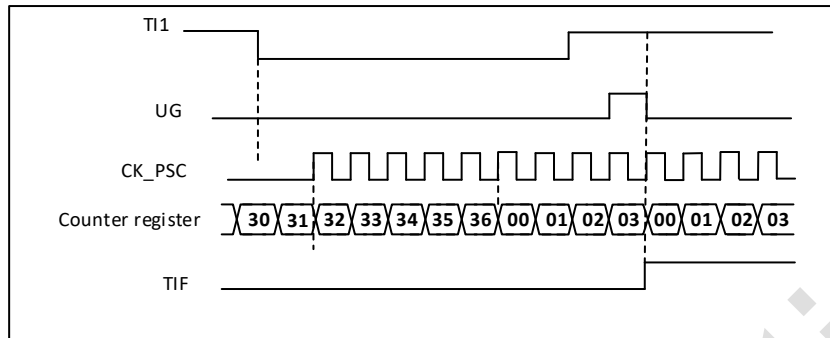


Figure 20-28 Control circuit in reset mode

20.2.15.2. Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the counter increments only when TI1 is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIM15_CCMR1 register. Set CC1P=1 (and CC1NP=0) in the TIM15_CCER register to determine the polarity (only detect low level).
- Set SMS=101 in the TIM15_SMCR register to configure the timer in gated mode; set TS=00101 in the TIM15_SMCR register to select TI1 as the input source.
- Enable the counter by writing CEN=1 in the TIM15_CR1 register. In gated mode, if CEN=0, the counter cannot start regardless of the trigger input level.

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in TIM15_SR is set when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

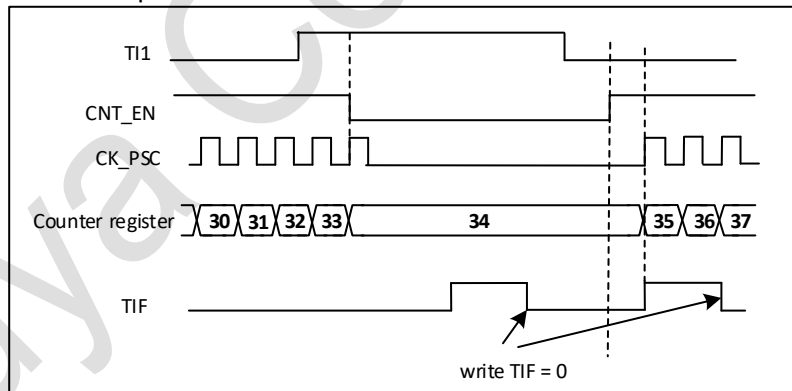


Figure 20-29 Control circuit in Gated mode

20.2.15.3. Slave mode: Trigger mode

The selected event on the input enables the counter.

In the following example, the counter starts incrementing on the rising edge of the TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we do not need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIM15_CCMR1 register. Set CC2P=1 (and CC2NP=0) in the TIM15_CCER register to determine the polarity (only detects low level).
- Set SMS=110 in the TIM15_SMCR register to configure the timer in trigger mode; set TS=00110 in the TIM15_SMCR register to select TI2 as the input source.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set. The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

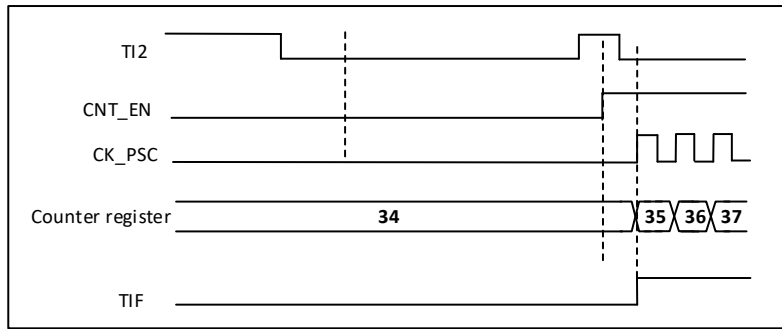


Figure 20-30 Control circuit in trigger mode

20.2.15.4. Slave mode: external clock mode 2 + trigger mode

External clock mode 2 can be used together with another slave mode (except external clock mode 1). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input when operating in reset mode, gated mode or trigger mode. It is not recommended to use the TS bit of the TIM15_SMCR register to select ETR as TRGI.

In the following example, once a rising edge appears on TI1, the counter increments once at each rising edge of ETR:

- Configure the external trigger input circuit through the TIM15_SMCR register:
 - ETF = 0000: no filter
 - ETPS = 00: prescaler disabled
 - ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.
- Configure the channel 1 as follows, to detect rising edges on TI1:
 - IC1F = 0000: no filter
 - The capture prescaler is not used for triggering, so it does not need to be configured.
 - CC1S=01 in TIM15_CCMR1 register to select only the input capture source
 - Set CC1P=0 in the TIM15_CCER register to determine the polarity (only detects rising edges).
- Set SMS=110 in the TIM15_SMCR register to configure the timer in trigger mode. Set TS=00101 in the TIM15_SMCR register to select TI1 as the input source.

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges. The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

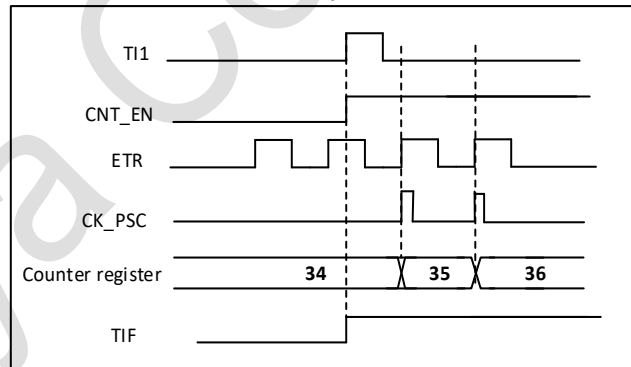


Figure 20-31 Control circuit in external clock mode 2 + trigger mode

20.2.16. Timer synchronization

Different timers are internally connected for timer synchronization or linking functions. When one Timer is configured in Master mode, it can reset, start, stop or clock the counter of another Timer configured in Slave mode.

For detailed descriptions, refer to the relevant sections in TIM2/TIM3.

20.3. TIM15 Interrupt

Table 20-1 TIM15 Interrupt

Interrupt event	Event flag	Interrupt enable control bit
update	UIF	UIE
Compare/Capture 1	CC1IF	CC1IE
Compare/Capture 2	CC2IF	CC2IE

COM	COMIF	COMIE
Trigger	TIF	TIE
Brake	BIF	BIE

20.4. TIM15 Debug Mode

When the chip enters debug mode, the TIM15 counter can continue to operate normally or stop working according to the setting of DBG_TIM15_STOP in the DBG module.

20.5. TIM15 registers

20.5.1. TIM15 control register 1 (TIM15_CR1)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
						RW	RW	RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	2'b0	Clock division factor. These 2 bits define the division ratio between the timer clock (CK_INT) frequency and the dead-time generator and sampling clock (t _{DTS}) used by the digital filter (ETR, Tlx). 00: t _{DTS} = t _{CK_INT} 01: t _{DTS} = 2 x t _{CK_INT} 10: t _{DTS} = 4 x t _{CK_INT} 11: reserved, do not use this configuration.
7	ARPE	RW	0	Auto-reload preload enable 0: TIM15_ARR register is not buffered 1: TIM15_ARR register is buffered
6:4	Reserved	-	-	Reserved
3	OPM	RW	0	One-pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled
1	UDIS	RW	0	Update disable Software enables/disables UEV event generation through this bit. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller Shadow registers are loaded with preload values. 1: UEV disabled. The update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). The counter and prescaler are reinitialized if the UG bit is set or a hardware reset is issued by the slave mode controller.
0	CEN	RW	0	Counter enable. 0: Counter disabled 1: Enable counter

				Note: External clock, gated mode, and encoder mode can only be used after software sets the CEN bit. Trigger mode can automatically set the CEN bit to 1 via hardware.

20.5.2. TIM15 control register 2 (TIM15_CR2)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	OIS2	OIS1N	OIS1	Res.	MMS[2:0]			CCDS	CCUS	Res.	CCPC
					RW	RW	RW		RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10	OIS2	RW	0	Output idle state 2 (OC2 output) Define reference OIS1.
9	OIS1N	RW	0	Output idle state 1 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=0 Note: This bit cannot be modified after LOCK (TIM15_BDTR register) levels 1, 2, or 3 have been set.
8	OIS1	RW	0	Output idle state 1 (OC1 output) 0: OC1=0 after a dead-time when MOE=0 1: OC1=0 after a dead-time when MOE=1 Note: This bit cannot be modified after LOCK (TIM15_BDTR register) levels 1, 2, or 3 have been set.
7	Reserved	-	-	Reserved
6:4	MMS	RW	0	Master mode selection. These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 0000: Reset --- The UG bit of the TIM1_EGR register serves as the trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 0001: Enable --- The counter enable signal CNT_EN serves as the trigger output (TRGO). Sometimes it is necessary to start multiple timers simultaneously or control a slave timer for a period of time. When configured in gated mode, the counter enable is generated by the logical AND of the CEN control bit and the trigger input signal. When the counter enable signal is controlled by the trigger input, there will be a delay on TRGO unless master/slave mode is selected (see the description of the MSM bit in the TIM1_SMCR register). 0010: Update --- The update event serves as the trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 0011: Compare pulse --- When a capture or a compare match occurs, a positive pulse (TRGO) is generated at the trigger output when the CC1IF flag is set to 1. 100: Compare --- The OC1REF signal is used as the trigger output (TRGO). 101: Compare --- The OC2REF signal is used as the trigger output (TRGO). Others: Reserved
3	CCDS	RW	0	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
2	CCUS	RW	0	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit only. 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit or when a rising edge occurs on TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved	-	-	Reserved

0	CCPC	RW	0	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded. 1: CCxE, CCxNE, and OCxM bits are preloaded; after setting this bit, these bits are updated only when a commutation event (COM) occurs (COMG bit set to 1 or a rising edge detected on TRGI, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output.

20.5.3. TIM15 slave mode control register (TIM15_SMCR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS[5:4]	Res.	Res.	Res.	Res.	Res.
										RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]			ETF[3:0]			MSM		TS[2:0]		Res.		SMS[2:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	Reserved
21:20	TS[5:4]	RW	000	See TS description for details
19:16	Reserved	-	-	Reserved
15	ETP	RW	0	External trigger polarity This bit selects whether ETR or inverted ETR is used for trigger operations. 0: ETR is not inverted, active high or rising edge 1: ETR inverted, active low or falling edge
14	ECE	RW	0	External clock enable. This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal. Notes: 1: Setting the ECE bit has the same effect as selecting External Clock Mode 1 and connecting TRGI to ETRF (SMS=111 and TS=00111). 2: External Clock Mode 2 can be used simultaneously with the following slave modes: Reset Mode, Gated Mode, and Trigger Mode. However, in such cases, TRGI must not be connected to ETRF (TS bits must not be 00111). 3: If both External Clock Mode 1 and External Clock Mode 2 are enabled, the external clock input is ETRF.
13:12	ETPS[1:0]	RW	00	External trigger prescaler The frequency of the external trigger signal ETR must not exceed 1/4 of the TIM1CLK frequency. The frequency of ETR can be reduced by enabling the prescaler. This method is very useful when inputting a fast external clock. 00: Prescaler OFF 01: ETR frequency divided by 2 10: ETR frequency divided by 4 11: ETR frequency divided by 8
11:8	ETF[3:0]	RW	0000	External trigger filter. This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. In practice, the digital filter is an event counter that records N events before considering a valid output edge. 0000: No filter, sampling is done at f_{DTS} 0001: $f_{SAMPLING} = f_{CK\_INT}$, $N = 2$ 0010: $f_{SAMPLING} = f_{CK\_INT}$, $N = 4$ 0011: $f_{SAMPLING} = f_{CK\_INT}$, $N = 8$ 0100: $f_{SAMPLING} = f_{DTS}/2$, $N = 6$ 0101: $f_{SAMPLING} = f_{DTS}/2$, $N = 8$ 0110: Sampling frequency $f_{SAMPLING} = f_{DTS}/4$, $N=6$ 0111: Sampling frequency $f_{SAMPLING} = f_{DTS}/4$, $N=8$ 1000: Sampling frequency $f_{SAMPLING} = f_{DTS}/8$, $N=6$ 1001: Sampling frequency $f_{SAMPLING} = f_{DTS}/8$, $N=8$ 1010: Sampling frequency $f_{SAMPLING} = f_{DTS}/16$, $N=5$ 1011: Sampling frequency $f_{SAMPLING} = f_{DTS}/16$, $N=6$ 1100: $f_{SAMPLING} = f_{DTS}/16$, $N = 8$ 1101: $f_{SAMPLING} = f_{DTS}/32$, $N = 5$ 1110: $f_{SAMPLING} = f_{DTS}/32$, $N = 6$

				1111: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 8$
7	MSM	RW	0	Master/slave mode. 0: No action 1: The event on the trigger input (TRGI) is delayed to achieve perfect synchronization between the current timer and its slave timer (via TRGO). It is useful if we want to synchronize several timers on a single external event.
6:4	TS[2:0]	RW	0	Trigger selection These 5 bits select the trigger input used to synchronize the counter. 00000: Internal Trigger 0 (ITR0) 00001: Internal Trigger 1 (ITR1) 00010: Internal Trigger 2 (ITR2) 00011: Internal Trigger 3 (ITR3) 00100: TI1 edge detection (TI1F_ED) 00101: Filtered timer input 1 (TI1FP1) 00110: Filtered timer input 2 (TI1FP2) 00111: External trigger input (ETRF) 01000: Internal Trigger 4 (ITR4) 01001: Internal Trigger 5 (ITR5) Others: Reserved Note: These bits can only be changed when unused (e.g., SMS=0000) to avoid false edge detection during transitions. Note: For ITRx mapping, refer to the "Peripheral Interconnection" section.
3	Reserved	-	-	Reserved
2:0	SMS[2:0]	RW	0	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description). 000: Slave mode disabled---If CEN=1, the prescaler is directly clocked by the internal clock. 100: Reset mode---The rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update register signal. 101: Gated mode---The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger mode---The counter starts on the rising edge of the trigger input TRGI (but is not reset), only controlling the counter's start. 111: External clock mode 1---The rising edge of the selected trigger input (TRGI) clocks the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS=00100). This is because TI1F_ED outputs a pulse every time TI1F changes, but the gated mode checks the level of the trigger input.

20.5.4. TIM15 DMA/interrupt enable register (TIM15_DIER)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	COMDE	Res.	Res.	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	Res.	Res.	CC2IE	CC1IE	UIE
	RW	RW			RW	RW	RW	RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
14	TDE	RW	0	Trigger DMA request enable 0: Trigger DMA request disabled 1: Trigger DMA request enabled
13	COMDE	RW	0	COM DMA request enable 0: COM DMA request disabled 1: COM DMA request enabled
12:11	Reserved	-	-	Reserved

10	CC2DE	RW	0	Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled 1: CC2 DMA request enabled
9	CC1DE	RW	0	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled 1: CC1 DMA request enabled
8	UDE	RW	0	Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled
7	BIE	RW	0	Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled
6	TIE	RW	0	Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled
5	COMIE	RW	0	COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4:3	Reserved	-	-	Reserved
2	CC2IE	RW	0	Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled
1	CC1IE	RW	0	Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

20.5.5. TIM15 status register (TIM15_SR)

Offset address: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res.	Res.	Res	Res.	Res.	Res.	Res	Res	Res.	Res.	Res.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	CC2OF	CC1OF	Res	BIF	TIF	COMIF	Res	Res	CC2IF	CC1IF	UIF
.	.	.	.	.	RC_W0	RC_W0	.	RC_W0	RC_W0	RC_W0	.	.	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10	CC2OF	RC_W0	0	Capture/compare 2 overcapture flag. Refer to CC1OF description
9	CC1OF	RC_W0	0	Capture/compare 1 overcapture flag. This flag can be set to 1 by hardware only when the corresponding channel is configured as input capture. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: When CC1IF is set to 1, the TIM15_CCR1 register has captured the counter value and CC1IF has been set.
8	Reserved	-	-	Reserved
7	BIF	RC_W0	0	Brake interrupt flag. This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input. An interrupt will be generated if BIE=1 in the TIM1_DIER register.
6	TIF	RC_W0	0	Trigger interrupt flag. In all modes except gated mode, this bit is set to 1 by hardware when a valid edge is detected at the TRGI input after enabling the slave mode controller. In gated mode, this flag will be set to 1 when the counter starts or stops, but it must be cleared by software.

				0: No trigger event occurred. 1: Trigger interrupt pending.
5	COMIF	RC_W0	0	COM interrupt flag This bit is set to 1 by hardware once a COM event occurs (when CCxE, CCxNE, OCxM have been updated). It is cleared by software. 0: No update occurred. 1: COM interrupt pending.
4:2	Reserved	-	-	Reserved
2	CC2IF	RC_W0	0	Capture/Compare 2 interrupt flag. Refer to CC1IF description
1	CC1IF	RC_W0	0	Capture/Compare 1 interrupt flag. If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value. It is cleared by software. 0: No match; 1: The content of the counter TIM15_CNT matches the content of the TIM15_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIM15_CCR1 register. 0: No input capture occurred 1: An input capture has occurred and the counter value has been loaded into TIM15_CCR1 (a edge with the selected polarity has been detected on IC1).
0	UIF	RC_W0	0	Update interrupt flag. This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: – If UDIS=0 in the TIM15_CR1 register, an update event is generated (counter overflow); – If UDIS=0 and URS=0 in the TIM15_CR1 register, an update event is generated when UG=1 in the TIM15_EGR register (software reinitializes CNT);

20.5.6. TIM15 event generation register (TIM15_EGR)

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	TG	COMG	Res.	Res.	CC2G	CC1G	UG
								W	W	W			W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	BG	W	0	Generate a braking event. This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt or DMA transfer can occur if enabled.
6	TG	W	0	Trigger event generated. This bit is set '1' by the software to generate a trigger event, and '0' is automatically cleared by the hardware. 0: No action 1: TIF=1 in the TIM15_SR register. If the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA are generated.
5	COMG	W	0	Capture/Compare event, generates control update. This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit acts only on channels that have a complementary output.

4:3	Reserved	-	-	Reserved
2	CC2G	W	0	Capture/Compare 2 event generated. Refer to CC1G description
1	CC1G	W	0	Capture/Compare 1 event generated. This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as input: CC1IF is set. If the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA are generated. If channel CC1 is configured as input: The current counter value is captured into the TIM15_CCR1 register, CC1IF is set. If the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA are generated. The CC1OF flag is set if the CC1IF flag was already set.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected).

20.5.7. TIM15 Capture/Compare mode register (TIM15_CCMR1).

Offset address: 0x18

Reset value: 0x0000 0000

The same register can be used for input capture mode or output compare mode. The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function for input capture and for output compare modes. For any given bit, OCxx indicates the function of the bit when the channel is configured as an output, while ICxx indicates the function when the channel is configured as an input.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OC2M [2:0]			OC2PE	OC2FE	CC2S [1:0]		Res.	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
IC2F [3:0]				IC2PSC [1:0]				IC1F [3:0]			IC1PSC [1:0]				
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

20.5.7.1. Input capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	IC2F [3:0]	RW	4'b0	Input capture 2 filter Refer to the description of IC1F[3:0].
11:10	IC2PSC [1:0]	RW	2'b0	Input/capture 2 prescaler Refer to the description of IC1PSC[1:0].
9:8	CC2S [1:0]	RW	2'b0	Capture/Compare 2 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM15_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIM15_CCER).
7:4	IC1F [3:0]	RW	4'b0	Input capture 1 filter This bit field defines the sampling frequency of the TI1 input and the sampling length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f _{DTS} 0001: f _{SAMPLING} = f _{CK_INT} , N = 2 0010: f _{SAMPLING} = f _{CK_INT} , N = 4 0011: f _{SAMPLING} = f _{CK_INT} , N = 8 0100: f _{SAMPLING} = f _{DTS} /2, N = 6 0101: f _{SAMPLING} = f _{DTS} /2, N = 8

				0110: $f_{\text{SAMPLING}} = f_{\text{DTS}}/4$, $N = 6$ 0111: $f_{\text{SAMPLING}} = f_{\text{DTS}}/4$, $N = 8$ 1000: $f_{\text{SAMPLING}} = f_{\text{DTS}}/8$, $N = 6$ 1001: $f_{\text{SAMPLING}} = f_{\text{DTS}}/8$, $N = 8$ 1010: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 5$ 1011: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 6$ 1100: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 8$ 1101: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 5$ 1110: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 6$ 1111: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 8$
3:2	IC1PSC [1:0]	RW	2'b0	Input/capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0' (TIM15_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S [1:0]	RW	2'b0	CC1S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM15_SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM15_CCER).

20.5.7.2. Output compare mode

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	Reserved
14:12	OC2M [2:0]	RW	3'b0	Output Compare 2 mode Refer to the description of OC1M.
11	OC2PE	RW	0	Output Compare 2 preload enable. Refer to the description of OC1PE.
10	OC2FE	RW	0	Output Compare 2 fast enable. Refer to the description of OC1FE.
9:8	CC2S	RW	0	Capture/Compare 2 selection This bit defines the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is only valid when the internal trigger input is selected via the TS bit (TIM15_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIM15_CCER).
7	Reserved	-	-	Reserved
6:4	OC1M [2:0]	RW	00	Output compare 1 mode This bit defines the behavior of the output reference signal OC1REF for OC1 and OC1N. OC1REF is active high, while the active level of OC1 and OC1N depends on the CC1P and CC1NP bits. 000: Frozen. The comparison between the output comparison register TIM1_CCR1 and the counter TIM1_CNT does not work for OC1REF; 001: Set channel 1 to output the active level upon match. When the value of counter TIM15_CNT matches the capture/compare register 1 (TIM15_CCR1), OC1REF is forced high. 010: Set channel 1 to output inactive level upon match. When the value of counter TIM15_CNT matches the capture/compare register 1 (TIM15_CCR1), OC1REF is forced low.

				011: Toggle When TIM15_CCR1 = TIM15_CNT, the level of OC1REF is toggled. 100: Force inactive level OC1REF is forced low. 101: Force active level OC1REF is forced high. 110: PWM mode 1—During up-counting, channel 1 is at the active level when TIM15_CNT < TIM15_CCR1, otherwise it is at the inactive level. 111: PWM mode 2—During up-counting, channel 1 is at the inactive level when TIM15_CNT < TIM15_CCR1, otherwise it is at the active level. Note 1: Once the LOCK level is set to 3 (LOCK bit in the TIM15_BDTR register) and CC1S = 00 (this channel is configured as output), this bit cannot be modified. Note: In PWM mode 1 or 2, the OCREF level changes when the result of the comparison changes or when the output compare mode switches from frozen to PWM mode.
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on TIM15_CCR1 disabled. TIM15_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM15_CCR1 enabled. Read/Write operations access the preload register. TIM15_CCR1 preload value is loaded in the active register at each update event. Note 1: Once the LOCK level is set to 3 (LOCK bit in the TIM15_BDTR register) and CC1S = 00 (this channel is configured as output), this bit cannot be modified. Note: The PWM mode can be used without validating the preload register only in one pulse mode. Else the behavior is not guaranteed.
2	OC1FE	RW	0	Output Compare 1 fast enable. This bit is used to accelerate the effect of trigger input events on CC outputs. 0: Even if the trigger is active, CC1 will operate normally based on the counter and CCR1 values. When a trigger input edge occurs, the minimum delay to activate CC1 output is 5 clock cycles. 1: A valid edge on the trigger input is equivalent to a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. The delay between sampling the trigger input and CC1 output is reduced to 3 clock cycles. OCxFE is effective only when the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIM15_SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM15_CCER).

20.5.8. TIM15 capture/compare enable register (TIM15_CCER)

Offset address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2NP	Res.	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	CC2NP	RW	0	[CC2NP,CC2P] bit selects the polarity of TI1FP2 and TI2FP2 as trigger or capture signals. Refer to CC1P description.

6	Reserved	-	-	Reserved
5	CC2P	RW	0	Input/Capture 2 output polarity. Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 output enable Refer to CC1E description.
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity. 0: OC1N active high. 1: OC1N active low. Note: Once the LOCK level (LOCK bit in TIM15_BDTR register) is set to 3 or 2 and CC1S=00 (channel configured as output), this bit cannot be modified.
2	CC1NE	RW	0	Input/Capture 1 complementary output enable 0: Off – OC1N output is disabled, so the output level of OC1N depends on the values of MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits. 1: On - OC1N signal output to the corresponding output pin, its output level depends on the values of MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits. On channels having a complementary output, this bit is pre-loaded. If the CCPC bit in the TIM1_CR2 register is set to 1, the CC1NE valid bit will only take the new value from the preload bit when a commutation event (COM) is generated.
1	CC1P	RW	0	Input/Capture 1 output polarity. CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: The [CC1NP, CC1P] bits select the polarity of TI1FP1 and TI2FP1 as trigger or capture signals: 00: Non-inverted/rising edge. TIxFP1 rising edge active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 non-inverted (gated mode, encoder mode). 01: Inverted/falling edge. TIxFP1 falling edge active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 inverted (gated mode, encoder mode). 10: reserved, do not use this configuration. 11: Non-inverted/both edges. TIxFP1 rising and falling edges both active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 non-inverted (gated mode). This configuration must not be used in encoder mode. Note 1: For complementary output channels, this bit is pre-loaded. If the CCPC bit in the TIM1_CR2 register is set to 1, the actual valid bit of CC1P will only load the preload value when a COM event occurs. Note 2: Once the LOCK level (LOCK bit in TIM15_BDTR register) is set to 3 or 2 and CC1S=00 (channel configured as output), this bit cannot be modified.
0	CC1E	RW	0	Input/Capture 1 output enable CC1 channel configured as output: 0: Off - OC1 output is disabled, so the output level of OC1 depends on the values of MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits. 1: On - OC1 signal is output to the corresponding output pin, and its output level depends on the values of MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits. CC1 channel configured as input: This bit determines whether the counter value can be captured by the TIM15_CCR1 register. 0: Capture disabled. 1: Capture enabled. Note: For complementary output channels, this bit is pre-loaded. If the CC1PC bit in the TIM1_CR2 register is set, the actual valid bit of CC1E will only load the preload value when a COM event occurs.

Table 20-2 Output control bits for complementary channels OCx and OCxN with brake function

Control bits					Output states	
MOE	OSSI	OSSR	CCxE	CCxNE	OCx output status	OCxN output state
1	x	x	0	0	Output disabled (not timer-driven), OCx=0, OCx_EN=0	Output disabled (not timer-driven), OCxN=0, OCxN_EN=0
		0	0	1	Output disabled (not timer-driven), OCx=0, OCx_EN=0	OCxREF + polarity OCxN=OCxREF XOR CCxNP, OCxN_EN=1
		0	1	0	OCxREF + polarity OCx=OCREF xor CCxP, OCx_EN=1	Output disabled (not timer-driven), OCxN=0, OCxN_EN=0
		x	1	1	OCREF + polarity + dead time, OCx_EN=1	Complement of OCREF (not OCREF) + polarity + dead time, OCxN_EN=1
		1	0	1	Off state (output enabled and at inactive level), OCx=CCxP, OCx_EN=1	OCxREF + polarity OCxN=OCxREF XOR CCxNP, OCxN_EN=1
		1	1	0	OCxREF + polarity OCx=OCxREF XOR CCxP, OCx_EN=1	Off-State (output enabled with inactive state), OCxN=CCxNP, OCxN_EN=1
0	0	x	x	x	Output disabled (not timer-driven)	
	1		0	0		
	1		0	1	Off-State (output enabled with inactive state) Asynchronous: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1 (if BRK is triggered). Subsequently (only valid when BRK is triggered), if the clock is present: after a dead time, assuming OISx and OISxN do not both correspond to the active levels of OCx and OCxN, then OCx=OISx and OCxN=OISxN.	
	1		1	0		
	1		1	1		

20.5.9. TIM15 counter register (TIM15_CNT)

Offset address: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	0	Counter value

20.5.10. TIM15 prescaler register (TIM15_PSC)

Offset address: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	0	Prescaler value. The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). The PSC contains the value to be loaded into the active prescaler register each time an update event occurs, including when the counter is cleared via the UG bit in the TIM15_EGR register, or when cleared via the trigger controller when configured in "reset mode".

20.5.11. TIM15 auto-reload register (TIM15_ARR)

Offset address: 0x2c

Reset value: 0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	ARR[15:0]	RW	0xFFFF	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

20.5.12. TIM15 repetition counter register (TIM15_RCR)

Offset address: 0x30

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	REP[7:0]	RW	8'b0	Value of the repetition counter When the preload function is enabled, these bits allow the user to set the update rate of the compare register (i.e., the periodic transfer from the preload register to the active register); when the update interrupt is enabled, it also affects the rate at which update interrupts are generated. Each time the down-counter REP_CNT reaches 0, an update event is generated, and the counter REP_CNT restarts counting from the REP value. Since REP_CNT only reloads the REP value when a repetitive update event (UIF) occurs, a new value written to the TIM1_RCR register will only take effect at the next repetitive update event. This means that in PWM mode, (REP+1) corresponds to: - the number of PWM periods in edge-aligned mode

20.5.13. TIM15 capture/compare register 1 (TIM15_CCR1)

Offset Address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1 [15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR1 [15:0]	RW	0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 contains the value (preload value) of the current capture/compare register 1 to be loaded. If the preload feature is not selected in the TIM15_CCMR1 register (OC1PE bit), this value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the counter TIM15_CNT, and outputs the signal on the OC1 port. If channel CC1 is configured as input: CCR1 is the counter value at the time of the last input capture 1 event (IC1). At this time, the TIM15_CCR1 register can only be read and cannot be programmed.

20.5.14. TIM15 capture/compare register 2 (TIM15_CCR2)

Offset Address: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2 [15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR2 [15:0]	RW	0	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 contains the value (preload value) of the current capture/compare register 2 to be loaded. If the preload feature is not selected in the TIM15_CCMR1 register (OC2PE bit), this value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual capture/compare register contains the value to be compared with the counter TIM15_CNT and outputs the signal on the OC2 port. If channel CC2 is configured as input: CCR2 is the counter value at the time of the last input capture 2 event (IC2). At this time, only the TIM15_CCR2 register can be read, and it cannot be programmed.

20.5.15. TIM15 brake and dead-time register (TIM15_BDTR)

Offset Address: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as one of the break inputs is active. It is set by software or automatically depending on the AOE bit. It is acting only on the channels which are configured in output. 0: Disable OC and OCN outputs or force them to idle state, depending on the OSSI bit; 1: If the corresponding enable bits (CCxE, CCxNE bits of the TIM15_CCER register) are set, the OC and OCN outputs are enabled.
14	AOE	RW	0	Automatic output enable 0: MOE can be set only by software; 1: MOE can be set by software or automatically at the next update event (if none of the break inputs is active). Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM15_BDTR register).
13	BKP	RW	0	Break polarity 0: Break input BRK is active low; 1: Break input BRK is active high Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM15_BDTR register).
12	BKE	RW	0	Break enable 0: Break function disabled 1: Break function enabled Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM15_BDTR register).
11	OSSR	RW	0	Off-state selection for Run mode This bit is used when MOE=1 on channels having a complementary output which are configured as outputs. OSSR bit is

				not implemented if no complementary output is implemented in the timer. Refer to the detailed description of OC/OCN enable. 0: In inactive state, disable OC/OCN outputs (OC/OCN enable output signal = 0); 1: When in the inactive state, once CCxE=1 or CCxNE=1, the OC/OCN output is enabled and outputs the inactive level. Note: This bit can not be modified as long as LOCK level 2 has been set (LOCK bits in TIM15_BDTR register).
10	OSSI	RW	0	Off-state selection for Idle mode This bit is used when MOE=0 and on channels configured as outputs. Refer to the detailed description of OC/OCN enable. 0: In inactive state, disable OC/OCN outputs (OC/OCN enable output signal = 0); 1: In inactive state, OC/OCN first outputs its inactive level once CCxE=1 or CCxNE=1. Note: This bit can not be modified as long as LOCK level 2 has been set (LOCK bits in TIM15_BDTR register).
9:8	LOCK[1:0]	RW	2'b0	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: Lock level 1, cannot write to DTG/BKE/BKP/AOE bits of TIM15_BDTR register, or OISx/OISxN bits of TIM15_CR2 register; 10: Lock level 2, cannot write to bits in lock level 1, nor to CC polarity bits (CCxP/CCxNP bits in TIM15_CCER register once the corresponding channel is configured as output via CCxS bits) or OSSR/OSSI bits; 11: Lock level 3, cannot write to bits in lock level 2, nor write to CC control bits (once the relevant channel is configured as output via CCxS bits, the OCxM/OCxPE bits in TIM15_CCMRx register); Note: The LOCK bits can be written only once after the reset. Once the TIM15_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG[7:0]	RW	8'b0	Dead-time generator setup. This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. $DTG[7:5]=0xx \Rightarrow DT=DTG[7:0] \times T_{dtg}, \quad T_{dtg} = T_{DTS};$ $DTG[7:5]=10x \Rightarrow DT=(64+DTG[5:0]) \times T_{dtg}, \text{ where } T_{dtg} = 2 \times T_{DTS};$ $DTG[7:5]=110 \Rightarrow DT=(32+DTG[4:0]) \times T_{dtg}, \text{ where } T_{dtg} = 8 \times T_{DTS};$ $DTG[7:5]=111 \Rightarrow DT=(32+DTG[4:0]) \times T_{dtg}, \text{ where } T_{dtg} = 16 \times T_{DTS};$ Example if $T_{DTS}=125\text{ns}$ (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps Note: Once LOCK level (LOCK bits in TIM15_BDTR register) is set to 1, 2, or 3, these bits cannot be modified.

20.5.16. TIM15 input selection register (TIM15_TISEL)

Offset address: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	TI2SEL [3:0]				Res.	Res.	Res.	Res.	TI1SEL [3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:8	TI2SEL [3:0]	RW	4'b0	TI2 input selection. 0000: TIM15_CH2 0001: COMP2_OUT

				0010: COMP1_OUT Other: Reserved
7:4	Reserved	-	-	Reserved
3:0	TI1SEL [3:0]	RW	4'b0	TI1 input selection. 0000: TIM15_CH1 0001: LSE_CSS 0010: COMP1_OUT 0011: COMP2_OUT Other: Reserved

20.5.17. TIM15 alternate function option register 1 (TIM15_AF1)

Offset address: 0x60

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETRSEL[3:2]	
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL[1:0]	Res.	Res.	BKCOMP2_P	BKCOMP1_P	BKIN_P	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BKCOMP2_E	BKCOMP1_E	BKIN_E
RW	RW			RW	RW	RW							RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	Reserved
17:14	ETRSEL[3:0]	RW	4'b0	External trigger source selection This bit segment is used to select an ETR input source 0000: TIM15_ETR 0001: COMP1_OUT 0010: COMP2_OUT 0011: Reserved 0100: Reserved 0101: ADC_AWD Others: Reserved Note: Once the LOCK level (LOCK bit in TIM15_BDTR register) is set to 1, this bit cannot be modified.
13:12	Reserved	-	-	Reserved
11	BKCOMP2P	RW	0	Brake input polarity control from Comparator 2. This bit selects the brake input polarity from comparator 2 and must be configured together with the BKP polarity bit. 0: Comparator 2 brake input polarity not inverted (active low when BKP=0, active high when BKP=1) 1: Comparator 2 brake input polarity inverted (active high when BKP=0, active low when BKP=1) Note: Once the LOCK level (LOCK bit in TIM15_BDTR register) is set to 1, this bit cannot be modified.
10	BKCOMP1P	RW	0	Brake input polarity control from Comparator 1. This bit selects the brake input polarity from comparator 1 and must be configured together with the BKP polarity bit. 0: Comparator 1 brake input polarity not inverted (active low when BKP=0, active high when BKP=1) 1: Comparator 1 brake input polarity inverted (active high when BKP=0, active low when BKP=1) Note: Once the LOCK level (LOCK bit in TIM15_BDTR register) is set to 1, this bit cannot be modified.
9	BKINP	RW	0	BKIN input polarity. This bit selects the alternate function of the BKIN input polarity and must be configured at the same time as the BKP polarity bit 0: BKIN input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: BKIN input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in TIM15_BDTR register) is set to 1, this bit cannot be modified.
8:3	Reserved	-	-	Reserved
2	BKCOMP2E	RW	0	Comparator 2 brake input enable. Comparator 2 brake input is ORed with other brake sources as the brake input. 0: Comparator 2 brake input disabled 1: Comparator 2 brake input enabled

				Note: Once the LOCK level (LOCK bit in TIM15_BDTR register) is set to 1, this bit cannot be modified.
1	BKCOMP1E	RW	0	Comparator 1 brake input enable. Comparator 1 brake input is ORed with other brake sources as the brake input. 0: Comparator 1 brake input off 1: Comparator 1 brake input enabled Note: Once the LOCK level (LOCK bit in TIM15_BDTR register) is set to 1, this bit cannot be modified.
0	BKINE	RW	1	BKIN input enabled. Comparator 1 brake BKIN input is ORed with other brake sources as the brake input. 0: BKIN input off 1: BKIN input on Note: Once the LOCK level (LOCK bit in the TIM15_BDTR register) is set to 1, this bit cannot be modified.

20.5.18. TIM15 DMA control register (TIM15_DCR)

Offset address: 0x3DC

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset value	Function
31:13	Reserved	-	-	Reserved
12:8	DBL[4:0]	RW	5'b0	DMA burst length These bits define the DMA transfer length (when reading or writing to the address of the TIM15_DMAR register, the timer performs a continuous transfer), i.e., define the number of bytes to be transferred: 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers 10001: 18 transfers
7:5	Reserved	-	-	Reserved
4:0	DBA[4:0]	RW	5'b0	DBA[4:0]: DMA base address These bits define the base address for DMA in continuous mode (when reading or writing to the address of the TIM15_DMAR register). DBA is defined as an offset from the address of the TIM15_CR1 register: 00000: TIM15_CR1 00001: TIM15_CR2 00010: TIM15_SMCR

20.5.19. TIM15 DMA continuous transfer address register (TIM15_DMAR)

Offset address: 0x3E0

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[31:0]	RW	0	DMA register for burst accesses Reading or writing to the TIM15_DMAR register results in access operations to the following register address: TIM15_CR1 address + (DBA + DMA pointer) x4, where: TIM15_CR1 address is the address of the control register 1; "DBA" is the base address defined in the TIM15_DCR register; "DMA pointer" is an offset automatically controlled by DMA, which depends on the DBL defined in the TIM15_DCR register.

underflow value in downcounting) and the UDIS bit in the TIMx_CR1 register is 0. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in the TIMx_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR1 register.

21.2.2.1. Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly:

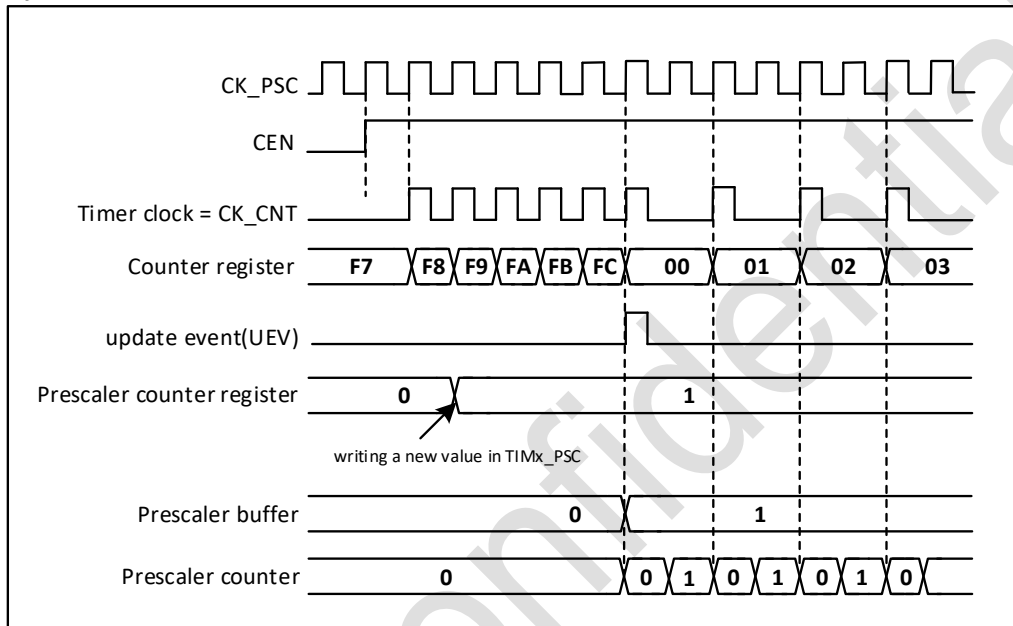


Figure 21-2 Counter timing diagram when prescaler division changes from 1 to 2

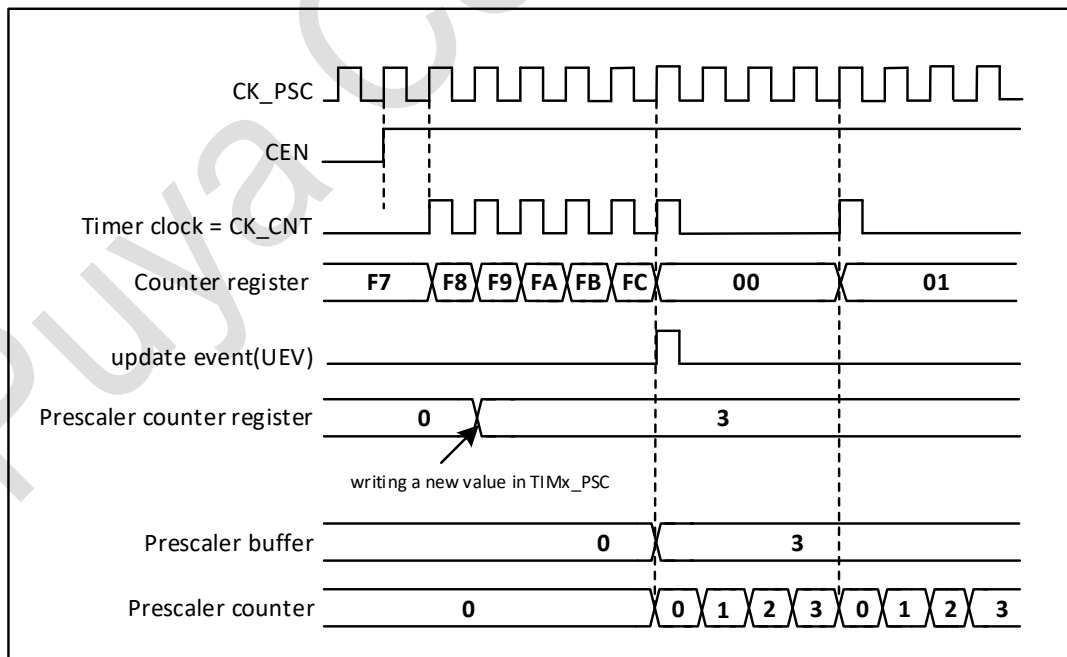


Figure 21-3 Counter timing diagram when prescaler division changes from 1 to 4

21.2.3. Counting mode (upcounting)

In upcounting mode, the counter counts from 0 to the auto-reload value (TIMx_ARR register), then restarts from 0 and generates a counter overflow event.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

Setting the UDIS bit in the TIMx_CR1 register disables the update event (UEV); this also prevents updating the shadow registers when writing new values to the preload registers. Then no update event occurs until the UDIS bit has been written to 0. Even so, the counter is still cleared to '0' when an update event should occur, and the prescaler count is also cleared (though the prescaler value remains unchanged). Additionally, if the URS bit (update request selection) in the TIMx_CR1 register is set, setting the UG bit generates an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all registers are updated, and the update flag (UIF bit in the TIMx_SR register) is set to 1 (depending on the URS bit).

- The repetition counter is reloaded with the value in the TIMx_RCR register.
- The auto-load shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The figure below shows several counter behaviors at different frequencies when TIMx_ARR=0x36.

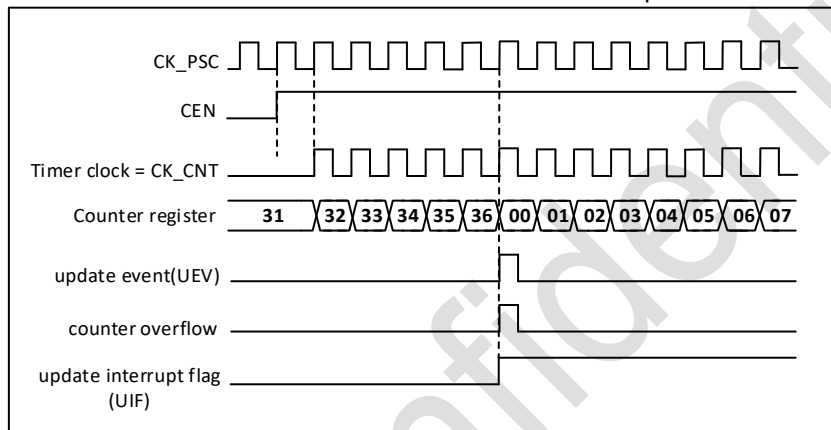


Figure 21-4 Counter timing diagram, internal clock divided by 1

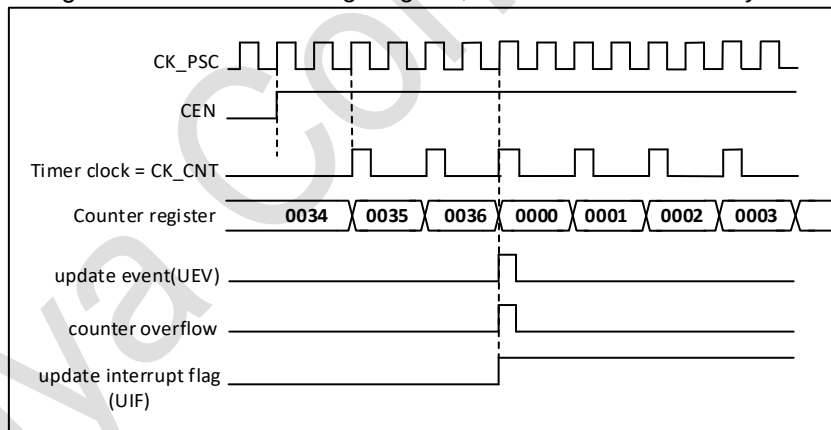


Figure 21-5 Counter timing diagram, internal clock divided by 2

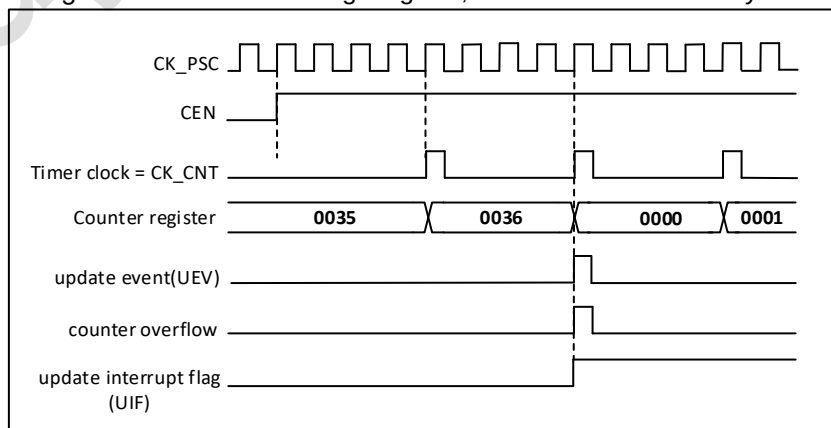


Figure 21-6 Counter timing diagram, internal clock divided by 4

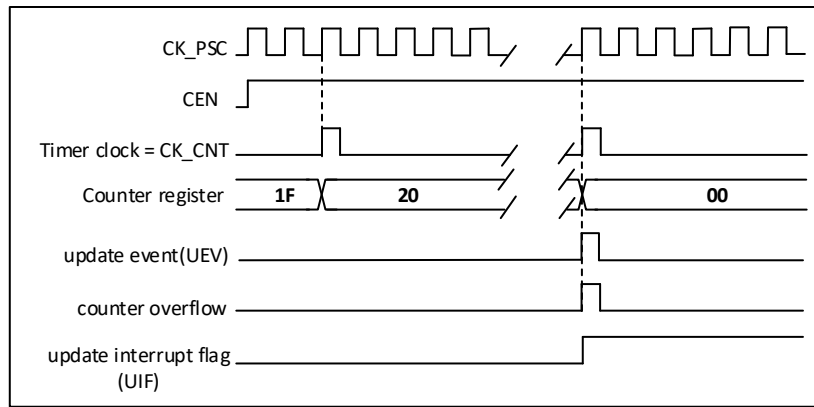


Figure 21-7 Counter timing diagram, internal clock divided by N

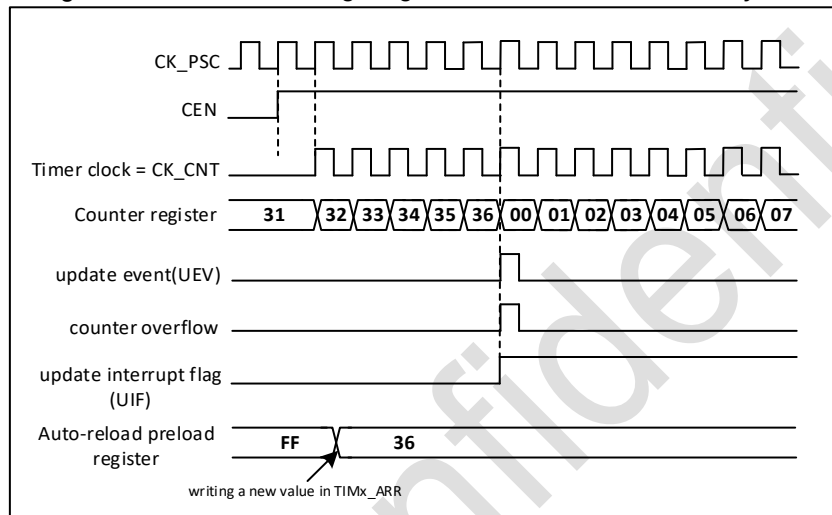


Figure 21-8 Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

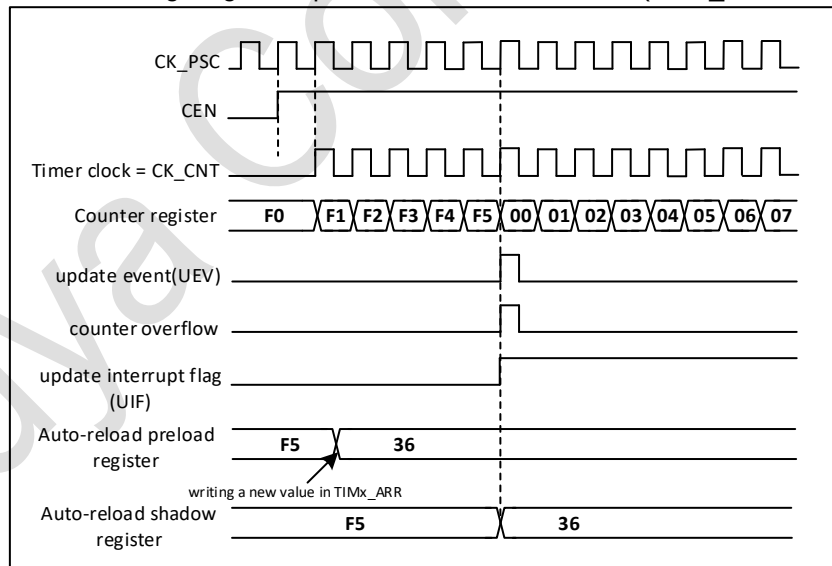


Figure 21-9 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preload)

21.2.4. Repetition counter

The time base unit describes how the update event (UEV) is generated when the counter overflows, which actually occurs only when the repetition counter counts down to zero. This is useful for generating PWM signals.

This means that on every Nth counter overflow, data is transferred from the preload register to the shadow register (TIMx_ARR auto-reload register, TIMxPSC prescaler register, and capture/compare register TIMx_CCRx in compare mode), where N is the value in the TIMx_RCR repetition counter register.

The repetition counter decrements on each counter overflow in increment counter mode.

The repetition counter is auto-reloaded, and the repetition rate is defined by the value of the TIMx_RCR register. When an update event is generated by software (by setting the UG bit in TIMx_EGR) or by the hardware slave mode controller, an update event occurs immediately regardless of the repetition counter value, and the content of the TIMx_RCR register is reloaded into the repetition counter.

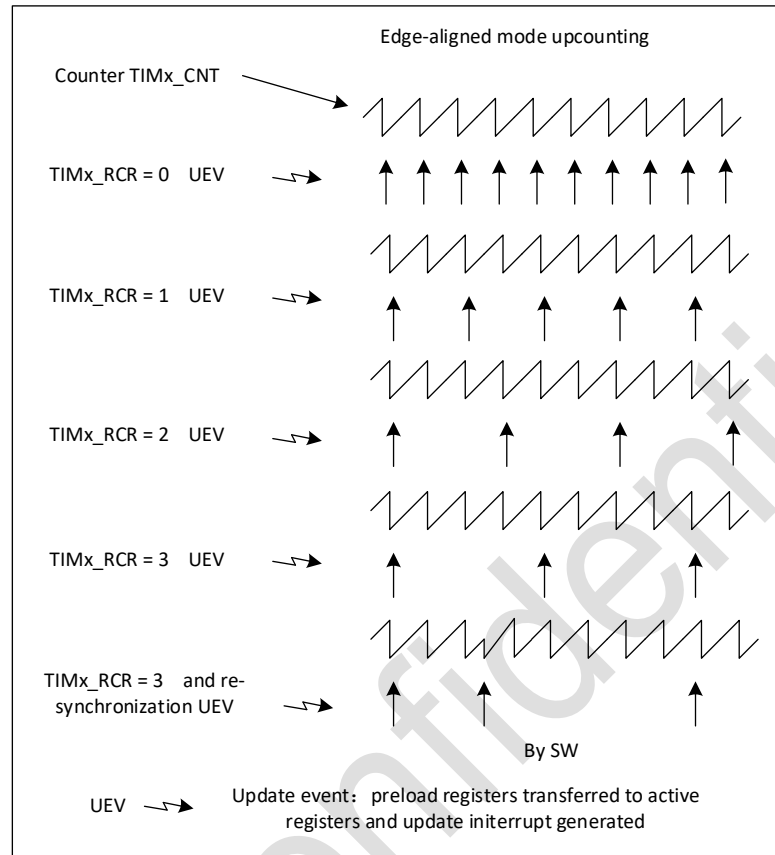


Figure 21-10 Example of update frequency under different modes and TIMx_RCR register settings

21.2.5. Clock sources

The counter clock is provided by the Internal clock (CK_INT) source. The CEN (in the TIMx_CR1 register) and UG bits (in the TIMx_EGR register) are actual control bits and can be changed only by software (except for UG that remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

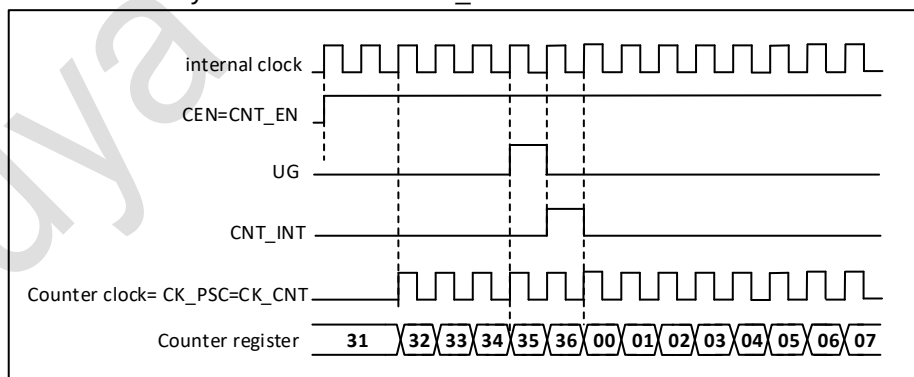
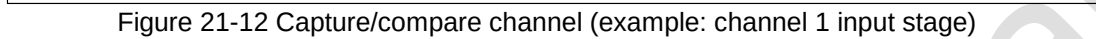


Figure 21-11 Control circuit in normal mode, internal clock divided by 1

21.2.6. Capture/Compare channels

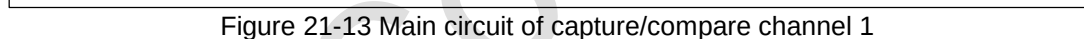
Each capture/compare channel is built around a capture/compare register (including a shadow register), a capture input section (digital filter, multiplexer, and prescaler), and an output section (comparator and output control).

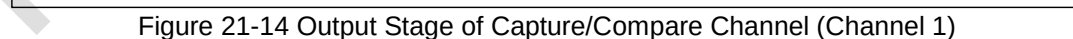
The input stage samples the corresponding Tlx input to generate a filtered signal TlxF. Then, an edge detector with polarity selection generates a signal (TlxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).



The output stage generates an intermediate waveform (active high) which is then used for re

ADD Bus





The capture/compare block is made of one preload register and one shadow register. Write

In capture mode, captures are actually done in the shadow register, which is copied into the

In compare mode, the content of the preload register is copied into the shadow register when

Input capture mode:

In Input capture mode, the

CCXIF flag (TIMx_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when you write it to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input rises. To do this, use the following procedure:

- Use the TI1SEL[3:0] bits in the TIMx_TISEL register to select the correct TI1x source (internal or external)
- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMx_CCR1 register becomes read-only.
- Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register)). Let's imagine that, when toggling, the input signal is not stable during at most 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at f_{DTS} frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the active transition edge for the TI1 channel by writing CC1P=0 (rising edge) (and CC1NP=0) in the TIMx_CCER register.
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PSC bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- If the CC1DE bit is set, a DMA request will be generated.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

21.2.8. Forced Output Mode

In output mode (CCxS=00 in the TIMx_CCMRx register), the output compare signals (OCxREF and the corresponding OCx) can be directly forced to active or inactive by software, independent of the comparison result between the output compare register and the counter. To force an output compare signal (OCxREF/OCx) to its active level, you just need to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCxREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level. The OCxREF signal can be forced low by writing the OCxM bits to 0100 in the TIMx_CCMRx register.

Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

21.2.9. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed. When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCxM=0000), be set active (OCxM=0001), be set inactive (OCxM=0010) or can toggle (OCxM=0011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).

- If the corresponding DMA enable bit is set (CCxDE bit in the TIMx_DIER register, with the CCDS bit in the TIMx_CR2 register selecting the DMA request function), a DMA request is generated.

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register. In output compare mode, the update event UEV has no effect on OCxREF and OCx output.

The timing accuracy can reach one counting cycle of the counter. The output compare mode can also be used to output a single pulse (in one-pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:
 - Set OCxM=0011 to toggle the OCx output pin when the counter matches CCRx,
 - Write OCxPE = 0 to disable preload register
 - Write CCxP = 0 to select active high polarity
 - Write CCxE = 1 to enable the output
5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated at any time via software to control the output waveform, provided the preload register is not used (OCxPE='0'; otherwise, the shadow register of TIMx_CCRx can only be updated upon the next update event). An example is given in the figure below.

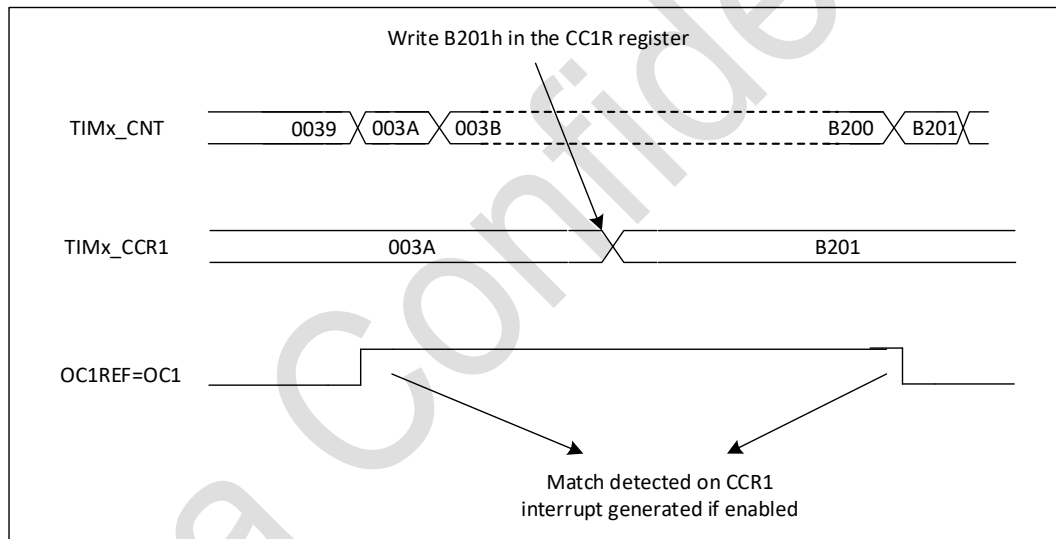


Figure 21-15 Output Compare Mode, Toggling OC1

21.2.10. PWM mode

PWM mode allows generation of a signal whose frequency is determined by the TIMx_ARR register and duty cycle by the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bits in the TIMx_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and finally, the ARPE bit in the TIMx_CR1 register must be set to enable the auto-reload preload register (in upcounting or center-aligned mode).

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIMx_EGR register.

The polarity of OCx can be set via software using the CCxP bit in the TIMx_CCER register, which can be configured as active high or active low. The output enable of OCx is controlled by a combination of CCxE, CCxNE, MOE, OSSI, and OSSR bits (TIMx_CCER and TIMx_BDTR registers). Refer to the TIMx_CCER register description for more details.

In PWM modes (Mode 1 or Mode 2), TIMx_CNT and TIMx_CCRx are continuously compared to determine whether $TIMx\_CNT \leq TIMx\_CCRx$ is satisfied.

The timer can generate edge-aligned PWM only when the counter is counting up.

21.2.10.1. PWM edge-aligned mode

The reference PWM signal OCxREF is high as long as $TIMx\_CNT < TIMx\_CCRx$ else it becomes low. If the compare value in $TIMx\_CCRx$ is greater than the auto-reload value (in $TIMx\_ARR$) then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'.

Figure below shows some edge-aligned PWM waveforms in an example where $TIMx\_ARR=8$.

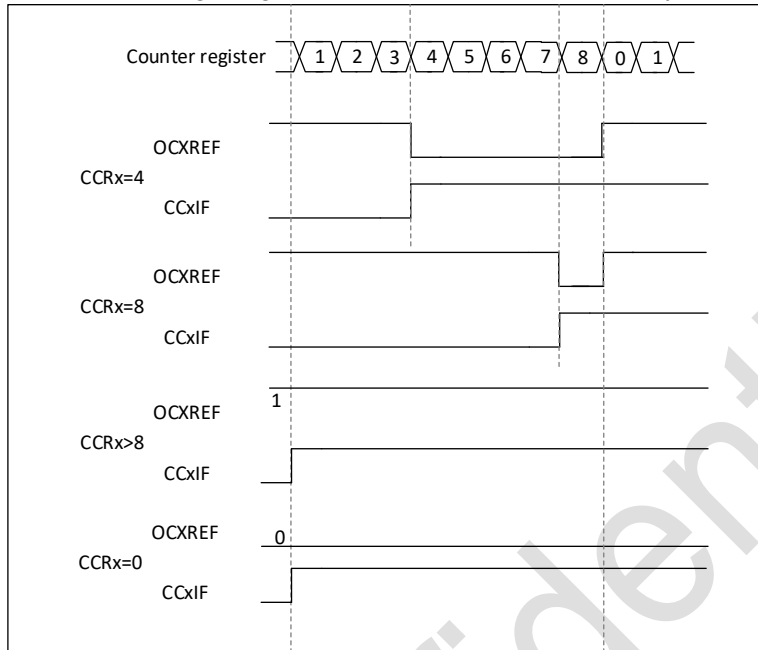


Figure 21-16 Edge-aligned PWM output waveform (ARR=8)

21.2.11. Complementary outputs and dead-time insertion

TIM16/17 can output a pair of complementary signals and manage the instantaneous turn-off and turn-on of the outputs. This time is generally known as dead-time, and you have to adjust it depending on the devices you have connected to the outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches...)

You can select the polarity of the outputs (main output OCx or complementary OCxN) independently for each output. This is done by writing to the CCxP and CCxNP bits in the $TIMx\_CCER$ register.

The complementary signals OCx and OCxN are controlled by a combination of the following control bits: CCxE and CCxNE bits in the $TIMx\_CCER$ register, and MOE, OISx, OISxN, OSS1, and OSSR bits in the $TIMx\_BDTR$ and $TIMx\_CR2$ registers (refer to the control bits for complementary output channels OCx and OCxN with brake function in the table following $TIMx\_CCER$). In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. There is one 8-bit dead-time generator for each channel. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal except for the rising edge, which is delayed relative to the reference rising edge.
- The OCxN output signal is the opposite of the reference signal except for the rising edge, which is delayed relative to the reference falling edge.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose $CCxP=0$, $CCxNP=0$, $MOE=1$, $CCxE=1$ and $CCxNE=1$ in these examples)

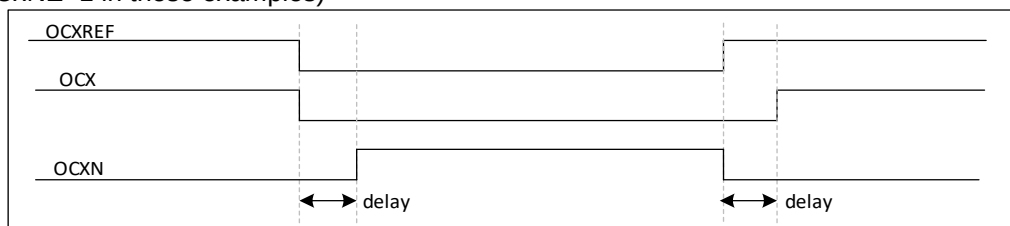


Figure 21-17 Complementary output with dead-time insertion

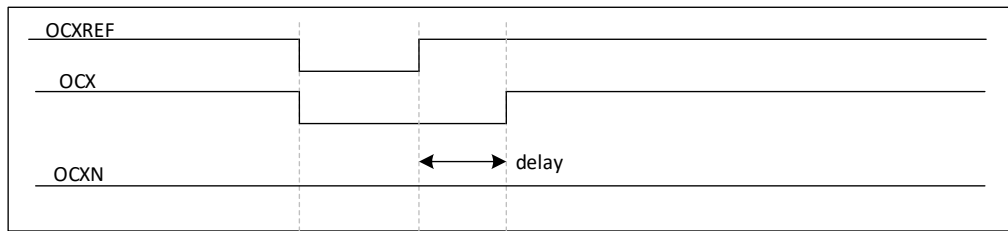


Figure 21-18 Dead-time waveform where the delay time is greater than the negative pulse width

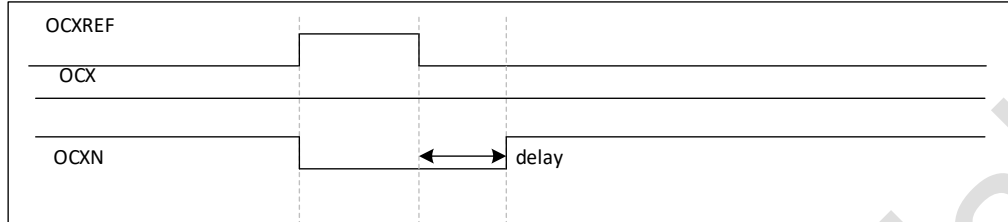


Figure 21-19 Dead-time waveform where the delay time exceeds the positive pulse width

The dead-time delay is the same for each of the channels and is programmable with the DTG bits in the TIMx_BDTR register.

21.2.11.1. Redirect OCxREF to OCx or OCxN

In output modes (forced mode, output compare mode, or PWM mode), OCxREF can be redirected to the output of OCx or OCxN by configuring the CCxE and CCxNE bits in the TIMx_CCER register.

This feature allows sending a special waveform (e.g., PWM or static active level) on one output while keeping the complementary output at an inactive level when the complementary output is inactive. Alternatively, both outputs can be kept at inactive levels simultaneously, or both can be kept at active levels with complementary outputs and dead-time.

Note: When only OCxN is enabled (CCxE = 0, CCxNE = 1), it does not invert and becomes high immediately when OCxREF is active. For example, if CCxNP=0 then OCxN=OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE=CCxNE=1) OCx becomes active when OCxREF is high whereas OCxN is complemented and becomes active when OCxREF is low.

21.2.12. Using the break function

When the brake function is used, both the output enable signal and the inactive level signal will be modified according to additional control bits (MOE, OSS1, and OSSR in the TIMx_BDTR register, OISx and OISxN in the TIMx_CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time.

The brake source (BRK) signal channel can be an external input signal (BKIN) or the following internal signals:

- Core LOCKUP output
- the PVD output
- Clock failure event signal generated by CSS detection
- the output from a comparator
- SRAM byte parity error
- FLASH ECC error

A break event can also be generated by software through the BG bit in the TIM15_EGR register. Regardless of the value of the BKE enable bit, a brake can be generated via software using BG.

When exiting from reset, the break circuit is disabled and the MOE bit is low. You can enable the break function by setting the BKE bit in the TIMx_BDTR register. The break input polarity can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time. When the BKE and BKP bits are written, a delay of 1 APB clock cycle is applied before the writing is effective. Consequently, it is necessary to wait 1APB clock period to correctly read back the bit after the write operation.

MOE falling edge can be asynchronous, thus a resynchronization circuit has been inserted between the actual signal (acting on the outputs) and the synchronous control bit (accessed in the TIMx_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if you write MOE to 1 whereas it was low, you must insert a delay (dummy instruction) before reading it correctly. This is because the write acts on the asynchronous signal whereas the read reflects the synchronous signal.

When a break occurs (selected level on the break input):

- The MOE bit is cleared asynchronously, putting the outputs in inactive state, idle state or in reset state (selected by the OSS1 bit). This feature functions even if the MCU oscillator is off.

- Each output channel is driven with the level programmed in the OISx bit in the TIMx_CR2 register as soon as MOE=0. If OSS1=0 then the timer releases the enable output else the enable output remains high.
- When complementary outputs are used:
 - The outputs are first put in reset state inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
 - If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note: Due to MOE resynchronization, the dead time is slightly longer than usual (approximately 2 timer clock cycles).
 - If OSS1=0 then the timer releases the enable outputs else the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.
- If the BIE bit in the TIMx_DIER register is set, an interrupt is generated when the brake status flags (SBIF and BIF bits in the TIMx_SR register) are '1'.
- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance. Else, MOE remains low until it is written to '1' again. In this case, it can be used for security and the break input can be connected to an alarm from power drivers, thermal sensors or any security components.

Note: The break inputs are active on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status flag BIF can not be cleared.

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the TIMx_BDTR register.

There are two ways to generate a brake:

- Through the BKR input with programmable polarity, while enabling BKE in the TIMx_BDTR register
- By software setting the BG bit in TIMx_EGR.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The application can choose from 3 levels of protection selected by the LOCK bits in the TIMx_BDTR register. The LOCK bits can be written only once after an MCU reset.

The figure below shows an example of behavior of the outputs in response to a break.

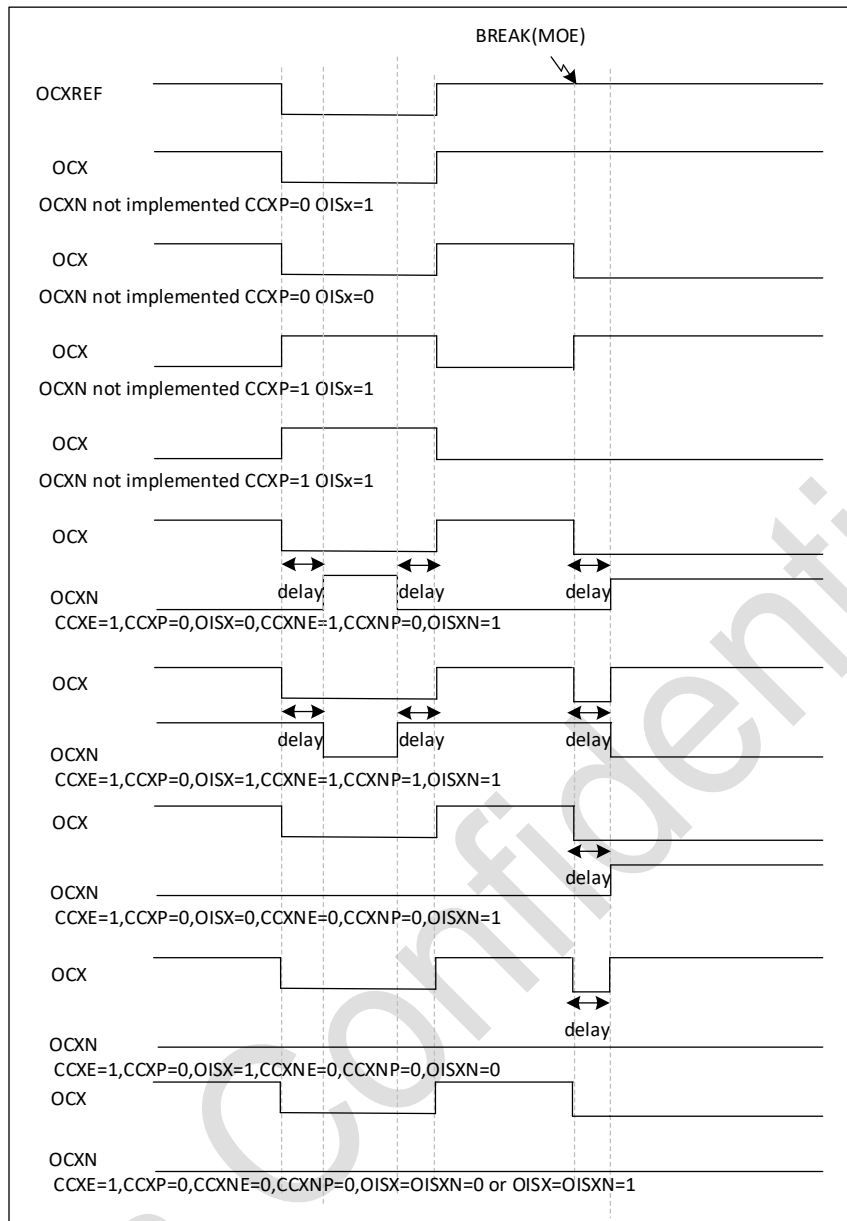


Figure 21-20 Different output behaviors in response to a brake event

21.3. TIM16/TIM17 interrupt

Table 21-1 TIM6/TIM7 interrupt

Interrupt event	Event flag	Interrupt enable control bit
update	UIF	UIE
Compare/Capture 1	CC1IF	CC1IE
Compare/Capture 2	CC2IF	CC2IE
COM	COMIF	COMIE
Trigger	TIF	TIE
Brake	BIF	BIE

21.4. TIM16/TIM17 debug mode

When the chip enters debug mode, depending on the setting of DBG_TIMx_STOP in the MCUDBG module, the TIM16/TIM17 counters can continue to operate normally or stop working.

21.5. TIM16/TIM17 registers

21.5.1. TIM16/17 Control Register 1 (TIMx_CR1) (x=16,17)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.	Res.	Res.	Res.	URS	UDIS	CEN
						RW	RW	RW					RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	2'b0	Clock division factor. These 2 bits define the division ratio between the timer clock (CK_INT) frequency and the dead-time generator and sampling clock (t _{DTS}) used by the digital filter (ETR, TIX). 00: t _{DTS} = t _{CK_INT} 01: t _{DTS} = 2 × t _{CK_INT} 10: t _{DTS} = 4 × t _{CK_INT} 11: reserved, do not use this configuration.
7	ARPE	RW	0	Auto-reload preload enable bit. 0: TIM1_ARR register is not buffered. 1: TIM1_ARR register is buffered.
6:3	Reserved	-	-	Reserved
2	URS	RW	0	Update request source. This bit is set and cleared by software to select the UEV event sources. 0: If update interrupt or DMA request is enabled, any of the following events generates an update interrupt or DMA request: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller 1: If update interrupt or DMA request is enabled, only counter overflow/underflow generates an update interrupt or DMA request
1	UDIS	RW	0	Update disable. Software enables/disables UEV event generation through this bit. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller Shadow registers are loaded with preload values. 1: UEV disabled. The update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). The counter and prescaler are reinitialized if the UG bit is set or a hardware reset is issued by the slave mode controller.
0	CEN	RW	0	Counter enable. 0: Counter disabled 1: Enable counter Note: External clock, gated mode, and encoder mode can only be used after software sets the CEN bit. Trigger mode can automatically set the CEN bit to 1 via hardware.

21.5.2. TIM16/17 Control Register 2 (TIMx_CR2) (x=16,17)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	OIS1N	OIS1	Res.	Res.	Res.	Res.	CCDS	Res.	Res.	CCPC
						RW	RW					RW			RW

Bit	Name	R/W	Reset Value	Function

31:10	Reserved	-	-	Reserved
9	OIS1N	RW	0	Output idle state 1 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=0 Note: This bit cannot be modified after LOCK levels 1, 2, or 3 have been set in the TIMx_BDTR register.
8	OIS1	RW	0	Output idle state 1 (OC1 output) 0: OC1=0 after a dead-time when MOE=0 1: OC1=0 after a dead-time when MOE=1 Note: This bit cannot be modified after LOCK (TIMx_BDTR register) levels 1, 2, or 3 have been set.
7:4	Reserved	-	-	Reserved
3	CCDS	RW	0	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
2:1	Reserved	-	-	Reserved
0	CCPC	RW	0	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded. 1: CCxE, CCxNE, and OCxM bits are preloaded; after setting this bit, these bits are updated only when a commutation event (COM) occurs (COMG bit set to 1 or a rising edge detected on TRGI, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output.

21.5.3. TIM16/17 DMA/Interrupt Enable Register (TIMx_DIER) (x=16,17)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CC1DE	UDE	BIE	Res.	COMIE	Res.	Res.	Res.	CC1IE	UIE
						RW	RW	RW		RW				RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	CC1DE	RW	0	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled 1: CC1 DMA request enabled
8	UDE	RW	0	Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled
7	BIE	RW	0	Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled
6	Reserved	-	-	Reserved
5	COMIE	RW	0	COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4:2	Reserved	-	-	Reserved
1	CC1IE	RW	0	Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

21.5.4. TIM16/17 Status Register (TIMx_SR) (x=16,17)

Offset address: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CC1OF	Res.	BIF	Res.	COMIF	Res.	Res.	Res.	CC1F	UIF
						RC_W0		RC_W0		RC_W0				RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	CC1OF	RC_W0	0	Capture/Compare 1 overcapture flag This flag can be set to 1 by hardware only when the corresponding channel is configured as input capture. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The TIMx_CCR1 register has captured the counter value when CC1IF is set to 1.
8	Reserved	-	-	Reserved
7	BIF	RC_W0	0	Brake interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input.
6	Reserved	-	-	Reserved
5	COMIF	RC_W0	0	COM interrupt flag This bit is set by hardware once a COM event occurs (when CCxE, CCxNE, OCxM have been updated). It is cleared by software. 0: No update occurred. 1: COM interrupt pending.
4:2	Reserved	-	-	Reserved
1	CC1IF	RC_W0	0	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value. It is cleared by software. 0: No match; 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. Or when When the value of TIMx_CCR1 is greater than TIMx_ARR, CC1IF will be set upon counter overflow during increment counting. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	RC_W0	0	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - If UDIS=0 in the TIMx_CR1 register, an update event (counter overflow) is generated when REP_CNT=0; If UDIS of the TIMx_CR1 register = 0 and URS = 0, an update occurs when UG of the TIMx_EGR register = 1

21.5.5. TIM16/17 Event Generation Register (TIMx_EGR) (x=16,17)

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	Res.	COMG	Res.	Res.	Res.	CC1G	UG
								W		W				W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	BG	W	0	Break generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action

				1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt or DMA transfer can occur if enabled.
6	Reserved	-	-	Reserved
5	COMG	W	0	Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit acts only on channels that have a complementary output.
4:2	Reserved	-	-	Reserved
1	CC1G	W	0	Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as input: CC1IF is set. If the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA are generated. If channel CC1 is configured as input: The current counter value is captured into the TIMx_CCR1 register, CC1IF is set. If the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. The CC1OF flag is set if the CC1IF flag was already set.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected).

21.5.6. TIM16/17 Capture/Compare Mode Register 1 (TIMx_CCMR1) (x=16,17)

Offset address: 0x18

Reset value: 0x0000 0000

The same register can be used for input capture mode or output compare mode. The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function for input capture and for output compare modes. For any given bit, OCxx indicates the function of the bit when the channel is configured as an output, while ICxx indicates the function when the channel is configured as an input.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [2:0]			OC1PE	Res.	CC1S [1:0]	
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1F [3:0]			IC1PSC [1:0]				
								RW	RW	RW	RW	RW	RW	RW	

21.5.6.1. Input capture mode:

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:4	IC1F [3:0]	RW	4'b0	Input capture 1 filter This bit field defines the sampling frequency of the TI1 input and the sampling length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f _{DTS} 0001: f _{SAMPLING} = f _{CK_INT} , N = 2 0010: f _{SAMPLING} = f _{CK_INT} , N = 4 0011: f _{SAMPLING} = f _{CK_INT} , N = 8 0100: f _{SAMPLING} = f _{DTS} /2, N = 6 0101: f _{SAMPLING} = f _{DTS} /2, N = 8 0110: f _{SAMPLING} = f _{DTS} /4, N = 6 0111: f _{SAMPLING} = f _{DTS} /4, N = 8 1000: f _{SAMPLING} = f _{DTS} /8, N = 6

				1001: $f_{\text{SAMPLING}} = f_{\text{DTS}}/8$, $N = 8$ 1010: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 5$ 1011: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 6$ 1100: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 8$ 1101: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 5$ 1110: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 6$ 1111: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 8$
3:2	IC1PSC [1:0]	RW	2'b0	Input/capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). Once CC1E=0 (in the TIMx_CCER register), the prescaler is reset. 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S [1:0]	RW	2'b0	CC1S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: Reserved 11: Reserved Note: CC1S is writable only when the channel is closed (CC1E=0 in the TIM16/17_CCER register).

21.5.6.2. Output compare mode

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6:4	OC1M [2:0]	RW	00	Output compare 1 mode This bit defines the behavior of the output reference signal OC1REF for OC1 and OC1N. OC1REF is active high, while the active level of OC1 and OC1N depends on the CC1P and CC1NP bits. 000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 001: Set channel 1 to output active level upon match. When the value of counter TIMx_CNT matches Capture/Compare Register 1 (TIMx_CCR1), force OC1REF to high level. 010: Set channel 1 to output inactive level upon match. When the value of counter TIMx_CNT matches Capture/Compare Register 1 (TIMx_CCR1), force OC1REF to low level. 011: Toggle When TIMx_CCR1=TIMx_CNT, OC1REF toggles. 100: Force inactive level Force OC1REF to low level. 101: Force active level Force OC1REF to high level. 110: PWM Mode 1---During up-counting, channel 1 is active when TIMx_CNT<TIMx_CCR1, otherwise it is inactive; 111: PWM Mode 2---During up-counting, channel 1 is inactive when TIMx_CNT<TIMx_CCR1, otherwise it is active; Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note: In PWM mode 1 or 2, the OCREF level changes when the result of the comparison changes or when the output compare mode switches from frozen to PWM mode.
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on TIMx_CCR1 disabled. TIM1_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIMx_CCR1 enabled. Read/Write operations access the preload register. TIM1_CCR1 preload value is loaded in the active register at each update event. Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output).
2	Reserved	-	-	Reserved
1:0	CC1S [1:0]	RW	00	Capture/Compare 1 selection

				This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: Reserved; 11: Reserved; Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

21.5.7. TIM16/17 Capture/Compare Enable Register (TIMx_CCER) (x=16,17)

Offset address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1NP	CC1NE	CC1P	CC1E
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity. 0: OC1N active high. 1: OC1N active low. This bit, together with CC1P, defines the polarity of TI1FP1/TI2FP1. Refer to the description of CC1P. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 3 or 2 and CC1S = 00 (channel configuration as output), this bit cannot be modified.
2	CC1NE	RW	0	Input/Capture 1 complementary output enable 0: Off – OC1N output is disabled, so the output level of OC1N depends on the values of MOE, OSS1, OSSR, OIS1, OIS1N, and CC1E bits. 1: On - OC1N signal output to the corresponding output pin, its output level depends on the values of MOE, OSS1, OSSR, OIS1, OIS1N, and CC1E bits.
1	CC1P	RW	0	Input/Capture 1 output polarity. CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: The [CC1NP, CC1P] bits select the polarity of TI1FP1 and TI2FP1 as trigger or capture signals: 00: Non-inverted/rising edge. TIxFP1 rising edge active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 non-inverted (gated mode, encoder mode). 01: Inverted/falling edge. TIxFP1 falling edge active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 inverted (gated mode, encoder mode). 10: reserved, do not use this configuration. 11: Non-inverted/both edges. TIxFP1 rising and falling edges both active (capture or trigger operation in reset mode, external clock mode, or trigger mode); TIxFP1 non-inverted (gated mode). This configuration must not be used in encoder mode. Note 1: Once the LOCK level (LOCK bit in TIMx_BDTR register) is set to 3 or 2 and CC1S=00 (channel configured as output), this bit cannot be modified. Note 2: For complementary output channels, this bit is pre-loaded. If the CCPC bit in the TIMx_CR2 register is set to 1, the actual effective CC1P bit will only load the preloaded value when a COM event occurs.
0	CC1E	RW	0	Input/Capture 1 output enable CC1 channel configured as output: 0: Disabled - OC1 output is disabled, so the output level of OC1 depends on the values of MOE, OSS1, OSSR, OIS1, OIS1N, and CC1NE bits. 1: On - OC1 signal output to the corresponding output pin, its output level depends on the values of MOE, OSS1, OSSR, OIS1, OIS1N, and CC1NE bits.

				CC1 channel configured as input: This bit determines whether the counter value can be captured into the TIMx_CCR1 register. 0: Capture disabled. 1: Capture enabled. Note: For complementary output channels, this bit is preloaded. If the CC1PC bit in the TIMx_CR2 register is set, the actual effective CC1E bit will only load the preloaded value when a COM event occurs.

Table 21-2 Output control bits for complementary channels OCx and OCxN with brake function

Control bits					Output states	
MOE	OSSI	OSSR	CCxE	CCxNE	OCx output status	OCxN output state
1	x	x	0	0	Output disabled (not timer-driven), OCx=0, OCx_EN=0	Output disabled (not timer-driven), OCxN=0, OCxN_EN=0
		0	0	1	Output disabled (not timer-driven), OCx=0, OCx_EN=0	OCxREF + polarity OCxN=OCxREF XOR CCxNP, OCxN_EN=1
		0	1	0	OCxREF + polarity OCx=OCREF xor CCxP, OCx_EN=1	Output disabled (not timer-driven), OCxN=0, OCxN_EN=0
		x	1	1	OCREF + polarity + dead time, OCx_EN=1	Complement of OCREF (not OCREF) + polarity + dead time, OCxN_EN=1
		1	0	1	Off state (output enabled and at inactive level), OCx=CCxP, OCx_EN=1	OCxREF + polarity OCxN=OCxREF XOR CCxNP, OCxN_EN=1
		1	1	0	OCxREF + polarity OCx=OCxREF XOR CCxP, OCx_EN=1	Off-State (output enabled with inactive state), OCxN=CCxNP, OCxN_EN=1
0	x	0	x	x	Output disabled (not timer-driven)	
		1	0	0		
		1	0	1		
		1	1	0		
		1	1	1	Off-State (output enabled with inactive state) Asynchronous: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1 (if BRK is triggered). Subsequently (only valid when BRK is triggered), if the clock is present: after a dead time, assuming OISx and OISxN do not both correspond to the active levels of OCx and OCxN, then OCx=OISx and OCxN=OISxN.	

21.5.8. TIM16/17 counter register (TIMx_CNT) (x=16,17)

Offset address: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	0	Counter value

21.5.9. TIM16/17 prescaler register (TIMx_PSC) (x=16,17)

Offset address: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	16'b0	Prescaler value.

				The counter clock frequency (CK_CNT) is equal to $f_{CK\_PSC} / (PSC[15:0] + 1)$. The PSC contains the value to be loaded into the active prescaler register each time an update event occurs (including when the counter is cleared via the UG bit in the TIMx_EGR register, or when cleared by the trigger controller while configured in "reset mode").											

21.5.10. TIM16/17 auto-reload register (TIMx_ARR) (x=16,17)

Offset address: 0x2C

Reset value: 0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	ARR[15:0]	RW	0xFFFF	Auto-reload value The ARR contains the value to be loaded into the actual auto-reload register. The counter is blocked while the auto-reload value is null.

21.5.11. TIM16/17 Repetition Counter Register (TIMx_RCR) (x=16,17)

Offset address: 0x30

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	REP[7:0]	RW	8'b0	Value of the repetition counter When the preload function is enabled, these bits allow the user to set the update rate of the compare register (i.e., the periodic transfer from the preload register to the active register); when the update interrupt is enabled, it also affects the rate at which update interrupts are generated. Each time the down-counter REP_CNT reaches 0, an update event is generated, and the counter REP_CNT restarts counting from the REP value. Since REP_CNT only reloads the REP value when a repetition update event (UIF) occurs, the new value written to the TIMx_RCR register only takes effect upon the next repetition update event. This means that in PWM mode, (REP+1) corresponds to: - the number of PWM periods in edge-aligned mode

21.5.12. TIM16/17 Capture/Compare Register 1 (TIMx_CCR1) (x=16,17)

Offset Address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R	RW/ R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR1 [15:0]	RW	0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1). At this time, only the TIMx_CCR1 register can be read, and it cannot be programmed.

21.5.13. TIM16/17 brake and dead-time register (TIMx_BDTR) (x=16,17)

Offset Address: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as one of the break inputs is active. It is set by software or automatically depending on the AOE bit. It is acting only on the channels which are configured in output. 0: Disable OC and OCN outputs or force them to idle state, depending on the OSSI bit; 1: If the corresponding enable bits (CCxE, CCxNE bits in the TIMx_CCER register) are set, the OC and OCN outputs are enabled.
14	AOE	RW	0	Automatic output enable 0: MOE can be set only by software; 1: MOE can be set by software or automatically at the next update event (if none of the break inputs is active). Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified.
13	BKP	RW	0	Break polarity 0: Break input BRK is active low; 1: Break input BRK is active high Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified.
12	BKE	RW	0	Break enable 0: Break function disabled 1: Break function enabled Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified.
11	OSSR	RW	0	Off-state selection for Run mode

				This bit is used when MOE=1 on channels having a complementary output which are configured as outputs. OSSR bit is not implemented if no complementary output is implemented in the timer. Refer to the detailed description of OC/OCN enable. 0: In inactive state, disable OC/OCN outputs (OC/OCN enable output signal = 0); 1: When in the inactive state, once CCxE=1 or CCxNE=1, the OC/OCN output is enabled and outputs the inactive level. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 2, this bit cannot be modified.
10	OSSI	RW	0	Off-state selection for Idle mode This bit is used when MOE=0 and on channels configured as outputs. Refer to the detailed description of OC/OCN enable. 0: In inactive state, disable OC/OCN outputs (OC/OCN enable output signal = 0); 1: In inactive state, OC/OCN first outputs its inactive level once CCxE=1 or CCxNE=1. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 2, this bit cannot be modified.
9:8	LOCK[1:0]	RW	2'b0	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: LOCK Level 1, DTG, BKE, BKP, AOE bits in TIMx_BDTR register and OISx and OISxN bits in TIMx_CR2 register can no longer be written. 10: Lock level 2, the bits in lock level 1 cannot be written, nor can the CC polarity bits (CCxP/CCxNP bits in the TIMx_CCER register, once the related channel is set as output via the CCxS bit) or the OSSR/OSSI bits. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIMx_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. Note: The LOCK bits can be written only once after the reset. Once the TIMx_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG[7:0]	RW	8'b0	Dead-time generator setup. This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. $DTG[7:5]=0xx \Rightarrow DT=DTG[7:0] \times T_{dtg}, \quad T_{dtg} = T_{DTS};$ $DTG[7:5]=10x \Rightarrow DT=(64+DTG[5:0]) \times T_{dtg},$ where $T_{dtg} = 2 \times T_{DTS};$ $DTG[7:5]=110 \Rightarrow DT=(32+DTG[4:0]) \times T_{dtg},$ where $T_{dtg} = 8 \times T_{DTS};$ $DTG[7:5]=111 \Rightarrow DT=(32+DTG[4:0]) \times T_{dtg},$ where $T_{dtg} = 16 \times T_{DTS};$ Example if $T_{DTS}=125ns$ (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps

				Note: Once the LOCK level (LOCK bit in the TIM15_BDTR register) is set to 1, 2, or 3, these bits cannot be modified.

21.5.14. TIM16/17 input selection register (TIMx_TISEL) (x=16,17)

Offset address: 0x5C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3:0	TISEL	RW	0	T11 input selection. 0000: TIMx_CH1 0001: Reserved 0010: MCO 0011: HSE_DIV32 0100: RTC alarm wakeup 0101: LSE_CSS 0110: LSI Other: Reserved

21.5.15. TIM16/17 alternate function option register 1 (TIMx_AF1) (x=16,17)

Offset address: 0x60

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	BKCOMP2 P	BKCOMP1 P	BKIN P	Res.	Res.	Res.	Res.	Res.	Res.	BKCOMP2 E	BKCOMP1 E	BKIN E
-	-	-	-	RW	RW	RW	-	-	-	-	-	-	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	BKCOMP2P	RW	0	Brake input polarity control from Comparator 2. This bit selects the brake input polarity from comparator 2 and must be configured together with the BKP polarity bit. 0: Comparator 2 brake input polarity not inverted (active low when BKP=0, active high when BKP=1) 1: Comparator 2 brake input polarity inverted (active high when BKP=0, active low when BKP=1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
10	BKCOMP1P	RW	0	Brake input polarity control from Comparator 1. This bit selects the brake input polarity from comparator 1 and must be configured together with the BKP polarity bit. 0: Comparator 1 brake input polarity not inverted (active low when BKP=0, active high when BKP=1) 1: Comparator 1 brake input polarity inverted (active high when BKP=0, active low when BKP=1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
9	BKINP	RW	0	BKIN input polarity. This bit selects the alternate function of the BKIN input polarity and must be configured at the same time as the BKP polarity bit

				0: BKIN input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: BKIN input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
8:3	Reserved	-	-	Reserved
2	BKCMP2E	RW	0	Comparator 2 brake input enable. Comparator 2 brake input is ORed with other brake sources as the brake input. 0: Comparator 2 brake input off 1: Comparator 2 brake input enabled Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
1	BKCMP1E	RW	0	Comparator 1 brake input enable. Comparator 1 brake input is ORed with other brake sources as the brake input. 0: Comparator 1 brake input disabled 1: Comparator 1 brake input enabled Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
0	BKINE	RW	1	BKIN input enabled. Comparator 1 brake BKIN input is ORed with other brake sources as the brake input. 0: BKIN input off 1: BKIN input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified.

21.5.16. TIM16/17 DMA control register (TIMx_DCR) (x=16,17)

Offset address: 0x3DC

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12:8	DBL[4:0]	RW	0 0000	DMA burst length These bits define the DMA transfer length (when reading or writing to the address of the TIMx_DMAR register, the timer performs a continuous transfer), i.e., define the number of bytes to be transferred: 00000: 1 transmission 00001: 2 transmissions 00010: 3 transfers 10001: 18 transfers
7:5	Reserved	-	-	Reserved
4:0	DBA[4:0]	RW	0 0000	DBA[4:0]: DMA base address These bits define the base address of the DMA in continuous mode (when read/write access are done through the TIMx_DMAR address). DBA is defined as an offset starting from the address of the TIMx_CR1 register. 00000: TIMx_CR1 00001: TIMx_CR2

				00011: TIMx_DIER

21.5.17. TIM16/17 DMA continuous transfer address register (TIMx_DMAR) (x=16,17)

Offset address: 0x3E0

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[31:0]	RW	0	DMA register for burst accesses A read or write operation to the DMAR register accesses the register located at the address: TIMx_CR1 address + (DBA + DMA pointer) x4, where: TIMx_CR1 address is the address of the control register 1; DBA is the DMA baseaddress configured in TIMx_DCR register; DMA index is automatically controlled by the DMA transfer, and ranges from 0 to DBL (DBL configured in TIMx_DCR).

22. Basic timers (TIM6/TIM7)

22.1. TIM6/TIM7 introduction

The basic timer TIM6 contains a 32-bit auto-reload counter, while TIM7 contains a 16-bit auto-reload counter. This counter is driven by a programmable prescaler.

These timers can not only be used as general-purpose timers to generate time bases but can also be specifically used to drive digital-to-analog converters (DACs). In fact, these timers are internally connected to the DAC and capable of driving the DAC through their trigger outputs.

The timers are completely independent, and do not share any resources.

22.2. TIM6/TIM7 Key Features

The main functions of the TIM6 and TIM7 timers include:

- 32/16-bit auto-reload counter (TIM6 configured as 32-bit counting)
- 16-bit programmable (can be modified in real time) prescaler, the frequency division coefficient of the counter clock frequency is any value between 1 and 65535
- DAC synchronous circuit is triggered.
- Generate interrupt/DMA when an update event (counter overflow) occurs

22.3. TIM6/TIM7 Functional Description

22.3.1. Functional block diagram

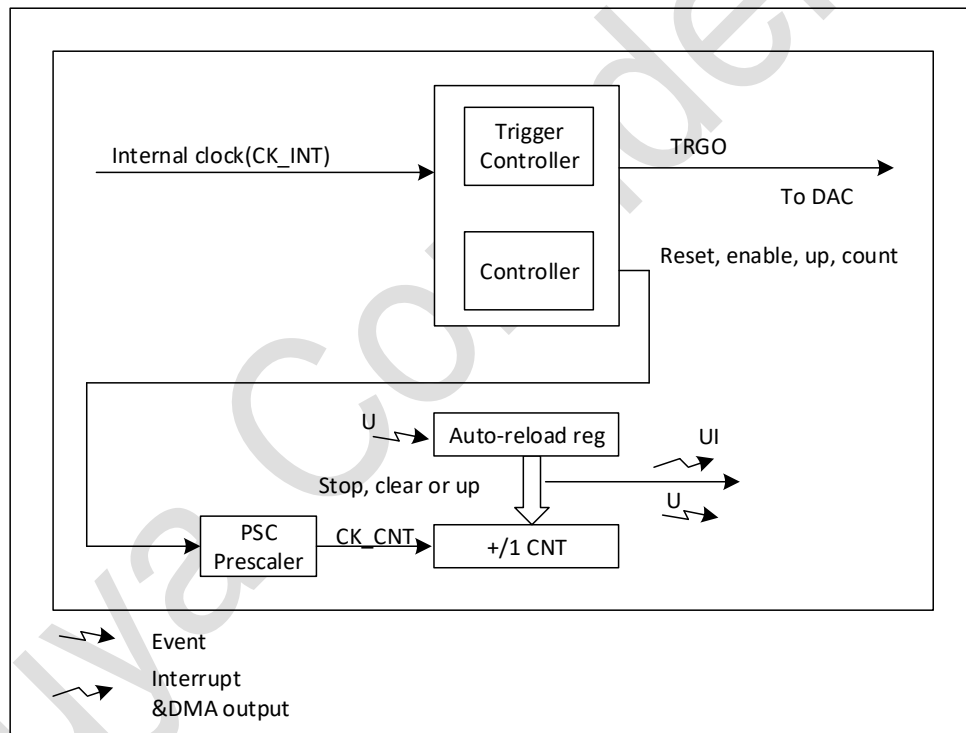


Figure 22-1 TIM6/TIM7 General Block Diagram

22.3.2. Time-base unit

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. An update event is generated when the counter reaches the overflow value (or underflow value in downcounting) and the UDIS bit in the TIMx_CR1 register is 0. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in the TIMx_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR1 register.

22.3.2.1. Prescaler description

The prescaler can divide the clock frequency of the counter by any value between 1 and 65536. It is a 16-bit counter controlled based on a 16-bit register (in the TIMx_PSC register). It can be changed

on the fly as this control register is buffered. The new prescalation parameters will be adopted when the next update event comes.

Figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly:

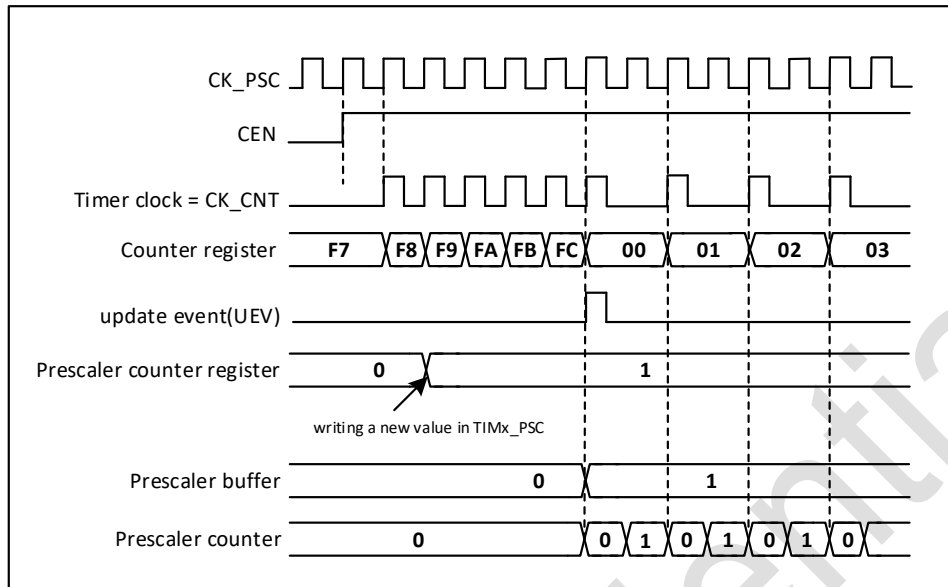


Figure 22-2 Counter timing diagram when prescaler division changes from 1 to 2

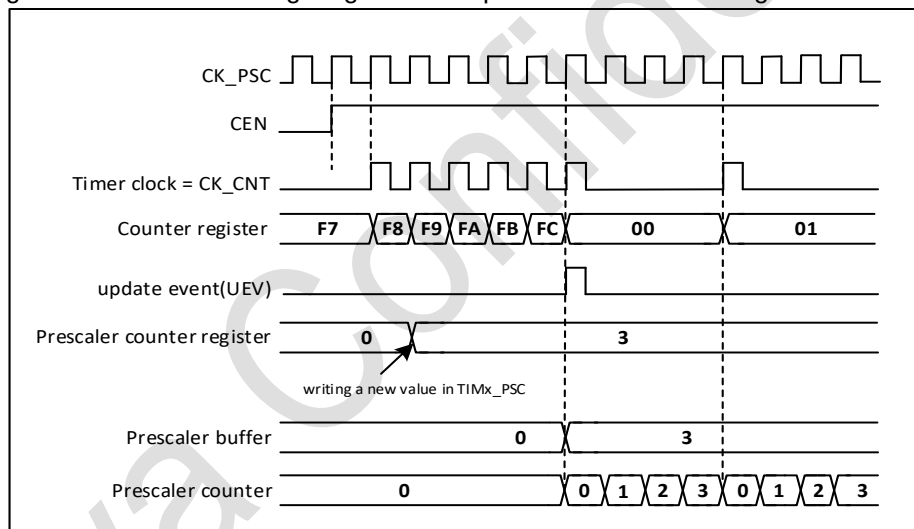


Figure 22-3 Counter timing diagram when prescaler division changes from 1 to 4

22.3.3. Counting mode (upcounting)

The counter counts from 0 to the auto-reload value (content of TIMx_ARR), then restarts counting from 0 and generates a counter overflow event.

Each counter overflow generates an update event. Setting the UG bit in the TIMx_EGR register (either by software or using a slave mode controller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. No update event will be generated until the UDIS bit is cleared '0'. Even so, when an update event should occur, the counter is still cleared '0', and the counter inside the prescaler is also cleared '0' (but the value of the prescaler remains unchanged).

Furthermore, if the URS bit in the TIMx_CR1 register is set (select the update request source), an update event UEV can be generated by setting the UG bit, but the UIF flag bit will not be set (i.e., no interrupt or DMA request will be generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, the following registers are all updated, and the hardware simultaneously sets the update flag bit (UIF bit in the TIMx_SR register) (depending on the URS bit):

- The auto-reload shadow register is reloaded with the value from the preload register (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

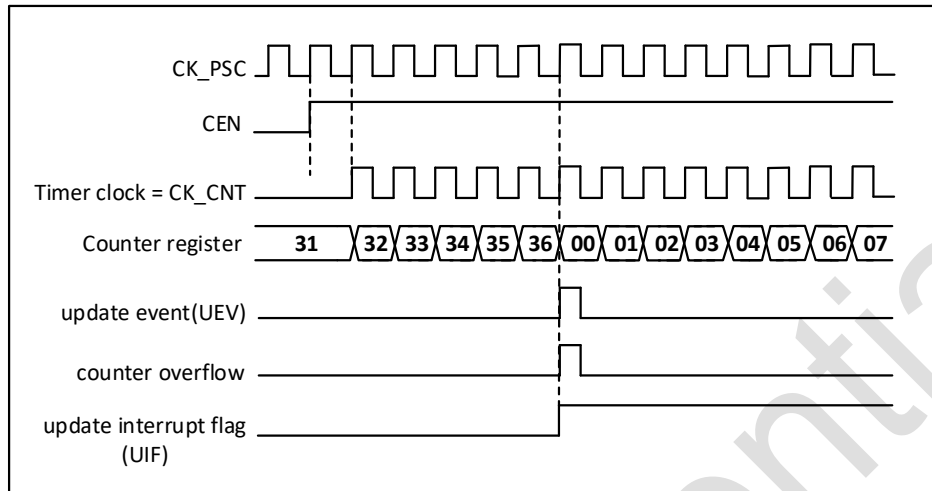


Figure 22-4 Counter timing diagram, internal clock divided by 1

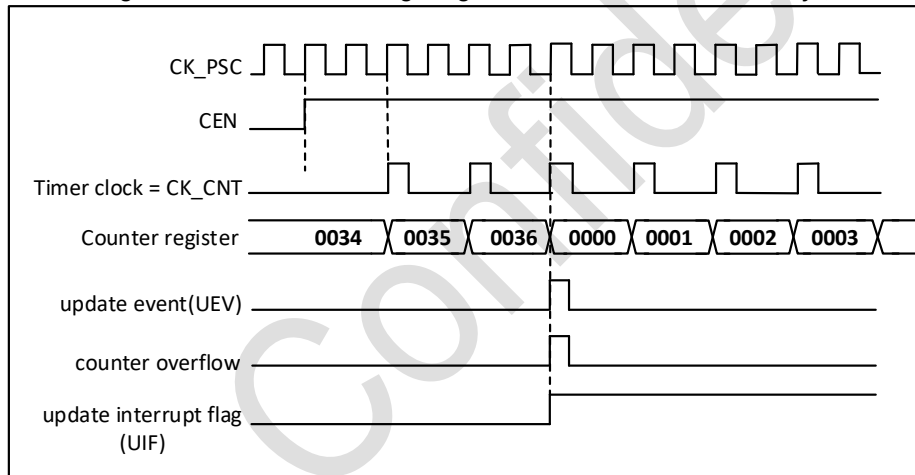


Figure 22-5 Counter timing diagram, internal clock divided by 2

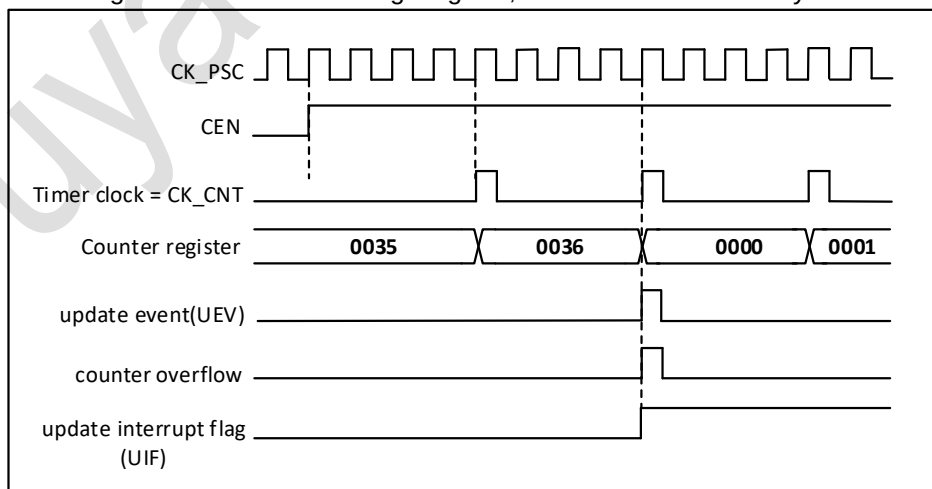


Figure 22-6 Counter timing diagram, internal clock divided by 4

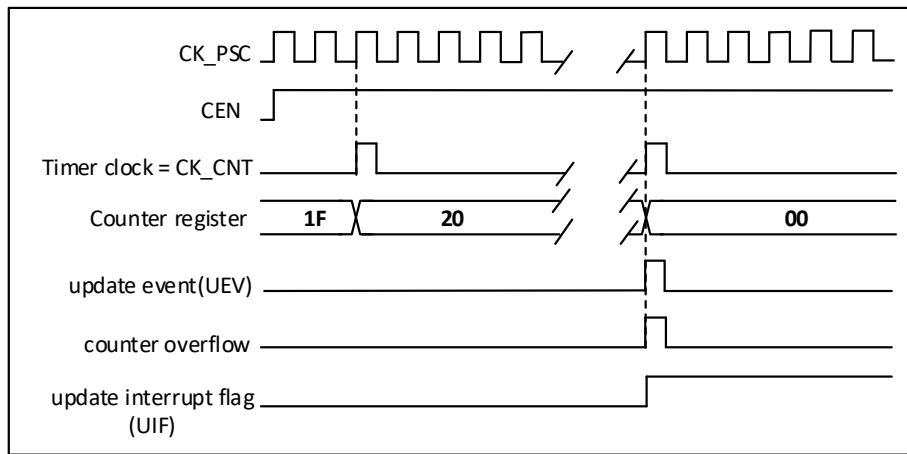


Figure 22-7 Counter timing diagram, internal clock divided by N

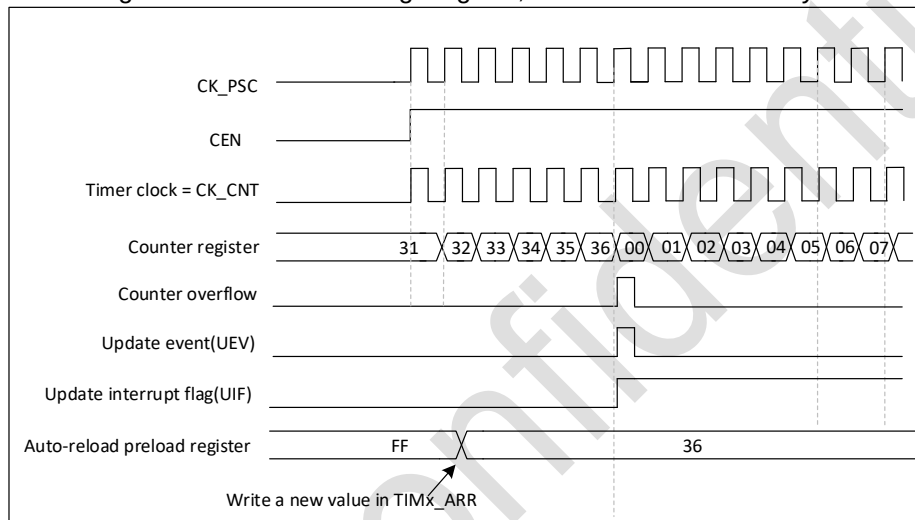


Figure 22-8 Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

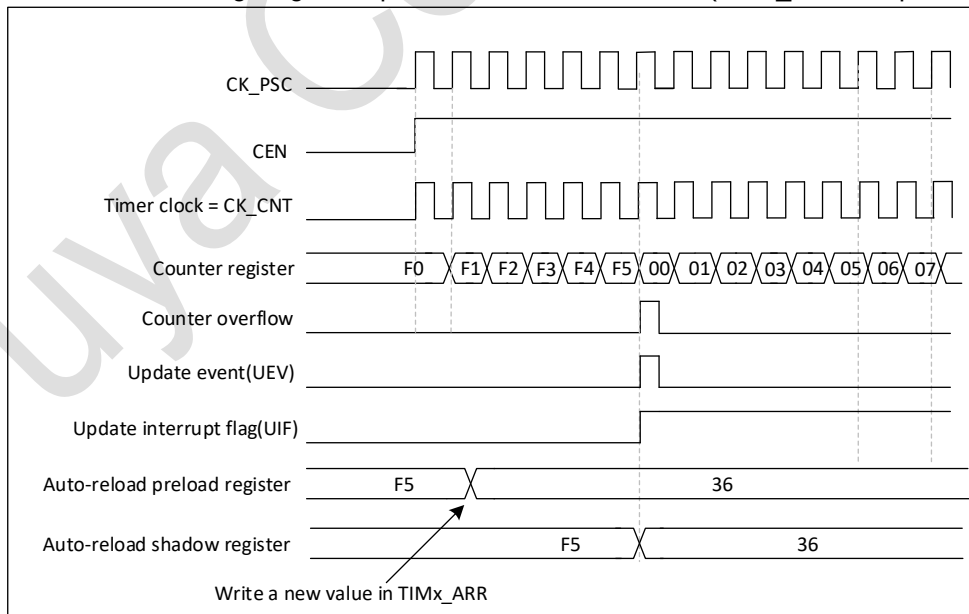


Figure 22-9 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

22.4. TIM6/TIM7 Debug Mode

When the microcontroller enters debug mode (Cortex®-M0+ core halted), depending on the setting of DBG_TIMx_STOP in the DBGMCU module, the TIM6 and TIM7 counters can either continue normal operation or stop.

22.5. TIM6/TIM7 Registers

22.5.1. TIM6/TIM7 Control Register 1 (TIMx_CR1)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
								RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered; 1: TIMx_ARR register is loaded into buffer.
6:4	Reserved	-	-	Reserved
3	OPM	RW	0	One-pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source Software selects the source of the UEV event through this bit. 0: If an update interrupt or DMA request is enabled, either of the following events generates an update interrupt or DMA request: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller 1: If the update interrupt or DMA request is enabled, only counter overflow/underflow generates an update interrupt or DMA request.
1	UDIS	RW	0	Update disable Software enables/disables UEV event generation through this bit. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller Shadow registers are loaded with preload values. 1: UEV disabled. The update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). The counter and prescaler are reinitialized if the UG bit is set or a hardware reset is issued by the slave mode controller.
0	CEN	RW	0	Counter enabled 0: Disable counter; 1: Enable counter. Note: The gated mode can only work after the CEN bit is set by software. The trigger mode can automatically set the CEN bit to 1 through hardware. In one-pulse mode, CEN is automatically cleared when an update event occurs.

22.5.2. TIM6/TIM7 Control Register 2 (TIMx_CR2)

Offset address: 0x04

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		MMS[2:0]		Res.	Res.	Res.	Res.
									RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6:4	MMS[2:0]	RW	3'b0	Master mode selection These 3 bits are used to select synchronization information (TRGO) to be sent to the slave timer in master mode. The combination is as follows: 000: Reset --- The UG bit of the TIMx_EGR register is used as a trigger output (TRGO). If the reset is generated by the trigger input (the slave mode controller is in reset mode), the signal on the TRGO will have a delay relative to the actual reset. 001: Enable --- The counter enable signal CNT_EN is used as the trigger output (TRGO). This trigger output can be used to start multiple timers simultaneously or control slave timers over a period of time. The counter enable signal is generated by the logical OR combination of the CEN control bit and the trigger input in gated mode. 010: Update --- The update event is selected as the trigger input (TRGO). For instance a master timer can then be used as a prescaler for a slave timer.
3:0	Reserved	-	-	Reserved

22.5.3. TIM6/TIM7 DMA/Interrupt Enable Register (TIMx_DIER)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIE
							RW								RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8	UDE	RW	0	Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled
7:1	Reserved	-	-	Reserved
0	UIE	RW	0	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

22.5.4. TIM6/TIM7 Status Register (TIMx_SR)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	Reserved
0	UIF	RC_W0	0	This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: An update event has occurred. This bit is set by hardware when the registers are updated:

				- If UDIS=0 in the TIMx_CR1 register, an update event is generated when the counter overflows; - If UDIS=0 and URS=0 in the TIMx_CR1 register, an update event is generated when UG=1 in the TIMx_EGR register (software reinitializes CNT);

22.5.5. TIM6/TIM7 Event Generation Register (TIMx_EGR)

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UG
															W

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	Reserved
0	UG	W	0	Update generation This bit is set to '1' by the software, and '0' is automatically cleared by the hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the counter of the prescaler is also cleared '0' (but the prescaler coefficient remains unchanged).

22.5.6. TIM6/TIM7 counter register (TIMx_CNT)

Offset address: 0x24

Reset value: 0x0000 0000

TIM6:

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

Bit	Name	R/W	Reset Value	Function
31:0	CNT[31:0]	RW	32'b0	Counter value

TIM7:

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	16'b0	Counter value

22.5.7. TIM6/TIM7 prescaler register (TIMx_PSC)

Offset address: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	16'b0	The value of the prescaler.

				The counter clock frequency (CK_CNT) is equal to $f_{CK\_PSC} / (PSC[15:0] + 1)$. PSC contains the value to be loaded into the active pre-scaler register on each update event (including when the counter is cleared via the UG bit in the TIMx_EGR register, or when cleared by the trigger controller when configured in "reset mode").

22.5.8. TIM6/TIM7 auto-reload register (TIMx_ARR)

Offset address: 0x2C

Reset value: 0xFFFF FFFF

TIM6:

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

Bit	Name	R/W	Reset Value	Function
31:0	ARR[31:0]	RW	32'hFFFF FFFF	The value for auto-reload. ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

TIM7:

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	ARR[15:0]	RW	16'hFFFF	Automatic Reload Value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

23. Pulse Width Modulation (PWM)

23.1. PWM main features

The main features of PWM include:

- 16-bit up, down, or up/down auto-reload counter
- Programmable cycle and duty cycle
- Supports counting when the external input clock is a frequency multiple of the system clock
- A programmable prescaler that allows the clock frequency of the counter to be divided from 1 to 65,536
- Support write protection for important registers
- Up to 4 independent channels
- Support complementary outputs with programmable dead time for 2 channels.
- Output polarity configurable
- Supports edge alignment and center alignment
- Brake input can set the output signal of the timer to a user-selectable safety configuration
- Support 1 brake input
- Supports external clock counting
- Support DMA function
- Interrupt event
 - Update: Counter overflow up, down
 - Output compare
 - Break input

23.2. PWM functional description

23.2.1. Functional block diagram

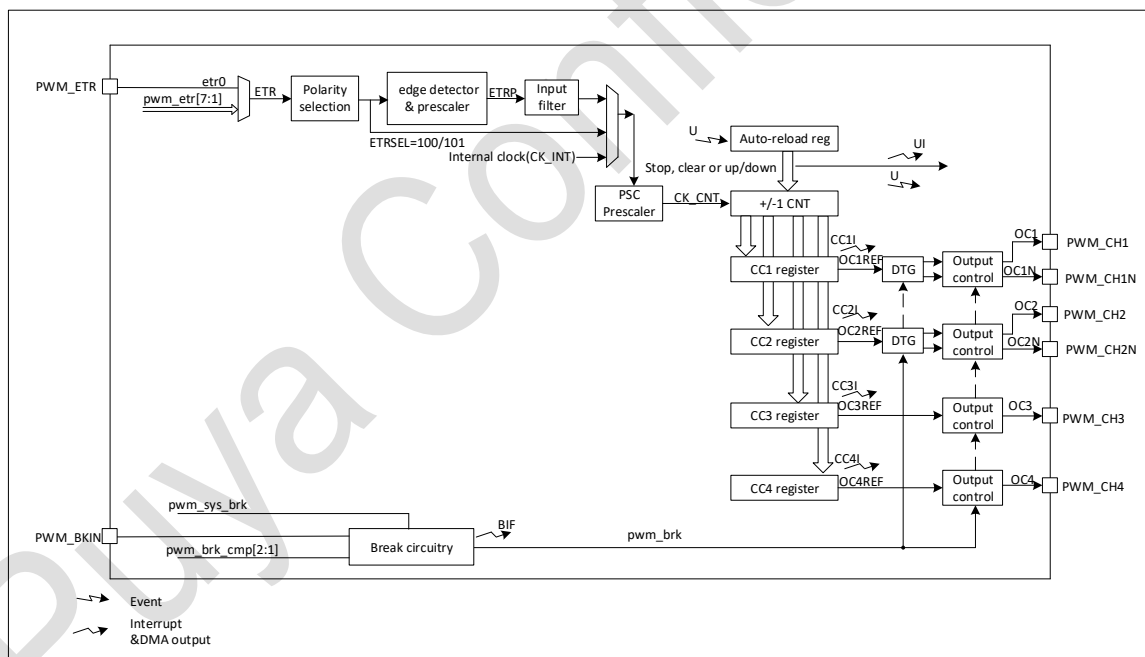


Figure 23-1 PWM block diagram

23.2.2. Counting modes

23.2.2.1. Upcounting mode

In upcounting mode, the counter counts from 0 up to the auto-reload value (content of PWM_ARR), then restarts from 0 and generates a counter overflow event.

Setting the UG bit in the PWM_EGR register (via software or using the slave mode controller) can also generate an update event.

The update event can be disabled by setting the UDIS bit in the PWM_CR1 register; this prevents updating the shadow registers when writing new values to the preload registers. No update event will be generated until the UDIS bit is cleared '0'. Even so, when an update event should occur, the counter is still cleared '0', and the counter inside the prescaler is also cleared '0' (but the value of the prescaler remains unchanged).

Additionally, if the URS bit (update request source selection) in the PWM_CR1 register is set, setting the UG bit can generate an update event (UEV) without setting the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the following registers are updated, and the hardware simultaneously sets the update flag (UIF bit in the PWM_SR register) based on the URS bit:

- The auto-reload shadow register is reloaded with the value from the preload register (PWM_ARR).
- The prescaler buffer is loaded with the value from the preload register (content of the PWM_PSC register).

The following figures show examples of counter behavior at different clock frequencies when PWM_ARR=0x36.

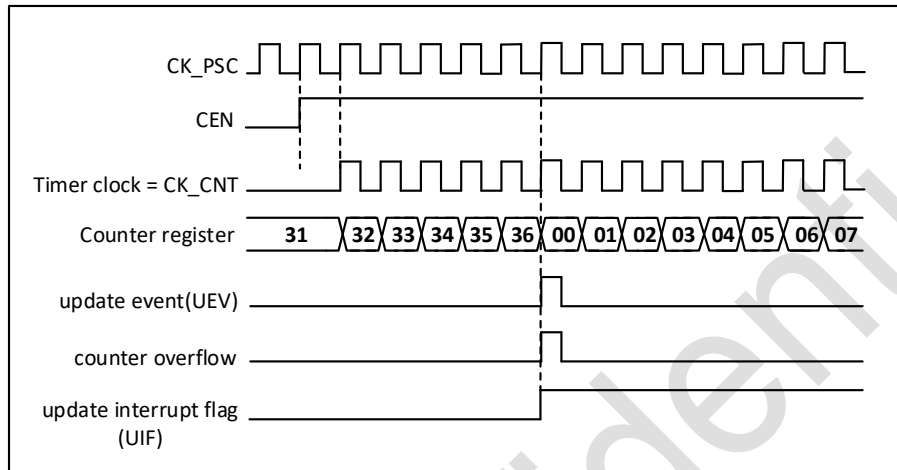


Figure 23-2 counter timing diagram with internal clock division factor of 1

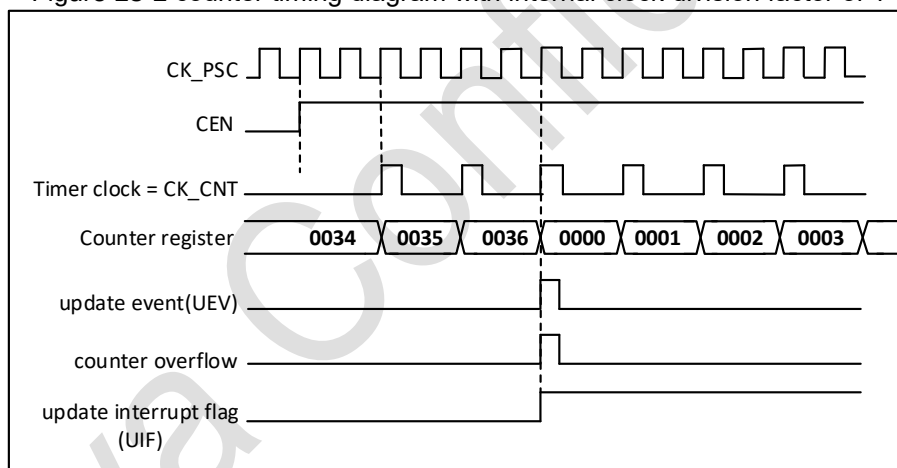


Figure 23-3 counter timing diagram with internal clock division factor of 2

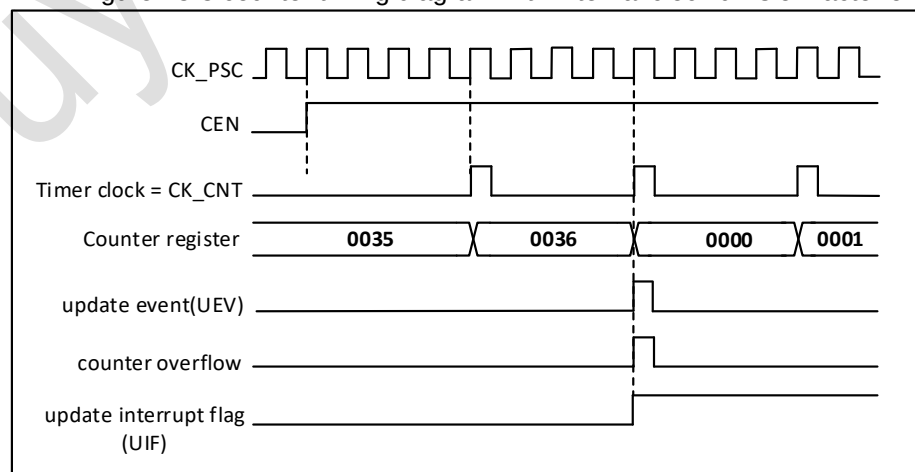


Figure 23-4 counter timing diagram with internal clock division factor of 4

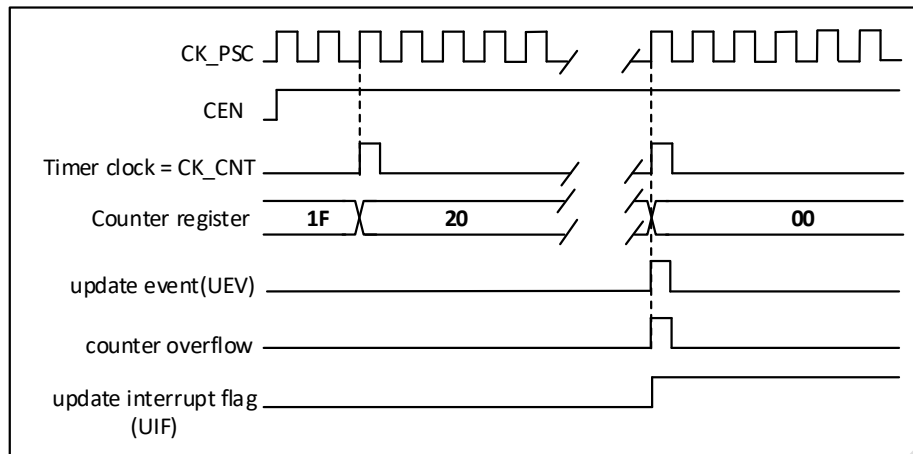


Figure 23-5 Counter timing diagram with internal clock division factor of N

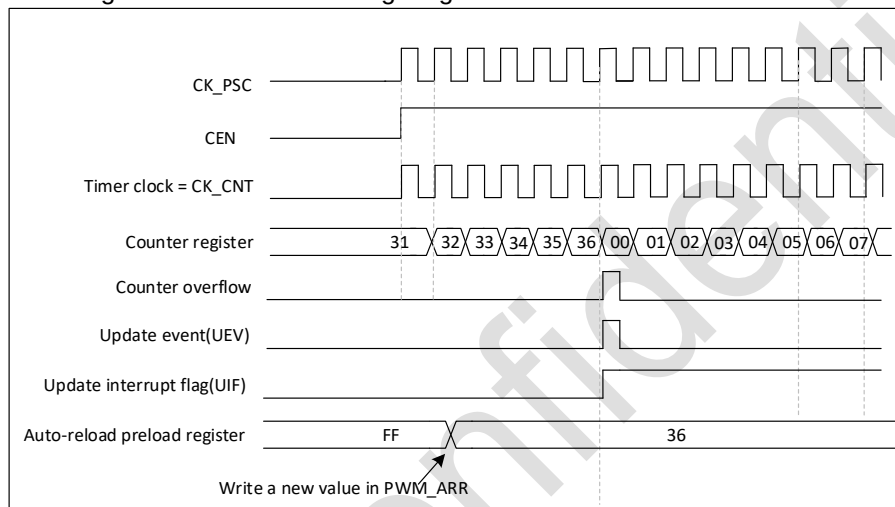


Figure 23-6 Counter timing diagram for update events when ARPE=0 (PWM_ARR not preloaded)

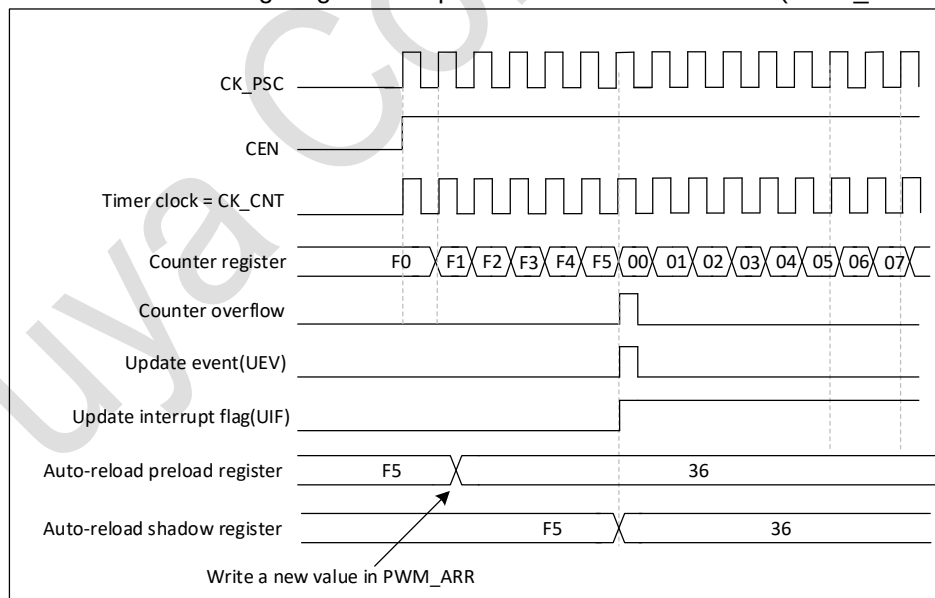


Figure 23-7 Counter timing diagram for update events when ARPE=1 (PWM_ARR preloaded)

23.2.2.2. Downcounting mode

In downcounting mode, the counter counts down from the auto-reload value (content of PWM_ARR) to 0, then restarts from the auto-reload value and generates a counter underflow event. Each count underflow generates an update event.

When an update event occurs, all the following registers are updated:

- The cycle shadow register is reset to the value of the cycle register (the contents of the PWM_ARR register).
- The buffer of the prescaler is placed into the value of the prescaler register (the content of the PWM_PSC register).

The following are some examples of the operation of the counter at different clock frequencies when PWM_ARR = 0x36.

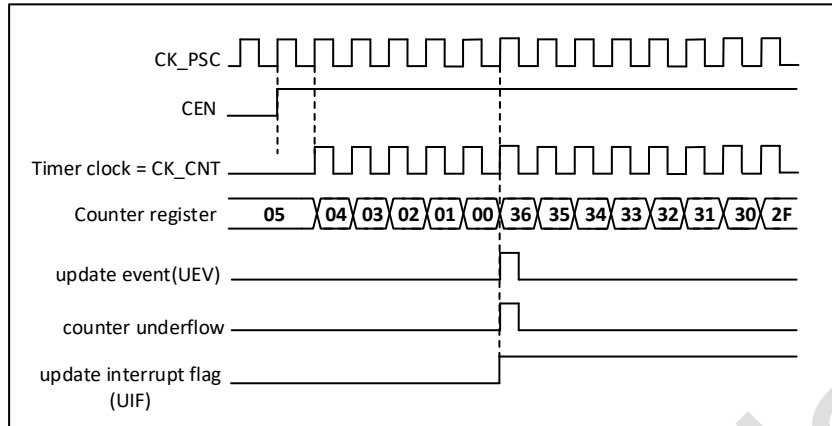


Figure 23-8 Counter timing diagram, internal clock divided by 1

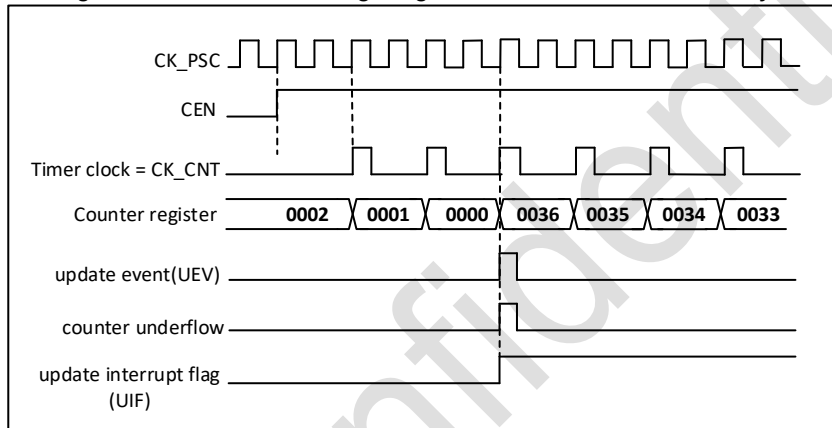


Figure 23-9 counter timing diagram with internal clock division factor of 2

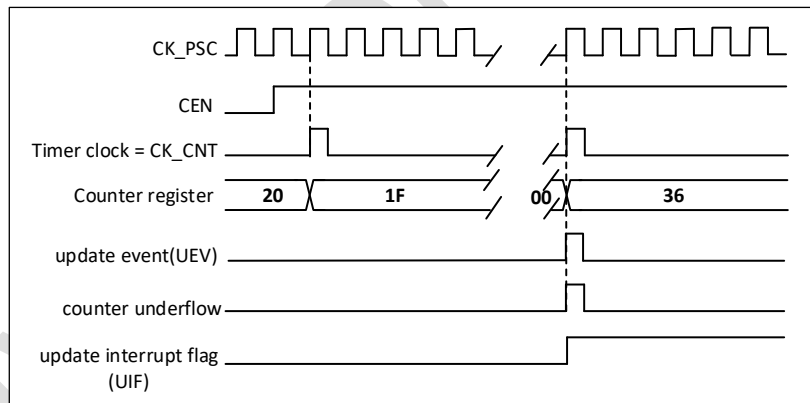


Figure 23-10 Counter timing diagram, internal clock divided by N

23.2.2.3. Center-aligned mode

In center-aligned mode, the counter starts counting from 0 up to the auto-reload value (PWM_ARR register) -1, generating a counter overflow event, then counts down to 1 and generates a counter underflow event; then starts counting again from 0.

Center-aligned mode can be obtained by configuring the CMS bits in the PWM_CR1 register to a value other than '00'.

In this mode, the DIR direction bit in the PWM_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter.

An update event can be generated every time a count overflow and every time a count underflow. The counter then starts counting again from 0, and the prescaler internal counter also starts counting again from 0.

When an update event occurs, all the registers are updated:

- The cycle shadow register is reset to the value of the cycle register (the contents of the PWM_ARR register).
- The buffer of the prescaler is placed into the value of the prescaler register (the content of the PWM_PSC register).

Here are some examples of how the counter operates at different clock frequencies:

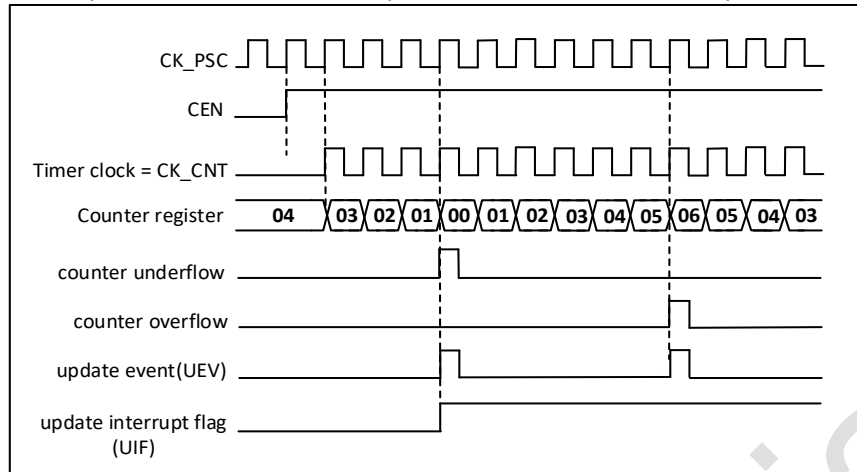


Figure 23-11 Counter timing diagram, internal clock division factor is 1, PWM_ARR = 0x6

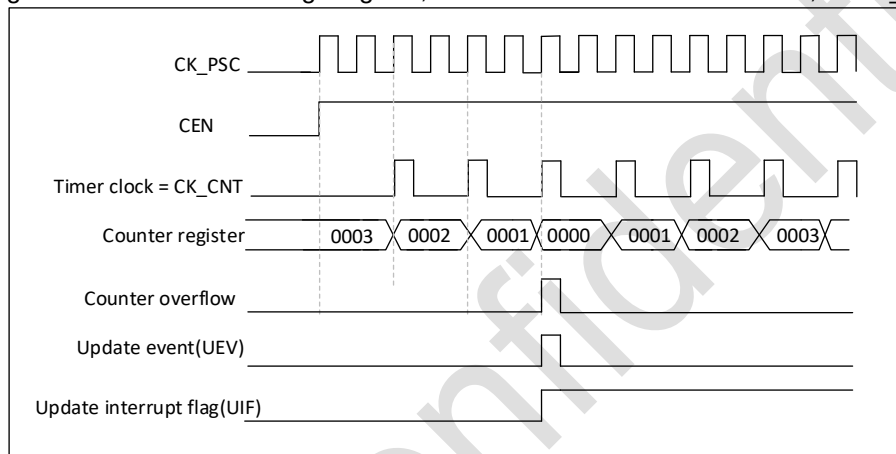


Figure 23-12 Counter timing diagram, internal clock divided by 2

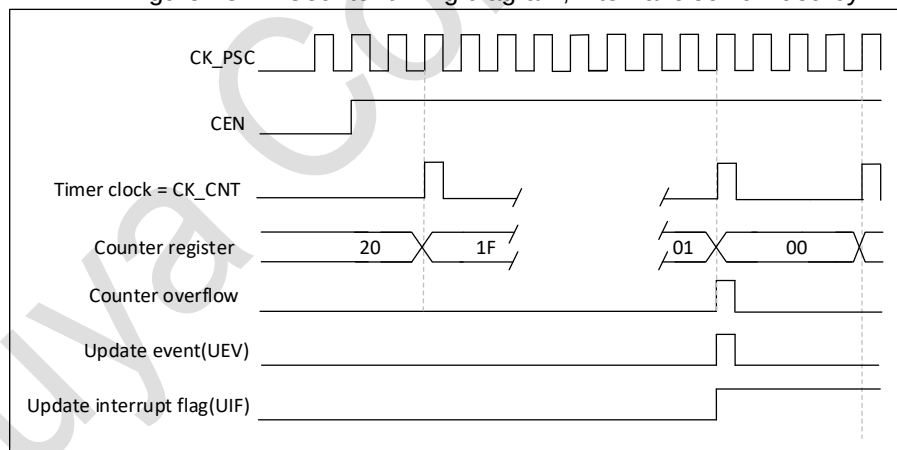


Figure 23-13 Counter timing diagram, internal clock divided by N

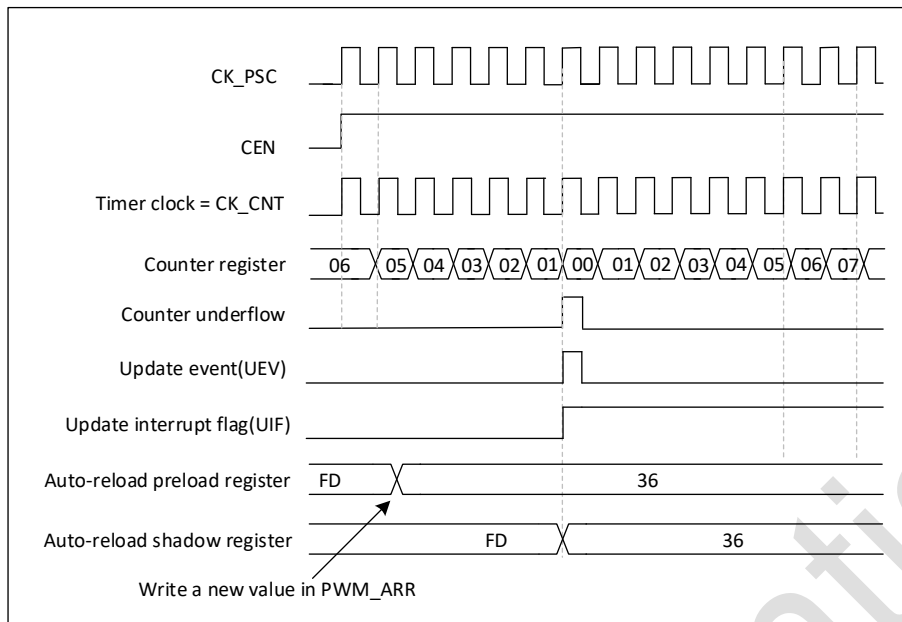


Figure 23-14 Counter timing diagram, update event with ARPE=1 (counter underflow)

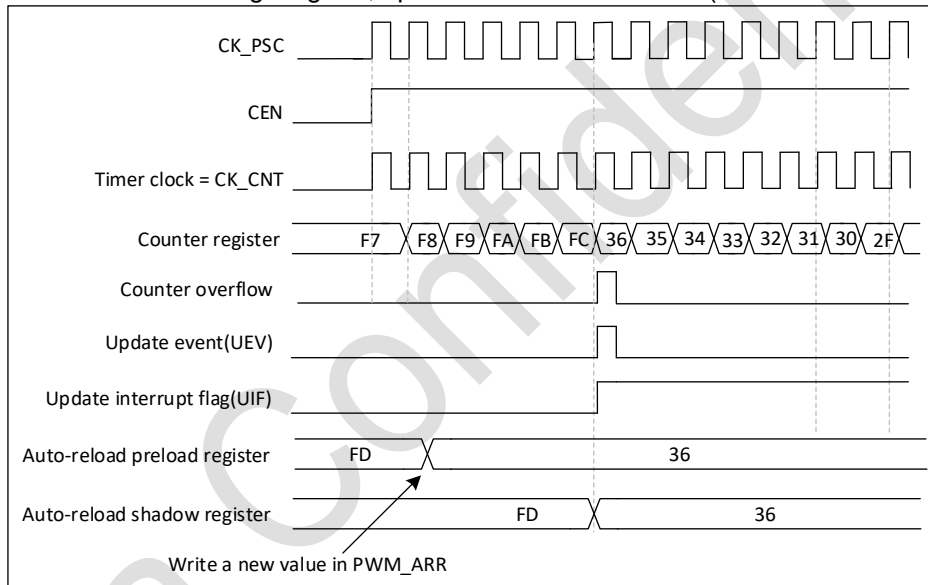


Figure 23-15 Counter timing diagram, update event when ARPE=1 (counter overflow)

23.2.3. Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode: External trigger input ETR

23.2.3.1. Internal clock source

As long as the CEN bit is written as '1', the clock of the prescaler is provided by the internal clock CK_INT.

The diagram below shows the operation of the control circuit and up counter in general mode, without the prescaler.

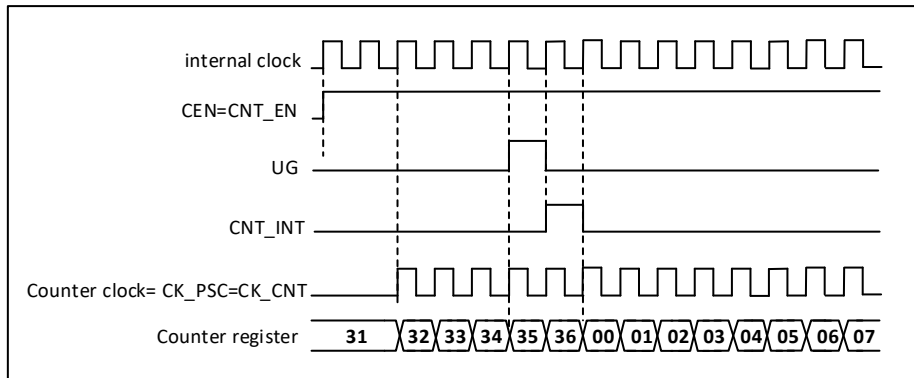


Figure 23-16 Control circuit in normal mode, internal clock divided by 1

23.2.3.2. External clock source mode

This mode is selected by making $ECE = 1$ in the PWM_SMCR register, and the counter can externally trigger each rising or falling edge of the ETR to count. When ETRSEL selects 100/101, the external clock can be higher than CK_INT, so frequency division and filtering operations cannot be performed.

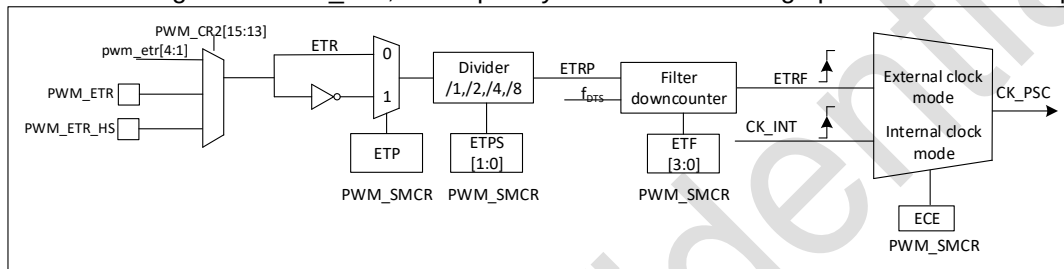


Figure 23-17 External trigger input block diagram

For example, to configure an up-counter that counts once every 2 rising edges under PWM_ETR_HS, follow these steps:

1. Select the external trigger source as PWM_ETR_HS by configuring ETRSEL[2:0]=101 in the PWM_CR2 register;
2. In this example, the filter cannot be enabled, configure ETF[3:0]=0000 in the PWM_SMCR register;
3. Set the prescaler, configure ETPS[1:0]=01 in the PWM_SMCR register;
4. Select the rising edge of the ETR input, configure ETP=0 in the PWM_SMCR register;
5. Enable external clock mode, write ECE=1 in the PWM_SMCR register;
6. Start the counter, write CEN=1 in the PWM_CR1 register;

The counter counts at every 2 ETR rising edges.

The delay between the rising edge of the ETR and the actual clock of the counter depends on the resynchronization circuit of the ETRP signal. If ETRSEL = 100/101, there is no delay between the rising edge of the ETR and the actual clock of the counter.

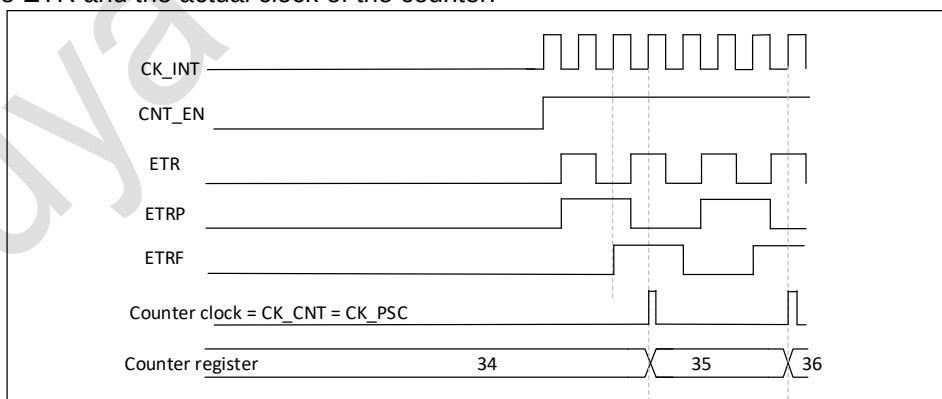


Figure 23-18 Control circuit in external clock mode

23.2.4. Output compare mode

Each comparison channel is surrounded by a comparison register (including a shadow register), including an output portion (comparator and output control).

The following two figures are a comparative channel overview.

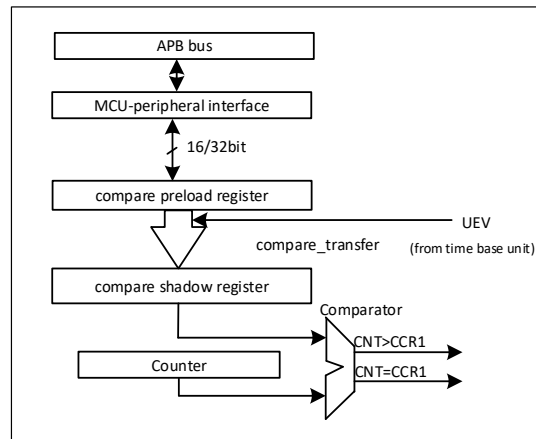


Figure 23-19 Compare the main circuit of channel 1

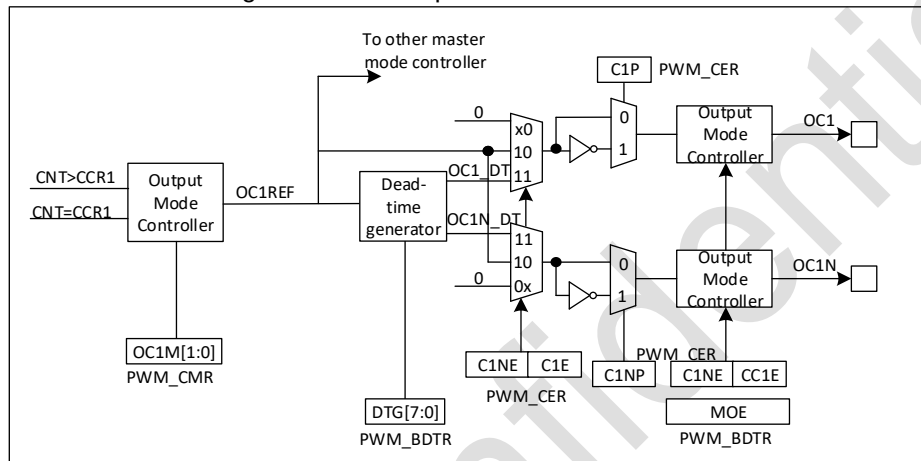


Figure 23-20 Output section of the comparison channel (Channel 1)

The comparison module consists of a preload register and a shadow register. Write and read always access the preload register.

In comparison mode, the contents of the preload register are copied into the shadow register, and the contents of the shadow register are then compared with the counter.

23.2.5. PWM mode

The module can generate a signal whose frequency is determined by the PWM_ARR register and whose duty cycle is determined by the PWM_CCRx register.

The OCxM bit in the PWM_CMR register is written to "10" (PWM Mode 1) or "11" (PWM Mode 2), and each OCx output channel can be independently set to generate a PWM.

If you want to change the cycle or duty cycle of the PWM, the new values will not take effect until the next cycle after you modify the PWM_ARR and PWM_CCRx registers.

The polarity of OCx can be set by software in the CxP bit in the PWM_CER register, and it can be set to high active or low active. The output of OCx is enabled by CxE control in PWM_CER.

In the PWM mode (Mode 1 or Mode 2), PWM_CNT and PWM_CCRx are always compared (depending on the counting direction of the counter) to determine whether $\text{PWM\_CCR}_x \leq \text{PWM\_CNT}$ or $\text{PWM\_CNT} \leq \text{PWM\_CCR}_x$ is consistent.

Depending on the state of the CMS bit in the PWR_CR1 register, the timer can generate either edge-aligned PWM signals or center-aligned PWM signals.

23.2.5.1. PWM edge-aligned mode

Upcounting configuration

Increment counting is performed when the DIR bit in the PWM_CR1 register is low. In the following example, we consider PWM mode 1. When $\text{PWM\_CNT} < \text{PWM\_CCR}_x$, PWM is an active level defined by CxP, otherwise it is an invalid level.

If the comparison value in PWM_CCRx is greater than the auto-reload value (PWM_ARR), the PWM output remains '1'. If the comparison value is 0, the output remains '0'.

The following figure shows an example of a PWM waveform with edge alignment at $\text{PWM\_ARR} = 8$.

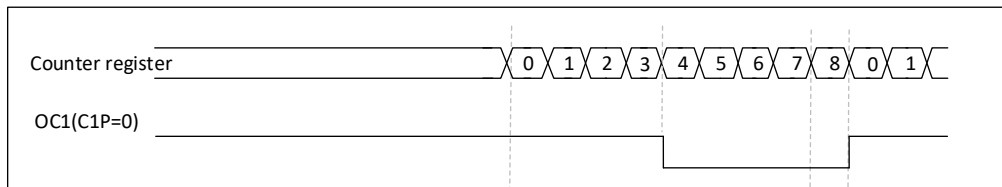


Figure 23-21 Edge-aligned PWM output, increment (ARR=8)

Downcounting configuration

Decremental counting is performed when the DIR bit in the PWM_CR1 register is high.

In PWM mode 1, the output is an invalid level when $PWM\_CNT > PWM\_CCRx$, and an active level otherwise. If the comparison value in PWM_CCRx is greater than the auto-reload value in PWM_ARR , the output remains '1'. 0% PWM is not possible in this mode.

23.2.5.2. PWM center-aligned mode

Center-aligned mode is enabled when the CMS bit in the PWM_CR1 register is set to 1. At this time, the count direction bit (DIR) in the PWM_CR1 register is updated by hardware. Do not modify it by software.

The following figure shows some examples of center-aligned PWM waveforms:

- $PWM\_ARR = 8$
- PWM mode is the PWM mode 1
- CMS=01 in the PWM_CR1 register

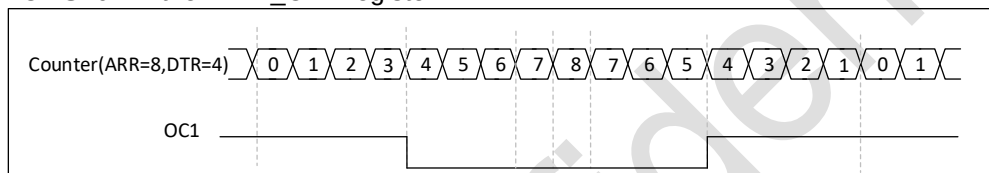


Figure 23-22 Center-aligned mode PWM output (ARR=8)

Hints on using center-aligned mode:

- When entering center-aligned mode, the current increment/decrement counting configuration is used; this means whether the counter increments or decrements depends on the current value of the DIR bit in the PWM_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- It is not recommended to rewrite the counter while running in center-aligned mode, as this may produce unpredictable results. Specifically:
 - If the value of the write counter is greater than the value of the auto-reload ($PWM\_CNT > PWM\_ARR$), the direction is not updated. For example, if the counter is incrementing, it will continue to increment.
 - If 0 or the value of PWM_ARR is written to the counter, the counting direction is updated, but no update event (UEV) is generated.
- The safest way to use center-aligned mode is to generate a software update (set the UG bit in the PWM_EGR register) before starting the counter and avoid modifying the counter value during counting.

23.2.6. Complementary outputs and dead-time insertion

The PWM can output two complementary signals and can manage the instantaneous turn-off and turn-on of the output. This time is generally known as dead-time, and you have to adjust it depending on the devices you have connected to the outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches...)

The CxP and CxNP bits in the PWM_CER register are configured, and the polarity (main output OCx or complementary output OCxN) can be independently selected for each output.

Setting the CxE and CxNE bits at the same time will insert the dead zone, and if there is a brake circuit, the MOE bit will also be set. There is one 8-bit dead-time generator for each channel.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The figure below shows the output of OCx and OCxN when inserting the dead band. (Assuming CxP = 0, CxNP = 0, MOE = 1, CxE = 1, and CxNE = 1)

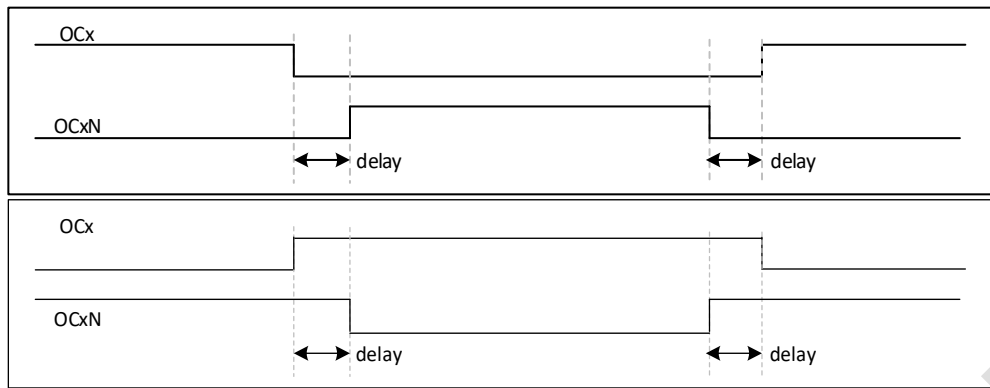


Figure 23-23 Complementary output with dead-time insertion

The dead time delay is the same for each channel and is programmed by the DTG bit in the PWM_BDTR register.

23.2.7. Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits. In any case, the OCx and OCxN outputs cannot be set both to active level at a given time.

The source for break (BRK) channel can be an external source connected to the BKIN pin or one of the following internal sources:

- the CPU LOCKUP output
- the PVD output
- a clock failure event generated by the CSS detector
- the output from a comparator
- SRAM byte parity error
- FLASH ECC error

When exiting from reset, the break circuit is disabled and the MOE bit is low. Setting the BKE bit in the PWM_BDTR register enables the brake function, and the polarity of the brake input signal can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time. When the BKE and BKP bits are written, a delay of 1 APB clock cycle is applied before the writing is effective. Consequently, it is necessary to wait 1APB clock period to correctly read back the bit after the write operation.

Because the MOE falling edge can be asynchronous, a resynchronization circuit is provided between the actual signal (acting on the output) and the synchronization control bit (in the PWM_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if you write MOE to 1 whereas it was low, you must insert a delay (dummy instruction) before reading it correctly. This is because the write acts on the asynchronous signal whereas the read reflects the synchronous signal.

When a break occurs (selected level on the break input):

- The MOE bit is cleared asynchronously, disabling the output. This feature remains valid even when the MCU's clock oscillator is turned off.
- If the BIE bit in the PWM_DIER register is set, an interrupt is generated when the brake status flag (BIF bit in the PWM_SR register) is '1'.

Note: The break inputs are active on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status flag BIF can not be cleared.

The brake may be generated by the BRK input, its effective polarity is programmable, and is turned on by the BKE bit in the PWM_BDTR register.

23.3. PWM interrupts

Table 23-1 PWM interrupts

Interrupt event	Event flag	Interrupt enable control bit
update	UIF	UIE
Output Compare 1	OC1IF	OC1IE
Output Compare 2	CC2IF	OC2IE
Output Compare 3	CC3IF	OC3IE
Output Compare 4	CC4IF	OC4IE
Brake	BIF	BIE

23.4. PWM Debug Mode

When the microcontroller enters debug mode (Cortex®-M0+ core stopped), depending on the DBG_PWM_STOP setting in the DBGMCU module, the PWM can either continue normal operation or stop.

23.5. PWM registers

23.5.1. PWM control register 1 (PWM_CR1)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	Res.	URS	UDIS	CEN
						RW	RW	RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	2'b0	Clock division factor. These 2 bits define the division ratio between the timer clock (CK_INT) frequency and the dead-time duration and the sampling clock (t_{DTs}) used by the dead-time generator and digital filter (ETR). 00: t_{DTs} = t_{CK_INT} 01: t_{DTs} = 2 x t_{CK_INT} 10: t_{DTs} = 4 x t_{CK_INT} 11: reserved, do not use this configuration.
7	ARPE	RW	0	Automatic reload preload allowed bit 0: PWM_ARR register is not buffered 1: PWM_ARR register is loaded into buffer
6:5	CMS[1:0]	RW	00	Counting mode. 00: Edge-aligned mode. The counter counts up or down based on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts alternately up and down. The output compare interrupt flag configured as an output channel is only set when the counter counts down. 10: Center-aligned mode 2. The counter counts alternately up and down. The output compare interrupt flag for a channel configured as output is set only when the counter counts up. 11: Center-aligned mode 3. The counter counts alternately up and down. The output compare interrupt flag for a channel configured as output is set both when the counter counts up and counts down. Note: When the counter is enabled (CEN=1), switching from edge-aligned mode to center-aligned mode is not allowed.
4	DIR	RW	0	Counting direction. 0: Counter counts up 1: Counter counts down Note: This bit is read-only when the counter is configured in center-aligned mode.
3:1	Reserved	-	-	Reserved
2	URS	RW	0	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: - Counter overflow/underflow - Setting the UG bit 1: If update interrupt or DMA request is enabled, only counter overflow/underflow generates an update interrupt or DMA request
1	UDIS	RW	0	Update disable Software enables/disables UEV event generation through this bit.

				0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow – Setting the UG bit Shadow registers are loaded with preload values. 1: UEV disabled. No update event is generated, and the shadow registers (ARR, PSC, CCRx) retain their values. If the UG bit is set, the counter and prescaler are reinitialized.
0	CEN	RW	0	Counter enabled. 0: Counter disabled 1: Enable counter Note: External clock, gated mode, and encoder mode can only be used after software sets the CEN bit. Trigger mode can automatically set the CEN bit to 1 via hardware.

23.5.2. PWM control register 2 (PWM_CR2)

Offset address: 0x04

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Re s.	Re s.	Re s.	Re s.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL[2:0]			Res.	BKCMP2P	BKCMP1P	BKINP	Re s.	Re s.	Re s.	Re s.	Res.	Res.	BKCMP2E	BKCMP1E	BKINE
RW	RW	RW		RW	RW	RW							RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:13	ETRSEL[2:0]	RW	3'b0	External trigger source selection This bit segment is used to select an ETR input source 000: PWM_ETR_LS (low speed) 001: COMP1_OUT 010: COMP2_OUT 011: Reserved (The above ETR sources can enable filtering) 100: SYS_CLK 101: PWM_ETR_HS (high speed) (When configured as 100/101, the trigger frequency can exceed PWM PCLK, so the frequency division and filtering functions cannot be enabled, and the glitch-free input needs to be ensured) Note: Once the LOCK level (LOCK bit in the PWM_BDTR register) is set to 1, these bits cannot be modified.
12	Reserved	-	-	Reserved
11	BKCMP2P	RW	0	Brake input polarity control from comparator 2. This bit selects the brake input polarity from comparator 2 and must be configured together with the BKP polarity bit. 0: Comparator 2 brake input polarity not inverted (active low when BKP=0, active high when BKP=1) 1: Comparator 2 brake input polarity inverted (active high when BKP=0, active low when BKP=1) Note: Once the LOCK level (LOCK bit in the PWM_BDTR register) is set to 1, this bit cannot be modified
10	BKCMP1P	RW	0	Brake input polarity control from comparator 1. This bit selects the brake input polarity from comparator 1 and must be configured together with the BKP polarity bit. 0: Comparator 1 brake input polarity not inverted (active low when BKP=0, active high when BKP=1) 1: Comparator 1 brake input polarity inverted (active high when BKP=0, active low when BKP=1) Note: Once the LOCK level (LOCK bit in the PWM_BDTR register) is set to 1, this bit cannot be modified
9	BKINP	RW	0	BKIN input polarity. This bit selects the alternate function of the BKIN input polarity and must be configured at the same time as the BKP polarity bit 0: BKIN input polarity does not flip (low active when BKP = 0, high active when BKP = 1)

				1: BKIN input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the PWM_BDTR register) is set to 1, this bit cannot be modified
8:3	Reserved	-	-	Reserved
2	BKCMP2E	RW	0	Comparator 2 brake input enable. Comparator 2 brake input is ORed with other brake sources as the brake input. 0: Comparator 2 brake input off 1: Comparator 2 brake input enabled Note: Once the LOCK level (LOCK bit in the PWM_BDTR register) is set to 1, this bit cannot be modified
1	BKCMP1E	RW	0	Comparator 1 brake input enable. Comparator 1 brake input is ORed with other brake sources as the brake input. 0: Comparator 1 brake input disabled 1: Comparator 1 brake input enabled Note: Once the LOCK level (LOCK bit in the PWM_BDTR register) is set to 1, this bit cannot be modified
0	BKINE	RW	1	BKIN input enabled. This bit is used to enable the standby function of BKIN when the brake is input. The BKIN input is OR logic with other brake sources as a brake input. 0: BKIN input off 1: BKIN input on Note: Once the LOCK level (LOCK bit in the PWM_BDTR register) is set to 1, this bit cannot be modified.

23.5.3. PWM slave mode control register (PWM_SMCR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE.	ETPS[1:0]			ETF[3:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW								

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	ETP	RW	0	External trigger polarity This bit selects whether ETR or the inverted ETR is used as the trigger operation. 0: ETR is not inverted, active high or rising edge 1: ETR inverted, active low or falling edge
14	ECE	RW	0	External clock enable bit This bit enables external clock mode 0: Disable external clock mode; 1: Enable external clock mode. The counter is driven by any active edge on the ETRF signal.
13:12	ETPS[1:0]	RW	2'b0	External trigger prescaler The frequency of the external trigger signal ETRP must be at most 1/4 of the counting clock frequency. When a faster external clock is input, the frequency of the ETRP can be reduced using prescalation. 00: Prescaler OFF 01: ETR frequency divided by 2 10: ETR frequency divided by 4 11: ETR frequency divided by 8 Note: Prescaling is not supported when PWR_CR2.ETRSEL selects 100/101.
11:8	ETF[3:0]	RW	4'b0	External Trigger Filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. In practice, the digital filter is an event counter that records N events before considering a valid output edge. 0000: no filter, asynchronous 0001: f _{SAMPLING} = f _{CK_INT} , N = 2 0010: f _{SAMPLING} = f _{CK_INT} , N = 4 0011: f _{SAMPLING} = f _{CK_INT} , N = 8

				0100: $f_{\text{SAMPLING}} = f_{\text{DTS}}/2$, $N = 6$ 0101: $f_{\text{SAMPLING}} = f_{\text{DTS}}/2$, $N = 8$ 0110: Sampling frequency $f_{\text{SAMPLING}} = f_{\text{DTS}}/4$, $N=6$ 0111: Sampling frequency $f_{\text{SAMPLING}} = f_{\text{DTS}}/4$, $N=8$ 1000: Sampling frequency $f_{\text{SAMPLING}} = f_{\text{DTS}}/8$, $N=6$ 1001: Sampling frequency $f_{\text{SAMPLING}} = f_{\text{DTS}}/8$, $N=8$ 1010: Sampling frequency $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N=5$ 1011: Sampling frequency $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N=6$ 1100: $f_{\text{SAMPLING}} = f_{\text{DTS}}/16$, $N = 8$ 1101: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 5$ 1110: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 6$ 1111: $f_{\text{SAMPLING}} = f_{\text{DTS}}/32$, $N = 8$ Note: Filtering is not supported when ETRSEL selects 100/101.
7:0	Reserved	-	-	Reserved

23.5.4. PWM DMA/Interrupt Enable Register (PWM_DIER)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res.	Res.	Res.	Res.	Res.	Res	Res	Res	Res.	Res.	Res.	Res.	Res
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	OC4D	OC3D	OC2D	OC1D	UD	BIE	Res	Res	OC4I	OC3I	OC2I	OC1I	UIE
.	.	.	E	E	E	E	E	.	.	.	E	E	E	E	.
.	.	.	RW	RW	RW	RW	RW	RW	.	.	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	OC4DE	RW	0	DMA request enable for output compare 4 0: Disable the output of DMA request for comparison 4 1: DMA request allowed to output comparison 4
11	OC3DE	RW	0	DMA request enable for output compare 3 0: Disable output of DMA request for comparison 3 1: DMA request allowed to output comparison 3
10	OC2DE	RW	0	DMA request enable for output compare 2 0: Disable the output of DMA request for comparison 2 1: DMA request allowed to output comparison 2
9	OC1DE	RW	0	DMA request enable for output compare 1 0: Disable the output of DMA request for comparison 1 1: DMA request allowed to output comparison 1
8	UDE	RW	0	Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled
7	BIE	RW	0	Brake interrupt 0: Break interrupt disabled 1: Break interrupt enabled
6:5	Reserved	-	-	Reserved
4	OC4IE	RW	0	Compare output 4 interrupt enable 0: Disable comparison output 4 interrupt 1: Allow comparison output 4 interrupt
3	OC3IE	RW	0	Compare output 3 interrupt enable 0: Disable comparison output 3 interrupt 1: Allow comparison output 3 interrupt
2	OC2IE	RW	0	Compare output 2 interrupt enable 0: Disable comparison output 2 interrupt 1: Allow comparison output 2 interrupt
1	OC1IE	RW	0	Compare output 1 interrupt enable 0: Disable comparison output 1 interrupt 1: Allow comparison output 1 interrupt
0	UIE	RW	0	Update interrupt enable 0: Update interrupt disabled. 1: Enable update interrupt

23.5.5. PWM status register (PWM_SR)

Offset address: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Re s.	Res.	Re s.	Re s.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR OK	PSC OK	CNT OK	CCR4 OK	CCR3 OK	CCR2 OK	CCR1 OK	Re s.	BIF	Re s.	Re s.	OC4I F	OC3I F	OC2I F	OC1F	UIF
RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0			RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	ARROK	RC_W0	0	PWM_ARR register update complete flag, valid only when configured to count asynchronously (ETRSEL configured to 100/101). ARROK is set by hardware to notify the application that the APB bus write to PWM_ARR has successfully completed when configured to count asynchronously. This register is cleared by the software.
14	PSCOK	RC_W0	0	PWM_PSC register update complete flag, valid only when configured to count asynchronously (ETRSEL configured to 100/101). PSCOK is set by hardware to notify the application that the APB bus write to PWM_PSC has successfully completed when configured to count asynchronously. This register is cleared by the software.
13	CNTOK	RC_W0	0	PWM_CNT register update complete flag, valid only when configured to count asynchronously (ETRSEL configured to 100/101). The CNTOK is set by the hardware to inform the application that the APB bus write to PWM_CNT has successfully completed when configured to count asynchronously. This register is cleared by the software.
12	CCR4OK	RC_W0	0	PWM_CCR4 register update complete flag, valid only when configured to count asynchronously (ETRSEL configured to 100/101). CCR4OK is set by hardware to notify the application that the APB bus write to PWM_CCR4 has successfully completed when configured to count asynchronously. This register is cleared by the software.
11	CCR3OK	RC_W0	0	PWM_CCR3 register update complete flag, valid only when configured to count asynchronously (ETRSEL configured to 100/101). CCR3OK is set by hardware to notify the application that the APB bus write to PWM_CCR3 has successfully completed when configured to count asynchronously. This register is cleared by the software.
10	CCR2OK	RC_W0	0	PWM_CCR2 register update complete flag, valid only when configured to count asynchronously (ETRSEL configured to 100/101). CCR2OK is set by hardware to notify the application that the APB bus write to PWM_CCR2 has successfully completed when configured to count asynchronously. This register is cleared by the software.
9	CCR1OK	RC_W0	0	PWM_CCR1 register update complete flag, valid only when configured to count asynchronously (ETRSEL configured to 100/101). CCR1OK is set by hardware to notify the application that the APB bus write to PWM_CCR1 has successfully completed when configured to count asynchronously. This register is cleared by the software.
8	Reserved	-	-	Reserved
7	BIF	RC_W0	0	Brake interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input.

6:5	Reserved	-	-	Reserved
4	OC4IF	RC_W0	0	Output Compare 4 Interrupt Flag Refer to OC1IF description
3	OC3IF	RC_W0	0	Output Compare 3 Interrupt Flag Refer to OC1IF description
2	OC2IF	RC_W0	0	Output Compare 2 Interrupt Flag Refer to OC1IF description
1	OC1IF	RC_W0	0	Output Compare 1 Interrupt Flag This flag is set by hardware when the counter matches the compare value. 0: No match; 1: The value of PWM_CNT matches the value of PWM_CCR1.
0	UIF	RC_W0	0	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update event generation (counter overflow or under-flow or set UG). This bit is set to 1 by hardware.

23.5.6. PWM Event Generation Register (PWM_EGR)

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UG
															W

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	Reserved
0	UG	W	0	Update generation This bit is set to '1' by the software, and '0' is automatically cleared by the hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the counter of the prescaler is also cleared '0' (but the prescaler coefficient remains unchanged).

23.5.7. PWM output comparison mode register 1 (PWM_CMR)

Offset address: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	OC4PE	OC3PE	OC2PE	OC1PE	OC4M [1:0]	OC3M [1:0]	OC2M [1:0]	OC1M [1:0]				
				RW	RW	RW	RW	RW	RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	OC4PE	RW	0	Output Compare 4 preload enable
10	OC3PE	RW	0	Output Compare 3 preload enable
9	OC2PE	RW	0	Output Compare 2 preload enable
8	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on PWM_CCR1 disabled. PWM_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Enable the preload function of the PWM_CCR1 register. Read and write operations only affect the preload register. The preload value of PWM_CCR1 is loaded into the current register upon each update event. Note 1: Once the LOCK level is set to 3 (the LOCK bit in the PWM_BDTR register), this bit cannot be modified.
7:6	OC4M [1:0]	RW	0	OC4 PWM Patterns. Refer to OC1M description.
5:4	OC3M [1:0]	RW	0	OC3 PWM Patterns. Refer to OC1M description.

3:2	OC2M [1:0]	RW	0	OC2 PWM Patterns. Refer to OC1M description.
1:0	OC1M [1:0]	RW	0	OC1 PWM mode. 10: PWM mode 1---During up-counting, channel 1 is active when PWM_CNT<PWM_CCR1, otherwise inactive; during down-counting, channel 1 is inactive when PWM_CNT>PWM_CCR1, otherwise active. 11: PWM mode 2---During up-counting, channel 1 is inactive when PWM_CNT<PWM_CCR1, otherwise active; during down-counting, channel 1 is active when PWM_CNT>PWM_CCR1, otherwise inactive. Other values: Reserved. Note: The active level is configured by the PWM_CER register, and the invalid level is inverted.

23.5.8. PWM output comparison enable register (PWM_CER)

Offset address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	C4P	C4E	Res.	Res.	C3P	C3E	C2NP	C2NE	C2P	C2E	C1NP	C1NE	C1P	C1E
		RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13	C4P	RW	0	OC4 output polarity. Refer to C1P description.
12	C4E	RW	0	OC4 output enabled. Refer to C1E description.
11:10	Reserved	-	-	Reserved
9	C3P	RW	0	OC3 output polarity. Refer to C1P description.
8	C3E	RW	0	OC3 output enabled. Refer to C1E description.
7	C2NP	RW	0	OC2 complementary output polarity. Refer to C1NP description.
6	C2NE	RW	0	OC2 complementary output enabled. Refer to C1NE description.
5	C2P	RW	0	OC2 output polarity. Refer to C1P description.
4	C2E	RW	0	OC2 output enabled. Refer to C1E description.
3	C1NP	-	0	OC1 complementary output polarity 0: OC1N active high. 1: OC1N active low. Note: Once the LOCK level (LOCK bit in PWM_BDTR register) is set to 3 or 2, this bit cannot be modified.
2	C1NE	RW	0	OC1 complementary output enable 0: Off - OC1N output disabled, output level is high impedance. 1: Enabled - OC1N signal is output to the corresponding pin, with its level being the complement of OC1.
1	C1P	RW	0	OC1 output polarity 0: OC1 active high. 1: OC1 active low. Note: Once the LOCK level (LOCK bit in PWM_BDTR register) is set to 3 or 2, this bit cannot be modified.
0	C1E	RW	0	OC1 output enabled 0: Disabled - OC1 output is disabled, output level is high impedance. 1: Enabled - OC1 signal is output to the corresponding pin, with its level depending on the value of C1P.

MOE	CxE	CxNE	OCx output status	OCxN output state
1	0	0	Output disabled (not timer-driven), OCx=0, OCx_EN=0	Output disabled (not timer-driven), OCxN=0, OCxN_EN=0
	0	1	Output disabled (not driven by timer), OCx=0, OCx_EN=0	OCxREF + polarity OCxN=OCxREF XOR CCxNP, OCxN_EN=1
	1	0	OCxREF + polarity OCx=OCREF xor CCxP, OCx_EN=1	Output disabled (not timer-driven), OCxN=0, OCxN_EN=0
	1	1	OCREF + polarity + dead time OCx_EN = 1	Complement of OCREF (OCREF not) + polarity + dead time OCxN_EN = 1
0	X	X	Output disabled (not driven by the timer anymore).	

23.5.9. PWM counter register (PWM_CNT)

Offset address: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	0	Counter value

23.5.10. PWM Prescaler (PWM_PSC)

Offset address: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC	RW	0	Prescaler value The clock frequency (CK_CNT) of the counter is equal to $f_{CK\_PSC}/(PSC[15:0] + 1)$. PSC contains the value to be loaded into the active prescaler register each time an update event occurs.

23.5.11. PWM Auto-reload Register (PWM_ARR)

Offset address: 0x2c

Reset value: 0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	ARR[15:0]	RW	16'hFFFF	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

23.5.12. PWM Comparison Register 1 (PWM_CCR1)

Offset Address: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1 [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR1 [15:0]	RW	16'b0	Compare 1 value. CCR1 contains the value loaded into the current Compare 1 register (preload value). If the preload feature is not selected in the PWM_CMCR register (OC1PE bit), this value takes effect immediately. Otherwise, the value only takes effect when an update event occurs.

				The actual compare register contains the value compared with the PWM_CNT counter, and outputs the PWM signal on the OC1 port.

23.5.13. PWM comparison register 2 (PWM_CCR2)

Offset Address: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2 [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR2 [15:0]	RW	0	Compare 2 value. CCR2 contains the value loaded into the current Compare 2 register (preload value). If the preload feature is not selected in the PWM_CMR register (OC2PE bit), this value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual compare register contains the value compared with the PWM_CNT counter, and outputs the PWM signal on the OC2 port.

23.5.14. PWM comparison register 3 (PWM_CCR3)

Offset Address: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3 [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR3 [15:0]	RW	0	Compare the value of 3. CCR3 contains the value loaded into the current Compare 3 register (preload value). If the preload feature is not selected in the PWM_CMR register (OC3PE bit), the value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual compare register contains the value compared with the PWM_CNT counter and outputs the PWM signal on the OC3 port.

23.5.15. PWM comparison register 4 (PWM_CCR4)

Offset address: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4 [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR4 [15:0]	RW	0	Compare values of 4 CCR4 contains the value loaded into the current Compare 4 register (preload value). If the preload feature is not selected in the PWM_CMR register (OC4PE bit), the value takes effect immediately. Otherwise, the value only takes effect when an update event occurs. The actual compare register contains the value compared with the PWM_CNT counter and outputs the PWM signal on the OC4 port.

23.5.16. PWM Brake and Dead Time Register (PWM_BDTR)

Offset Address: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	Res.	Res.	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	MOE	RW	0	Brake main output enable This bit is cleared asynchronously by hardware as soon as one of the break inputs is active. 0: Disable OCx and OCxN outputs; 1: Turn on OCx and OCxN outputs if the corresponding enable bits (CxEN, CxNE bits of PWM_CER register) are set.
14	AOE	RW	0	Automatic output enable 0: MOE can be set only by software; 1: MOE can be set by software or automatically at the next update event (if none of the break inputs is active). Note: Once the LOCK level (LOCK bit in the PWM_BDTR register) is set to 1, this bit cannot be modified.
13	BKP	RW	0	Break polarity 0: Break input BRK is active low; 1: Break input BRK is active high
12	BKE	RW	0	Break enable 0: Break function disabled 1: Break function enabled
11:10	Reserved	-	-	Reserved
9:8	LOCK[1:0]	RW	00	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: Lock level 1, cannot write to DTG/BKE/BKP/AOE bit of PWM_BDTR register 10: Lock level 2, cannot write bits in lock level 1, and cannot write CxP/CNxP polarity bits 11: Lock level 3, cannot write bits in lock level 2 and cannot write CxM control bits Note: After system reset, the LOCK bit can only be written once, and once written to the PWM_BDTR register, its contents freeze until reset.
7:0	DTG[7:0]	RW	0000 0000	Dead time setting. This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. $DTG[7:5]=0xx \Rightarrow DT=DTG[7:0] \times T_{dts}$, where $T_{dts} = T_{DTS}$; $DTG[7:5]=10x \Rightarrow DT=(64+DTG[5:0]) \times T_{dts}$, where $T_{dts} = 2 \times T_{DTS}$; $DTG[7:5]=110 \Rightarrow DT=(32+DTG[4:0]) \times T_{dts}$, where $T_{dts} = 8 \times T_{DTS}$; $DTG[7:5]=111 \Rightarrow DT=(32+DTG[4:0]) \times T_{dts}$, where $T_{dts} = 16 \times T_{DTS}$; Example if $T_{DTS}=125ns$ (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps Note: Once the LOCK level (LOCK bit in PWM_BDTR register) is set to 1, 2, or 3, these bits cannot be modified.

23.5.17. PWM DMA Control Register (PWM_DCR)

Offset Address: 0x48

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res.	Res	DBL[4:0]					Res	Res	Res	DBA[4:0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12:8	DBL	RW	0	DMA burst length These bits define the transfer length of the DMA in continuous mode (when reading or writing to the PWM_DMAR register, the timer makes a continuous transfer), i.e. defines the number of transfers, which can be half words (double bytes) or bytes: 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers 10001: 18 transfers Take this transfer as an example: DBL=7 bytes, DBA=PWM_CR1: --If DBL=7 bytes and DBA=PWM_CR1 represents the address of the data to be transferred, then the transfer address is given by: (Address of PWM_CR1) + DBA + (DMA index), where DMA index=DBL, Where (address of PWM_CR1) + DBA plus 7 gives the address where data will be written or read out, so that the transfer of data will occur in the 7 registers starting from address (address of PWM_CR1) + DBA. Depending on the setting of the DMA data length, the following can occur: --If the data is set to half-word (16-bit), then the data will be transferred to all 7 registers. --If the data is set to byte, the data will still be transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte, and so on. Therefore, for the timer, the user must specify the data width to be transmitted by the DMA.
7:5	Reserved	-	-	Reserved
4:0	DBA	RW	0	DMA base address These bits define the base address of the DMA in continuous mode (when reading or writing to the PWM_DMAR register), and the DBA is defined as the offset from the address where the PWM_CR1 register is located: 00000: PWM_CR1 00001: PWM_CR2 00010: PWM_SMCR 00011: PWM_DIER 00100: PWM_SR

23.5.18. DMA address for PWM continuous mode (PWM_DMAR)

Offset address: 0x4C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB	RW	0	DMA continuous transfer register. A read or write to the PWM_DMAR register results in an access operation to the register with the following address: PWM_CR1 address + (DBA + DMA index) x4 where: "PWM_CR1 address" is the address of control register 1 (PWM_CR1); "DBA" is the base address defined in the PWM_DCR register; The "DMA index" is an offset automatically controlled by the DMA, which depends on the DBL defined in the PWM_DCR register.

Note: When using the DMA continuous transfer function, the value of the CNDTR register in the corresponding DMA channel must correspond to the DBL value in the PWM_DCR register; otherwise, it will not function properly.

Puya Confidential

24. Infrared Interface (IRTIM)

An infrared interface (IRTIM) for remote control is integrated into the chip. It can realize the function of remote control together with an infrared LED.

To generate infrared remote control signals, the infrared interface must be enabled, and TIM16 channel 1 (TIM16_OC1) and TIM17 channel 1 (TIM17_OC1) must be correctly configured to produce the proper waveform.

The infrared receiver can be easily implemented through a basic input capture mode.

All standard infrared pulse modulation modes can be achieved by programming the output compare channels of the two timers.

TIM 17 is used to generate a high frequency carrier signal, while TIM 16 may generate a modulation envelope.

The infrared function outputs to the IR_OUT pin, and this function is activated by enabling the relevant multiplexing function bits in the GPIO_AFRx register.

LEDs require high sink current drive capability (for PB2 ~ PB9), which can be activated by the PB_EHS register in the SYSCFG_LED_CFG register, providing sufficient support for directly controlling the high sink current of infrared LEDs.

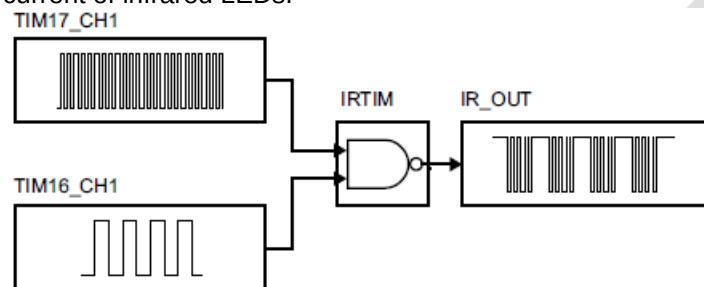


Figure 24-1 IRTIM Implementation

25. Low-Power Timer (LPTIM)

25.1. Introduction to LPTIM

This product includes two LPTIMs, where LPTIM1 supports 32-bit counting and LPTIM2 supports 16-bit counting. The ability of LPTIM to wake the system from low-power modes makes it suitable for implementing low-power applications except for standby mode. The LPTIM can operate without an internal clock source and can be used as a "pulse counter."

LPTIM introduces a flexible clocking scheme that delivers the required functionality and performance while minimizing power consumption.

25.2. LPTIM main features

- 16-bit/32-bit up counter
- 3-bit prescaler with 8 possible dividing factors (1, 2, 4, 8, 16, 32, 64, 128)
- Selectable clock
 - Internal clock sources: LSE, LSI, or APB clock
- 16/32-bit auto-reload register
- 16/32-bit compare register
- Single/continuous mode
- Optional software/hardware input trigger
- Configurable digital filter for trigger signals
- Configurable output:
 - Pulse
 - PWM
- Configurable IO trigger polarity
- Encoder Mode

- The digital filter protects the LPTIM internal trigger input. The digital filter sensitivity is controlled by the TRGFLT bit.

Note: Digital filter sensitivity is controlled by banks. It is not possible to configure each digital filter sensitivity individually within the same group.

The filter sensitivity acts on the number of consecutive equal samples detected on an input of the LPTIM to treat the signal level change as a valid transition. The figure below shows an example of the glitch filter behavior in the case of programming 2 consecutive samples.

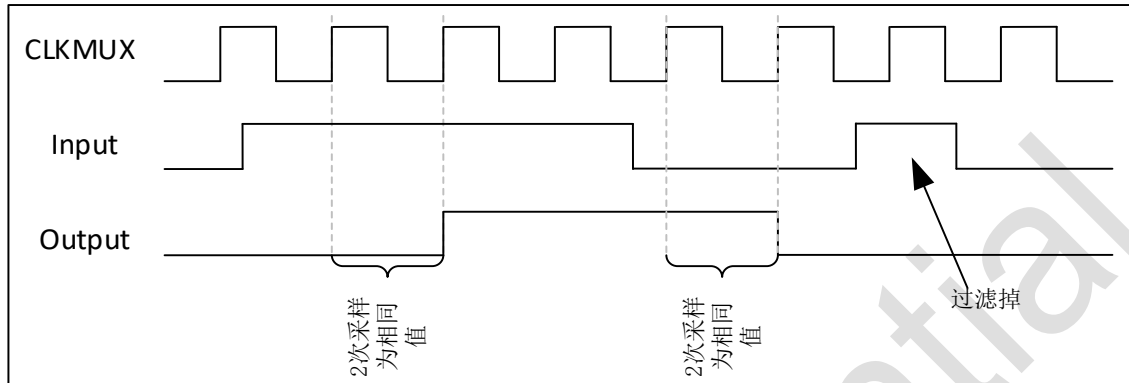


Figure 25-2 Example of glitch filter behavior (2 consecutive samples)

Note: If no internal clock signal is provided, the digital filter must be deactivated by setting the TRGFLT bit to "0". In this case, an external analog filter (IO module implementation) can be used to protect the LPTIM external input from glitch interference.

25.3.5. Prescaler

The LPTIM counter is driven by a configurable power-of-two prescaler. The prescaler division ratio is controlled by the PRESC[2:0].

The table below lists all the possible division ratios:

Table 25-2 Division Ratio of Prescaler

Programming	Dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

25.3.6. Trigger selection

The LPTIM counter can be started either by software or upon detection of an active edge of one of the 8 (optional maximum number of inputs) trigger inputs.

TRIGEN[1:0] is used to determine the LPTIM trigger source:

- When TRIGEN [1: 0] equals "00", the LPTIM counter starts once one of the CNTSTRT or SNGSTRT bits is set by the software. Additional values of TRIGEN [1: 0] can be used to configure the valid edge used by the trigger input, supporting rising and falling edges. Once a valid edge is detected, the LPTIM counter starts.
- When TRIGEN [1: 0] is not '00', TRIGSEL [2: 0] is used to select which of the 8 trigger inputs starts the counter.

External triggers are treated as asynchronous signals for LPTIM. Therefore, after the trigger detection, due to the synchronization, a delay of two counter clock cycles is required before the timer starts running.

If a new trigger event occurs when the timer has already started, it will be ignored.

The timer must be enabled before setting the SNGSTRT/CNTSTRT bit. When the timer is disabled, any writes to these bits will be discarded by the hardware.

Note: When the counter is started via software (TRIGEN [1: 0] = 00), there is a delay of 3 count clock cycles between the LPTIM_CR register update (setting one of the SNGSTRT or CNTSTRT bits) and the effective start of the counter.

25.3.7. Mode of operation

LPTIM has two operating modes as follows:

- Continuous counting mode: The timer runs freely, starting from the trigger event and stopping only when the timer is disabled.
- Single count mode: The timer starts with a trigger event and stops when the ARR value is reached.

25.3.7.1. Single-shot mode

To enable single count, the LPTIM_CR.SNGSTRT register bit must be set to 1.

Setting SNGSTRT will start the counter for a single count.

A new trigger event will re-start the timer. Any trigger events are ignored after the counter is started and until the ARR is reached.

If external trigger is selected, an external trigger event is generated after the SNGSTRT bit is set and the counter register is stopped (containing zero values), and LPTIM will start the counter for a new single counting cycle, as shown in the figure below.

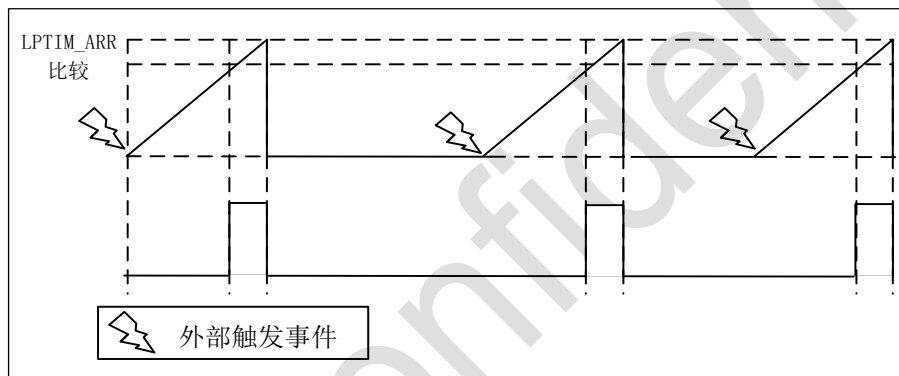


Figure 25-3 LPTIM output, single count mode

Single-shot mode is activated when the LPTIM_CFGR.WAVE bit is set to 1. In this case, the counter is only started once after the first trigger, and any subsequent trigger events are ignored. As shown in the following image:

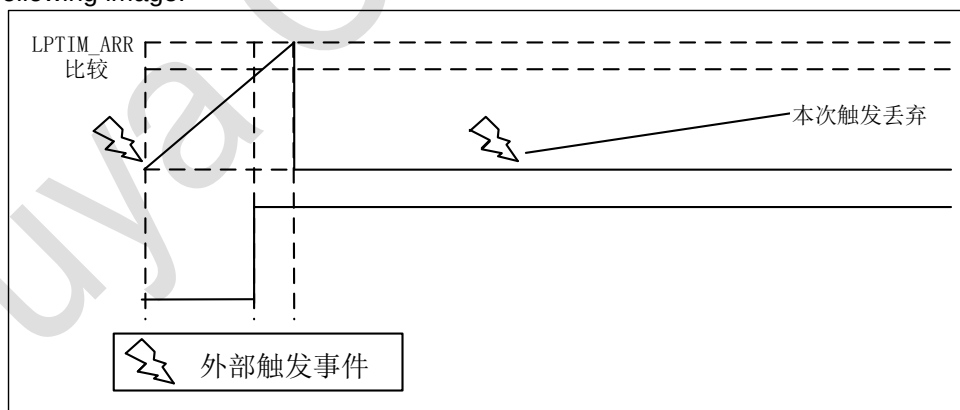


Figure 25-4 LPTIM output, single count mode (WAVE = 1)

25.3.7.2. Continuous mode

To enable the continuous counting, the LPTIM_CR.CNTSTRT bit must be set.

If external trigger is selected, external trigger events that arrive after CNTSTRT is set will start the counter for continuous counting. Any subsequent externally triggered events will be discarded as shown in the image below.

When the software starts (TRIGEN[1:0]='00'), setting LPTIM_CR.CNTSTRT will initiate the counter for continuous counting.

The continuous counting waveform is as follows:

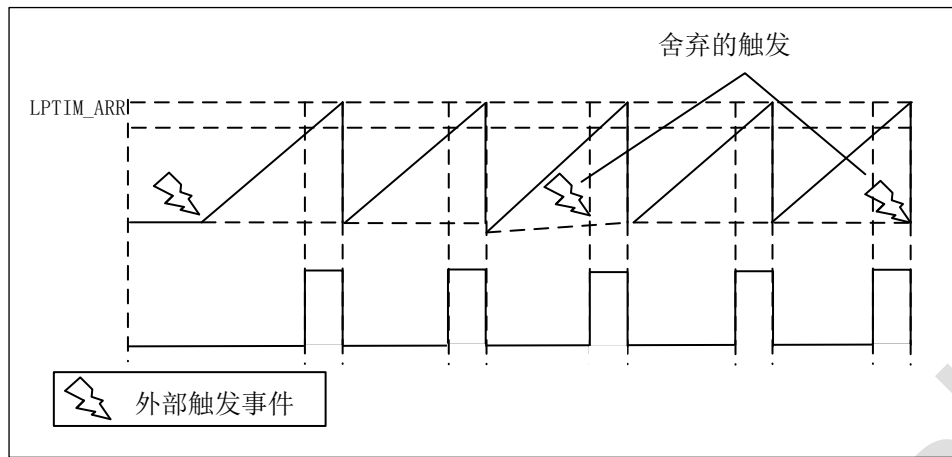


Figure 25-5 LPTIM Output, Continuous Counting Mode

The LPTIM_CR.SNGSTRT and LPTIM_CR.CNTSTRT bits can only be set when the timer is enabled (LPTIM_CR.ENABLE is '1').

It is possible to switch "on the fly" from one-shot mode to continuous mode:

- If the Continuous mode was previously selected, setting SNGSTRT will switch the LPTIM to the One-shot mode. The counter (if active) will stop upon reaching ARR.
- If one-shot mode was previously selected, setting LPTIM_CR.CNTSTRT switches the LPTIM to continuous mode. The counter (if active) will restart upon reaching ARR.

25.3.8. Waveform generation

Two registers, LPTIM_ARR (auto-reload register) and LPTIM_CMP (compare register), are used to generate several different waveforms on the LPTIM output.

The timer can generate the following waveforms:

- PWM mode: The LPTIM output is set when the counter value in LPTIM_CNT exceeds the comparison value in LPTIM_CMP. Once a match occurs between the LPTIM_ARR and LPTIM_CNT registers, the LPTIM output is reset.
- Single pulse mode: The output waveform is similar to the PWM mode of the first pulse, and then the output is permanently reset.
- Once set mode: The output waveform is similar to monopulse mode, except that the output will remain at the last signal level (depending on the polarity of the output configuration).

The above mode requires that the LPTIM_ARR register value be strictly greater than the LPTIM_CMP register value.

The LPTIM output waveform can be configured via the WAVE bit as follows:

- Resetting the WAVE bit to "0" forces LPTIM to generate either a PWM waveform or a monopulse waveform, depending on which bit is set: CNTSTRT or SNGSTRT.
- Setting the WAVE bit to "1" forces LPTIM to generate a set mode waveform once.

The WAVPOL bit controls the LPTIM output polarity. Changes take effect immediately, so the default output value will change right after reconfiguring the polarity, even before enabling the timer. Signals with frequencies up to 1/2 of the LPTIM clock frequency can be generated. The figure below shows three possible waveforms that can be generated on the LPTIM output. Furthermore, it shows the effect of changing polarity using WAVPOL bits.

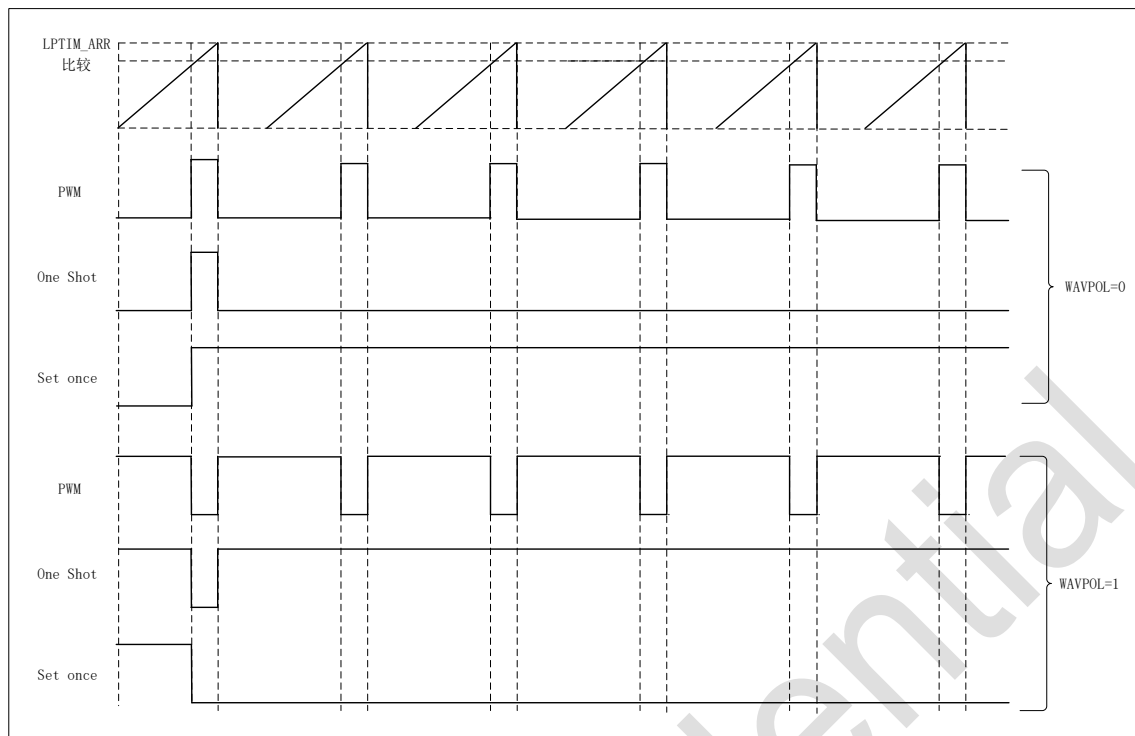


Figure 25-6 LPTIM waveform generation

25.3.9. Register update

The LPTIM_ARR register is updated immediately after the APB bus write operation, and if the timer has been started, it is updated in synchronization with the next LPTIM update event.

The PRELOAD bit controls how the LPTIM_ARR register is updated:

- When the PRELOAD bit is reset to '0', the LPTIM_ARR and the LPTIM_CMP registers are immediately updated after any write access.
- When the PRELOAD bit is set to "1": If the timer has been started, the LPTIM_ARR register will be updated at the end of the current cycle.

The LPTIM APB interface and the LPTIM kernel logic use different clocks, so there is a certain delay when the APB writes, and the written values are applied to the counter comparator. Within this latency period, any additional write into these registers must be avoided.

The ARROK flag and CMPOK in the LPTIM_ISR register indicate whether the write operation to the LPTIM_ARR and LPTIM_CMP registers is completed.

After writing to the LPTIM_ARR and LPTIM_CMP registers, a new write operation can be performed to the same register only after the previous write operation is completed. Any successive writes before the ARROK flag bit or the CMPOK flag position bit will result in unpredictable results.

25.3.10. Counter mode

The following counting modes can be selected:

- The LPTIM is timed by an internal clock source

The LPTIM clock is provided by an internal clock source, and the LPTIM counter is configured to update after each internal clock pulse.

25.3.11. Timer enable

The ENABLE bit located in the LPTIM_CR register is used to enable/disable the LPTIM kernel logic. After setting the ENABLE bit, a delay of 3 **counter clocks** (the clock obtained by dividing the core clock through PRESC) is required to truly enable LPTIM.

The LPTIM_CFGFR and LPTIM_IER registers must be modified only when the LPTIM is disabled.

25.3.12. Timer reset

To reset the contents of the LPTIM_CNT register, two reset mechanisms are provided as described below.

Note: Asynchronous reset and synchronous reset should be used mutually exclusively.

25.3.12.1. Asynchronous reset

The asynchronous reset is controlled by the RSTARE bit of the LPTIM_CR register. When this bit is set to 1, any access to the LPTIM_CNT register resets its contents to zero. The asynchronous reset should be triggered within a time frame when the LPTIM kernel clock is not provided.

It should be noted that in order to reliably read the LPTIM_CNT register, two read accesses must be performed and the results are compared. If the results are consistent, it is considered that the read value is reliable.

It is important to note that:

- When asynchronous reset is enabled, the results of two reads of the LPTIM_CNT register are different (the first read resets LPTIM_CNT).
- When PCLK is selected by the LPTIM counting clock, two consecutive accesses cannot guarantee that the read value is reliable.

25.3.12.2. Synchronous reset

The synchronous reset is controlled by the LPTIM_CR.COUNTRST bit. After setting the COUNTRST bit-field to '1', the reset signal is propagated in the LPTIM kernel clock domain. So it is important to note that a few clock pulses of the LPTIM kernel logic will elapse before the reset is taken into account. This will make the LPTIM counter count few extra pluses between the time when the reset is trigger, and it becomes effective.

Since the COUNTRST bit is located in the APB clock domain and the LPTIM counter is located in the LPTIM kernel clock domain, a delay of 3 clock cycles of the kernel clock is needed to synchronize the reset signal issued by the APB clock domain when writing '1' to the COUNTRST bit.

25.3.13. Encoder Mode

This mode allows processing of signals from the quadrature encoder used to detect the angular position of the rotating element. The encoder interface mode simply acts as an external clock with directional selection. This means that the counter just counts continuously between 0 and ARR values (0 to ARR or ARR to 0, depending on the direction). Therefore, LPTIM_ARR must be configured before startup. A clock signal is generated from two external input signals Input1 and Input2 to clock the LPTIM counter. The phase between these two signals determines the counting direction.

The encoder mode is only available when the LPTIM is counted by the internal clock source. The signal frequency on the Input1 and Input2 inputs must not exceed 1/4 of the LPTIM internal clock. This is a mandatory requirement to ensure the proper operation of LPTIM.

The direction change is indicated by two down and up flags in the LPTIM_ISR register. Additionally, if the DOWNIE bit is enabled, interrupts can be generated for event changes in both directions.

To activate encoder mode, the ENC bit must be set to "1". The LPTIM must first be configured in Continuous mode.

When the encoder mode is active, the LPTIM counter is automatically modified according to the speed and direction of the incremental encoder. Therefore, its contents always represent the position of the encoder. The counting direction indicated by the increment and decrement flags corresponds to the rotation direction of the encoder rotor.

Different counting cases may occur depending on the edge sensitivity of the CKPOL [1: 0] bit configuration. The table below summarizes the possible combinations, assuming that Input1 and Input2 do not switch at the same time.

Table 25-3 Counting method

Effective edge	Reverse signal level (input1 for input2 input2 for input1)	input1 signal		input2 signal	
		rising edge	falling edge	rising edge	falling edge
rising edge	High	Decreasing	Not count	Up	Not count
	Low	Up	Not count	Decreasing	Not count
falling edge	High	Not count	Up	Not count	Decreasing
	Low	Not count	Decreasing	Not count	Up

The figure below shows the count sequence for an encoder mode configured with dual-edge sensitivity.

In this mode, the prescaler division ratio must be equal to its reset value of 1 (PRESC [2: 0] bit must be "000").

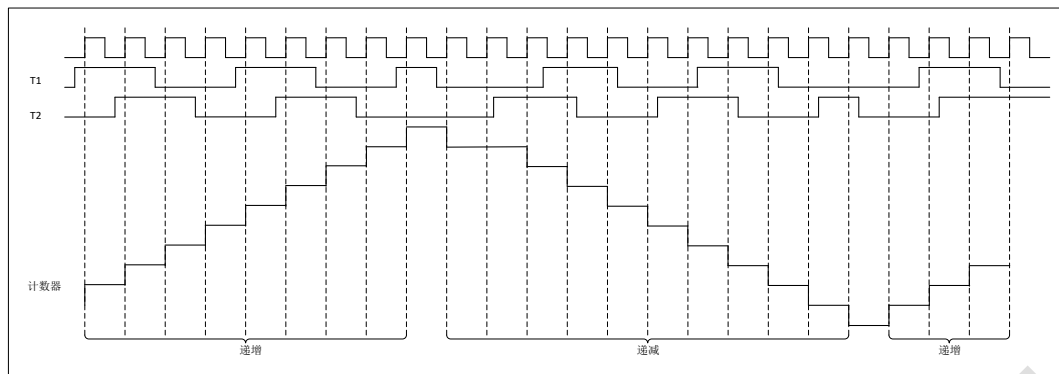


Figure 25-7 LPTIM counting pattern

25.3.14. Debug mode

When the chip enters debug mode, depending on the setting of the DBG_LPTIMx_STOP bit in the DBGMCU module, the LPTIM either continues normal operation or stops working.

25.3.15. Programming Guide

LPTIM initialization meets the following conditions:

1. After configuring LPTIM1EN/LPTIM2EN in the RCC module, immediately program LPTIM_CFGR.TRIGEN, then wait for 5 core clock cycles before configuring SNGSTRT or CNTSTRT in the LPTIM_CR register.

25.4. LPTIM low-power modes

Table 25-4 Impact of Low-Power Modes on LPTIM

Low-power modes	Description
Sleep	No effect. LPTIM interrupts (when enabled) will exit sleep mode.
Stop	No effect when LPTIM is clocked by LSE or LSI. LPTIM interrupts (when enabled) will exit all stop modes.
Standby	The LPTIM peripheral is powered down and must be reinitialized after exiting standby mode.

Note: When using LPTIM for wake-up, ensure the interval between two wake-ups is greater than 6 PCLK cycles; otherwise, unpredictable results may occur.

25.5. LPTIM interrupts

The following events generate an interrupt/wake-up event, if they are enabled through the LPTIM_IER register:

- Compare match (interrupt/wake-up)
- Auto-reload match (regardless of direction if in encoder mode) (interrupt/wake-up)
- Auto-reload register write complete (interrupt)
- Compare register write complete (interrupt)
- Direction change (encoder mode) (interrupt/wake-up)

Note: If the corresponding bit in the LPTIM_IER register (interrupt enable register) is set to 1 after the corresponding flag in the LPTIM_ISR register (status register) is set to 1, no interrupt will be generated.

Table 25-5 LPTIM interrupt events

Interrupt event	Description
Compare Match	An interrupt flag is generated when the contents of the counter register (LPTIM_CNT) match the contents of the comparison register (LPTIM_CMP).
Auto-reload match	Interrupt flag is raised when the content of the Counter register (LPTIM_CNT) matches the content of the Auto-reload register (LPTIM_ARR).
Automatic reload register write complete	Interrupt flag is raised when the write operation to the LPTIM_ARR register is complete.
Compare register write complete	An interrupt flag is generated when a write operation to the LPTIM_CMP register is completed.
Change of direction	Used in encoder mode. Two interrupt flags are embedded to indicate a change of direction: --UP flag indicates an increment counting direction change --DOWN flag denotes a decrement counting direction change

25.6. LPTIM registers

25.6.1. LPTIM interrupt and status register (LPTIM_ISR)

Address offset: 0x000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOW M	UP	ARR OK	CMP OK	Res.	ARR M	CMP M
									R	R	R	R		R	R

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6	DOWN	R	0	The counter changes from incremental count to decremental count In encoder mode, the DOWN bit is set by the hardware to inform the application that the counter has changed from an increment count to a decrement count. The DOWN flag may be cleared by writing 1 to the DOWNCF bit in the LPTIM_ICR register. Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
5	UP	R	0	The counter changed from decrement count to increment count In encoder mode, the UP bit is set by the hardware to inform the application that the counter has changed from a decrement count to an increment count. The UP flag may be cleared by writing 1 to the UPCF bit in the LPTIM_ICR register. Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
4	ARROK	R	0	Auto-reload register update OK ARROK is set by hardware to inform application that the APB bus write operation to the LPTIM_ARR register has been successfully completed. Writing 1 to LPTIM_ICR.ARROKCF clears the ARROK flag.
3	CMPOK	R	0	Compare Register Update OK CMPOK is set by hardware to notify the application that the APB bus write to the LPTIM_CMP register has successfully completed. Writing 1 to LPTIM_ICR.CMPOKCF clears the ARROK flag.
2	Reserved	-	-	Reserved
1	ARRM	R	0	Auto-reload match ARRM is set by hardware to inform application that LPTIM_CNT register's value reached the LPTIM_ARR register's value. Writing 1 to the ARRMCF bit of the LPTIM_ICR register clears the ARRM flag.
0	CMPM	R	0	Compare Match The CMPM bit is set by hardware to inform the application that the LPTIM_CNT register value has reached the value of the LPTIM_CMP register. Writing 1 to the CMPMCF bit of the LPTIM_ICR register clears the CMPM flag.

25.6.2. LPTIM Interrupt Clear Register (LPTIM_ICR)

Address offset: 0x004

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOW MCF	UPC F	ARR OKC F	CMP OKC F	Res.	ARR MCF	CMP MCF
									W	W	W	W		W	W

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved

6	DOWNCF	W	0	Count Direction Changes to Down Clear Flag Writing 1 to this bit clears the DOWN flag in the LPTIM_ISR register. Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
5	UPCF	W	0	Direction changed to up clear sign Writing 1 to this bit clears the UP flag in the LPTIM_ISR register. Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
4	ARROKCF	W	0	Automatic reload register updates OK clear flag. Writing 1 to this bit clears the ARROK flag in the LPTIM_ISR register.
3	CMPOKCF	W	0	Compare Register Update OK Clear Flag Writing 1 to this bit clears the CMPOK flag in the LPTIM_ISR register.
2	Reserved	-	-	Reserved
1	ARRMCF	W	0	Auto-reload match clear flag Writing 1 to this bit clears the ARRM flag in the LPTIM_ISR register.
0	CMPMCF	W	0	Compare Match Clear Flag Writing 1 to this bit clears the CMPM flag in the LPTIM_ISR register.

25.6.3. LPTIM interrupt enable register (LPTIM_IER)

Address offset: 0x008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOW NIE	UPIE	ARR OKIE	CMP OKIE	Res.	ARR MIE	CMP MIE
									RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6	DOWNIE	RW	0	Counter changed from incremental count to decremental count interrupt enabled 0: Disable DOWN interrupt 1: Enable DOWN interrupt Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
5	UPIE	RW	0	The counter is changed from decrement to increment interrupt enabled 0: Disable UP interrupt 1: Enable UP interrupt Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
4	ARROKIE	RW	0	Auto-reload register update OK interrupt enable 0: Disable ARROK interrupt 1: Enable ARROK interrupt
3	CMPOKIE	RW	0	Compare Register Update OK Interrupt Enable 0: Disable CMPOK interrupt 1: Enable CMPOK interrupt
2	Reserved	-	-	Reserved
1	ARRMIE	RW	0	Auto-reload match interrupt enable 0: Disable ARRM interrupt 1: Enable ARRM interrupt
0	CMPMIE	RW	0	Compare Match Interrupt Enable 0: Disable CMPM interrupt 1: Enable CMPM interrupt

25.6.4. LPTIM configuration register (LPTIM_CFGR)

Address offset: 0x00C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENC	Res.	PRELOAD	WAVPOL	WAVE	Res.	TRIGEN[1:0]		Res.

							RW		RW	RW	RW		RW	RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TRIGSEL[2:0]			Res.	PRESC[2:0]			Res.	TRGFLT[1:0]			Res.	Res.	Res.	CKPOL[1:0]		Res.
RW	RW	RW		RW	RW	RW		RW	RW					RW	RW	

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	ENC	RW	0	Encoder Mode Enable The ENC bit controls the Encoder mode 0: Disable encoder mode 1: Enable encoder mode Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
23	Reserved	-	-	Reserved
22	PRELOAD	RW	0	Register update mode. The preload bit controls the LPTIM_ARR register update mode. 0: Update registers after each APB bus write access; 1: The register is updated at the end of the current LPTIM counting cycle; Note: This register is for the LPTIM_ARR register only.
21	WAVPOL	RW	0	Waveform polarity WAVPOL bit control output polarity 0: The LPTIM output reflects the comparison between the LPTIM_CNT and LPTIM_CMP registers 1: The LPTIM output reflects the opposite value of the comparison result between the LPTIM_CNT and LPTIM_CMP registers
20	WAVE	RW	0	Waveform shape. WAVE bit control output shape 0: Disable the setting mode once, PWM or monopulse waveform depends on how the timer is started (CNTSTRT for PWM or SNGSTRT for monopulse waveform). 1: Activate the setting mode once
19	Reserved	-	-	Reserved
18:17	TRIGEN[1:0]	RW	0	Trigger enabling and polarity The TRIGEN bit controls whether the LPTIM counter is started by an external trigger. If the external trigger option is selected, there are two configurations to trigger valid edges: 00: Software trigger (count start is initiated by software) 01: Rising edge is valid edge 10: The falling edge is an effective edge 11: Reserved
16	Reserved	-	-	Reserved
15:13	TRIGSEL[2:0]	RW	0	Trigger selector The TRIGSEL bit selects among the following 8 available sources that will be the trigger source for the LPTIM trigger event: 000: lptim_ext_trig0 (LPTIM1_ETR for LPTIM1, LPTIM2_ETR for LPTIM2) 001: lptim_ext_trig1 (RTC Alarm A) 010: lptim_ext_trig2 (RTC alarm B) 011: lptim_ext_trig3 (TAMP event) 100: lptim_ext_trig4 (COMP1_OUT) 101: lptim_ext_trig5 (COMP2_OUT) 110: lptim_ext_trig6 (reserved) 111: lptim_ext_trig7 (reserved)
12	Reserved	-	-	Reserved
11:9	PRESC[2:0]	RW	0	Clock prescaler. The PRESC bits configure the prescaler division factor. The division coefficient can be selected from the following division coefficients: 000:/1; 001:/2; 010:/4; 011:/8; 100:/16;

				101:/32; 110:/64; 111:/128;
8	Reserved	-	-	Reserved
7:6	TRGFLT[1:0]	RW		Triggered configurable digital filter The TRGFLT value sets the number of consecutive equal sample values that should be detected when a level change occurs on an internal flip-flop before it is considered a valid level transition. An internal clock source must be present to use this feature. 00: Any trigger activity level change is considered a valid trigger 01: The trigger effective level change must be stable for at least 2 clock cycles before it is considered a valid trigger. 10: The trigger effective level change must be stable for at least 4 clock cycles before it is considered an effective trigger. 11: The trigger effective level change must be stable for at least 8 clock cycles before it is considered a valid trigger.
5:3	Reserved	-	-	Reserved
2:1	CKPOL[1:0]	RW	0	Encoder mode selection is active. 00: If LPTIM is configured to encoder mode (set ENC bit), encoder sub-mode 1 is active. 01: If LPTIM is configured to encoder mode (set ENC bit), encoder sub-mode 2 is active. 10: If LPTIM is configured to encoder mode (set ENC bit), encoder sub-mode 3 is active. 11: Reserved Note: This bit field is reserved if LPTIM does not support the encoder mode functionality.
0	Reserved	-	-	Reserved

Note: When configuring the CFGR register, it is recommended to write all configuration bits at once. If configured separately, a 3-core-clock interval is required between the two configurations.

25.6.5. LPTIM control register (LPTIM_CR)

Address offset: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res.	Res.	Res.	Res.	Res.
.	.	.	.	.	.	.	.	.	.	.					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RSTAR	COUNTRS	CNTSTR	SNGSTR	ENABL
.	.	.	.	.	.	.	.	.	.	.	E	T	T	T	E
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	Reserved
4	RSTARE	RW	0	Reset after read enable This bit is set and cleared by software. When RSTARE is set to '1', any read access to LPTIM_CNT register will asynchronously reset LPTIM_CNT register content.
3	COUNTRST	RW	0	Timer counter reset This bit is set by software and cleared by hardware. When set to '1' this bit will trigger a synchronous reset of the LPTIM_CNT counter register. Due to the synchronous nature of this reset, it only takes place after a synchronization delay of 3 LPTIM core clock cycles (LPTIM core clock may be different from APB clock). Caution: COUNTRST must never be set to '1' by software before it is already cleared to '0' by hardware. Software should consequently check that COUNTRST bit is already cleared to '0' before attempting to set it to '1'.
2	CNTSTRT	RW	0	Timer start in Continuous mode This bit is set by the software. If this bit is set when a single pulse mode counting is ongoing, then the timer will not stop at the next match between the LPTIM_ARR and LPTIM_CNT registers. The LPTIM counter keeps counting in continuous mode.

				Note: This bit can be set only when the LPTIM is enabled. This bit is cleared by hardware after synchronization to the kernel clock.
1	SNGSTRT	RW	0	LPTIM start in Single mode This bit is set by software and cleared by hardware. Setting this bit starts the LPTIM in single pulse mode Note: This bit can be set only when the LPTIM is enabled. It will be automatically reset by hardware.
0	ENABLE	RW	0	LPTIM enable. The ENABLE bit is set and cleared by software. 0: LPTIM is disabled 1: LPTIM is enabled

25.6.6. LPTIM comparison register (LPTIM_CMP)

Address offset: 0x014

Reset value: 0x0000 0000

LPTIM1:

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

Bit	Name	R/W	Reset Value	Function
31:0	CMP	RW	0x0000	Compare values. CMP is the comparative value of LPTIM This register can not be updated until LPTIM is enabled.

LPTIM2:

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CMP	RW	0x0000	Compare values. CMP is the comparative value of LPTIM This register can not be updated until LPTIM is enabled.

25.6.7. LPTIM Auto Reload Register (LPTIM_ARR)

Address offset: 0x018

Reset value: 0x0000 0001

LPTIM1:

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

Bit	Name	R/W	Reset Value	Function
31:0	ARR[31:0]	RW	32'h1	Auto reload value ARR is the auto-reload value of LPTIM. This register can not be updated until LPTIM is enabled.

LPTIM2:

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved

15:0	ARR[15:0]	RW	16'h1	Auto reload value ARR is the auto-reload value of LPTIM. This register can not be updated until LPTIM is enabled.

25.6.8. LPTIM Counter Register (LPTIM_CNT)

Address offset: 0x01C

Reset value: 0x0000 0000

LPTIM1:

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

Bit	Name	R/W	Reset Value	Function
31:0	CNT[31:0]	R	0x0000	Counter value When the LPTIM is running with an asynchronous clock, reading the LPTIM_CNT register may return unreliable values. So in this case it is necessary to perform two consecutive read accesses and verify that the two returned values are identical. It should be noted that for a reliable LPTIM_CNT register read access, two consecutive read operations are required and compared. When they are equal, the read access can be considered reliable.

LPTIM2:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	R	0	Counter value When the LPTIM is running with an asynchronous clock, reading the LPTIM_CNT register may return unreliable values. So in this case it is necessary to perform two consecutive read accesses and verify that the two returned values are identical. It should be noted that for a reliable LPTIM_CNT register read access, two consecutive read operations are required and compared. When they are equal, the read access can be considered reliable.

25.6.9. LPTIM Select Register (LPTIM_OR)

Address offset: 0x01C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IN1 [2:1]		IN1 [2:1]		IN2[0]	IN1[0]
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
5:4	IN2 [2:1]	RW	0	IN2SEL [1: 0]: LPTIM input 2 select The IN2SEL bit controls the LPTIM Input 2 multiplexer, which connects LPTIM Input 2 to one of the following available inputs. 00: COMP1 01: COMP2 Others: reserved
3:2	IN1 [2:1]	RW	0	IN2SEL [1: 0]: LPTIM input 1 select The IN2SEL bit controls the LPTIM Input 1 multiplexer, which connects LPTIM Input 1 to one of the following available inputs.

				00: COMP1 01: COMP2 Others: reserved
1	IN2[0]	RW	0	LPTIM input 2 remapping 1: Connect to COMP output indicated by IN2 [2: 1] 0: Connect to GPIO
0	IN1[0]	RW	0	LPTIM input 1 remap 1: Connect to COMP output indicated by IN1 [2: 1] 0: Connect to GPIO

Note: The LPTIM_OR register can only be modified when LPTIM is disabled (ENABLE bit is 0).

26. Independent watchdog (IWDG)

26.1. Introduction to IWDG

Independent watchdog (IWDG) is integrated in the chip, and this module has the characteristics of high-security level, accurate timing and flexible use. The independent watchdog peripheral detects and solves malfunctions due to software failure, and triggers system reset when the counter reaches a given timeout value.

The IWDG is clocked by LSI/LSE, allowing it to remain operational even if the main clock fails.

The IWDG is best suited for applications that require the watchdog to run as a totally independent process outside the main application, but have lower timing accuracy constraints.

26.2. IWDG main features

- Free-running downcounter
- Clock provided by LSI or LSE (can operate in stop or standby mode)
- Supports hardware mode
- Configurable to stop counting in stop mode
- Generates reset when the downcounter value reaches 0x000

26.3. IWDG functional description

26.3.1. IWDG block diagram

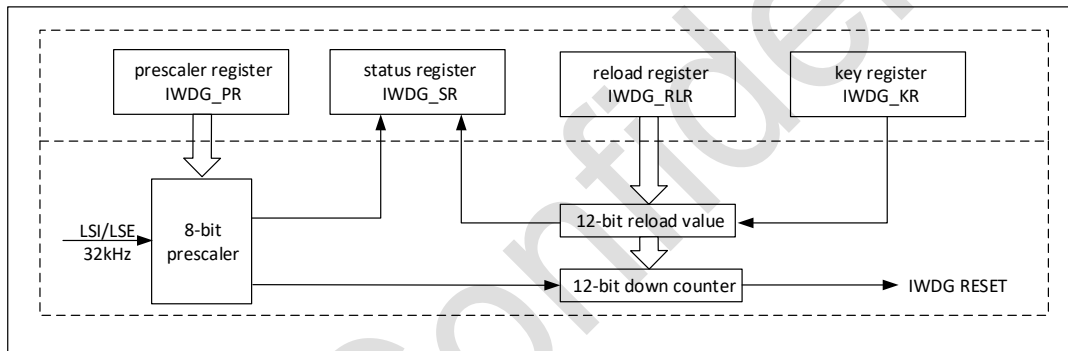


Figure 26-1 Independent watchdog block diagram

When writing 0x0000 CCCC to the IWDG key register (IWDG_KR), the counter starts decrementing from 0xFFFF. When it reaches the end of count value (0x000) a reset signal is generated (IWDG reset).

Whenever 0x0000 AAAA is written to the IWDG key register, the value of IWDG_RLR (reload register) is reloaded into the counter, and no reset is generated by IWDG.

Hardware watchdog

If the hardware watchdog is enabled in the user option bytes loaded at power-up, the IWDG is automatically enabled (with the LSI clock enabled as the counting clock for IWDG), and a reset signal is generated if the IWDG key register is not rewritten by software before the counter reaches its final value.

26.3.2. Register access protection

Write access to registers IWDG_PR and IWDG_RLR is protected. To modify them, the user must first write 0x0000 5555 to the IWDG_KR register. A write access to this register with a different value will break the sequence and register access will be protected again. This implies that it is the case of the reload operation (writing 0xAAAA).

The status register indicates whether the values of IWDG_PR and IWDG_RLR registers are being updated.

26.3.3. Debug mode

If the CPU enters debug mode, whether the IWDG continues counting or stops depends on the configuration of DBG_IWDG_STOP in the DBGMCU module.

26.3.4. Stop mode

The IWDG_STOP bit in the option bytes of the Flash information area controls whether the IWDG continues normal counting or stops counting when the system enters stop mode.

26.4. IWDG registers

26.4.1. IWDG key register (IWDG_KR)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY[15:0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	KEY[15:0]	W	0x00	Key value Software must write 0xAAAA to this register at regular intervals; otherwise, the watchdog will generate a reset when the counter reaches 0. 0x5555: Indicates permission to access IWDG_PR and IWDG_RLR registers Writing the key value 0xCCCC starts the watchdog (except if the hardware watchdog option is selected).

26.4.2. IWDG prescaler register (IWDG_PR)

Offset address: 0x04

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2:0	PR[2:0]	RW	0	Prescaler divider They are written by software to select the prescaler divider feeding the counter clock. PVU bit of the IWDG status register (IWDG_SR) must be reset in order to be able to change the prescaler divider. 000: 4 division 001: 8 division 010: 16 division 011: 32 division 100: 64 division 101: 128 division 110: 256 division 111: 256 division

26.4.3. IWDG reload register (IWDG_RLR)

Offset address: 0x08

Reset value: 0x0000 0FFF

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

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	RL[11:0]	RW	0xFFFF	IWDG counter reload value They are written by software to define the value to be loaded in the watchdog counter each time the value 0xAAAA is written in the IWDG key register

				(IWDG_KR). The watchdog counter counts down from this value. The timeout period is a function of this value and the clock prescaler. The RVU bit in the IWDG status register (IWDG_SR) must be reset to be able to change the reload value.

26.4.4. IWDG status register (IWDG_SR)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RVU	PVU
														R	R

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved
1	RVU	R	0	Watchdog counter reload value update This bit is set by hardware to indicate that an update of the window value is ongoing. It is reset by hardware when the reload value update operation is completed.
0	PVU	R	0	Watchdog prescaler value update This bit is set by hardware to indicate that an update of the prescaler value is ongoing. It is reset by hardware when the prescaler update operation is completed.

Note: Before updating IWDG_PR and IWDG_SR.RLR, wait for IWDG_PVU and IWDG_SR.RVU to be 0, respectively. However, after updating the prescaler and/or the reload value, it is not necessary to wait until RVU or PVU is reset before continuing code execution.

27. System Window Watchdog (WWDG)

27.1. Introduction to WWDG

The System Window Watchdog (WWDG) is typically used to detect software faults caused by external disturbances or unforeseen logical conditions that cause the application to deviate from its normal operation sequence. Unless the program refreshes the value of the downcounter before the T6 bit becomes 0, the watchdog circuit will generate a system reset when the preset time period is reached. A system reset will also be generated if the 7-bit downcounter value in the control register is refreshed before the downcounter reaches the window register value. This implies that the counter must be refreshed in a limited window.

The WWDG clock is provided by the APB clock after prescaling, and it detects abnormal late or early operations of the application through a configurable time window.

WWDG is most suitable for applications that require the watchdog to function within a precise timing window.

27.2. WWDG main features

- Supports hardware mode
- Programmable free-running downcounter
- Conditional reset
 - When the downcounter is below 0x40 (if the watchdog is activated)
 - Reset (if watchdog activated) if the downcounter is reloaded outside the window
- Early Wakeup Interrupt (EWI): Triggered when the downcounter equals 0x40 (if this function is enabled and the watchdog is activated)

27.3. WWDG functional description

If WWDG is activated (WWDG_CR.WDGA is set), and when the 7-bit downcounter (T[6:0] bits) decrements from 0x40 to 0x3F (T6 is cleared), a reset is triggered. If the counter value is greater than the memory window register value when the software reloads the counter, a reset is generated.

The application must periodically write to the WWDG_CR register during normal operation to prevent a reset. This write operation can only be performed when the counter value is less than the window register value and above 0x3F. The value to be stored in the WWDG_CR register must be between 0xFF and 0xC0.

27.3.1. WWDG Block Diagram

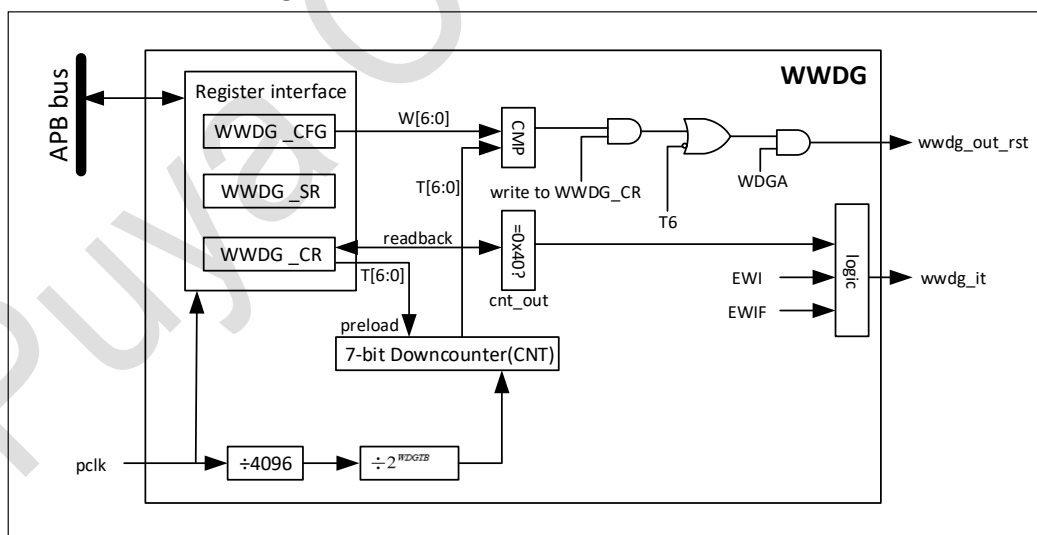


Figure 27-1 Window Watchdog Block Diagram

27.3.2. Enable Watchdog

When the WWDG_SW bit in the user option byte selects "Software Window Watchdog," the watchdog is always disabled after reset. The WWDG module is then enabled by setting the WDGA bit in the WWDG_CR register, and it cannot be disabled again unless a reset occurs.

When the WWDG_SW in the user option byte selects "Hardware WWDG," the watchdog is always enabled after reset and cannot be disabled.

27.3.3. Controls the countdown

This downcounter is free-running, counting down even if the watchdog is disabled. When enabling the watchdog, the T6 bit must be set to 1 to prevent an immediate reset.

The T[5:0] bits contain the incremental counter value before the WWDG generates a reset. This data indicates the time delay before the watchdog generates a reset.

The configuration register (WWDG_CFR) contains the upper window limit: to avoid a reset, the down-counter must be reloaded when its value is less than the window register value and greater than 0x3F.

27.3.4. Watchdog Interrupt Advanced Feature

The early wakeup interrupt (EWI) can be used if specific safety operations or data logging must be performed before the actual reset is generated. Enable the EWI interrupt by setting the EWI bit in the WWDG_CFR register. An EWI interrupt is generated when the down-counter reaches 0x40. Before resetting the device, the corresponding interrupt service routine (ISR) can be used to trigger specific operations (e.g., communication or data logging).

In some applications, the EWI interrupt can be used to manage software system checks and/or system recovery/functional degradation without generating a WWDG reset. In this case, the corresponding ISR can reload the WWDG counter to avoid a WWDG reset and then trigger the required operations.

Clear the EWI interrupt by writing 0 to the EWIF bit in the WWDG_SR register.

Note: If the EWI interrupt cannot be serviced due to system lock in higher-priority tasks, a WWDG reset will eventually occur.

27.3.5. How to Set the Watchdog Timeout

For the WWDG_CR register, typically write 1 to the T6 bit to avoid generating a reset.

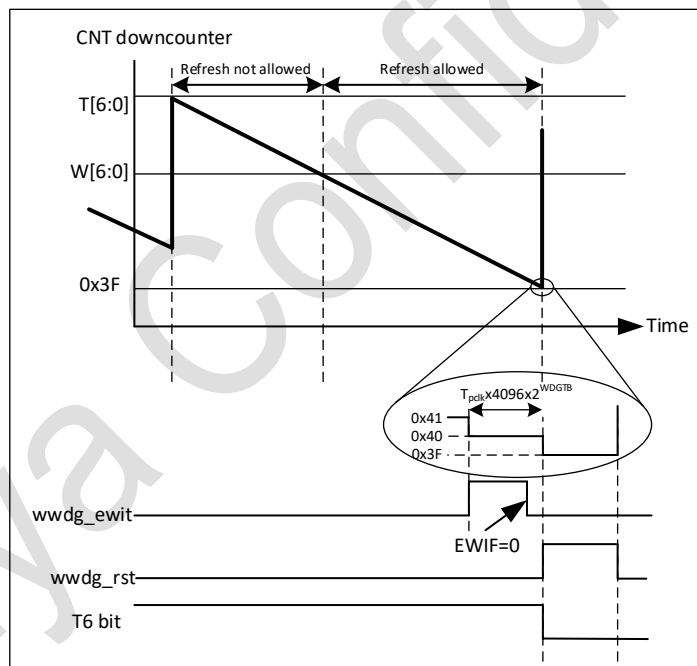


Figure 27-2 Window Watchdog Timing Diagram

The formula to calculate the WWDG timeout value is given by:

$$t_{\text{WWDG}} = t_{\text{PCLK}} \times 4096 \times 2^{\text{WDGTB}[1:0]} \times (\text{T}[5:0] + 1) \quad (\text{ms})$$

with:

t_{WWDG} : WWDG timeout

t_{PCLK} : APB clock cycle, measured in ms

4096: value corresponding to internal divider

For example, assuming an APB frequency of 72 MHz, set WDGTB[1:0] to 3 and T[5:0] to 63:

$$t_{\text{WWDG}} = (1/72000) \times 4096 \times 2^3 \times (63 + 1) = 29.127 \text{ ms.}$$

27.3.6. Debug mode

If the CPU enters debug mode, whether the WWDG continues counting or stops depends on the configuration of DBG_WWDG_STOP in the DBGMCU module.

27.4. WWDG registers

27.4.1. WWDG Control Register (WWDG_CR)

Offset address: 0x00

Reset value: 0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDGA	T [6: 0]						
								RS	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	WDGA	RS	0	Activation bit (Activation bit) This bit is set to '1' by software and can only be cleared to '0' by hardware after a reset. When WDGA = 1, the watchdog can generate a reset. 0: No Watch Dogs 1: Enable the watchdog
6:0	T [6: 0]	RW	7'h7F	7-bit counter (MSB to LSB). These bits contain the value of the watchdog counter. Decrements by 1 every (4096x2WDGTB) PCLK cycles. When the count value changes from 0x40 to 0x3F (T[6] becomes 0), a watchdog reset is generated.

27.4.2. WWDG Configuration Register (WWDG_CFR)

Offset address: 0x04

Reset value: 0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	EWI	WDGTB[1:0]		W [6: 0]						
						RS	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	EWI	RS	0	Early wake-up interrupt If this bit is set to 1, an interrupt is generated when the counter value reaches 0x40. This interrupt is cleared by hardware after a reset.
8:7	WDGTB[1:0]	RW	2'b0	Timer time base The time base of the prescaler can be modified as follows: 00: CK counter clock (PCLK divided by 4096) divided by 1 01: CK counter clock (PCLK divided by 4096) divided by 2 10: CK counter clock (PCLK divided by 4096) divided by 4 11: CK counter clock (PCLK divided by 4096) divided by 8
6:0	W [6: 0]	RW	7'h7F	7-bit window value. These bits contain the window value to be compared to the downcounter.

27.4.3. WWDG Status Register (WWDG_SR)

Offset address: 0x08

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	Reserved
0	EWIF	RC_W0	0	Early wake-up interrupt flag. When the counter value reaches 0x40, this bit is set to 1 by hardware. Writing 0 clears it, while writing 1 has no effect. This bit is also set to 1 when interrupts are disabled.

28. Real-Time Clock (RTC)

28.1. Introduction to RTC

The Real-Time Clock (RTC) provides an automatic wake-up unit for managing all low-power modes. The Real-Time Clock (RTC) is an independent BCD timer/counter. The RTC provides a time- of-day clock/calendar with programmable alarm interrupts.

As long as the power supply voltage remains within the operating range, the RTC will continue to operate regardless of the device's state (run mode, low-power mode, or in reset).

28.2. RTC main features

- The calendar includes sub - seconds, seconds, minutes, hours (in 12 - or 24 - hour format), day of the week, day, month, and year, presented in BCD (Binary - Coded Decimal) format
 - Automatically adjusts the number of days in a month to 28, 29 (in leap years), 30, or 31.
 - Two programmable alarms.
 - Allow in - runtime correction of 1 to 32,767 RTC clock pulses for synchronization with the main clock. This can be used to synchronize it with a master clock.
 - Reference clock detection: A more accurate secondary clock source (50 or 60 Hz) can be used to improve calendar accuracy.
 - The digital calibration circuit has a resolution of 0.95 ppm to compensate for quartz crystal errors.
 - The timestamp feature can be used to save calendar content. This function can be triggered by an event on the timestamp pin or by a tamper event.
 - A 17-bit auto-reload wake-up timer (WUT) for periodic events, with programmable resolution and period.
 - The RTC clock source could be any of the following ones:
 - HSE clock divided by 32
 - LSE clock
 - LSI clock
- When clocked by LSE or LSI, the RTC can operate in all low-power modes.
- Three dedicated maskable interrupts:
 - Alarm interrupt
 - Timestamp interrupt
 - Wake-up timer interrupt

28.3. RTC functional description

28.3.1. RTC block diagram

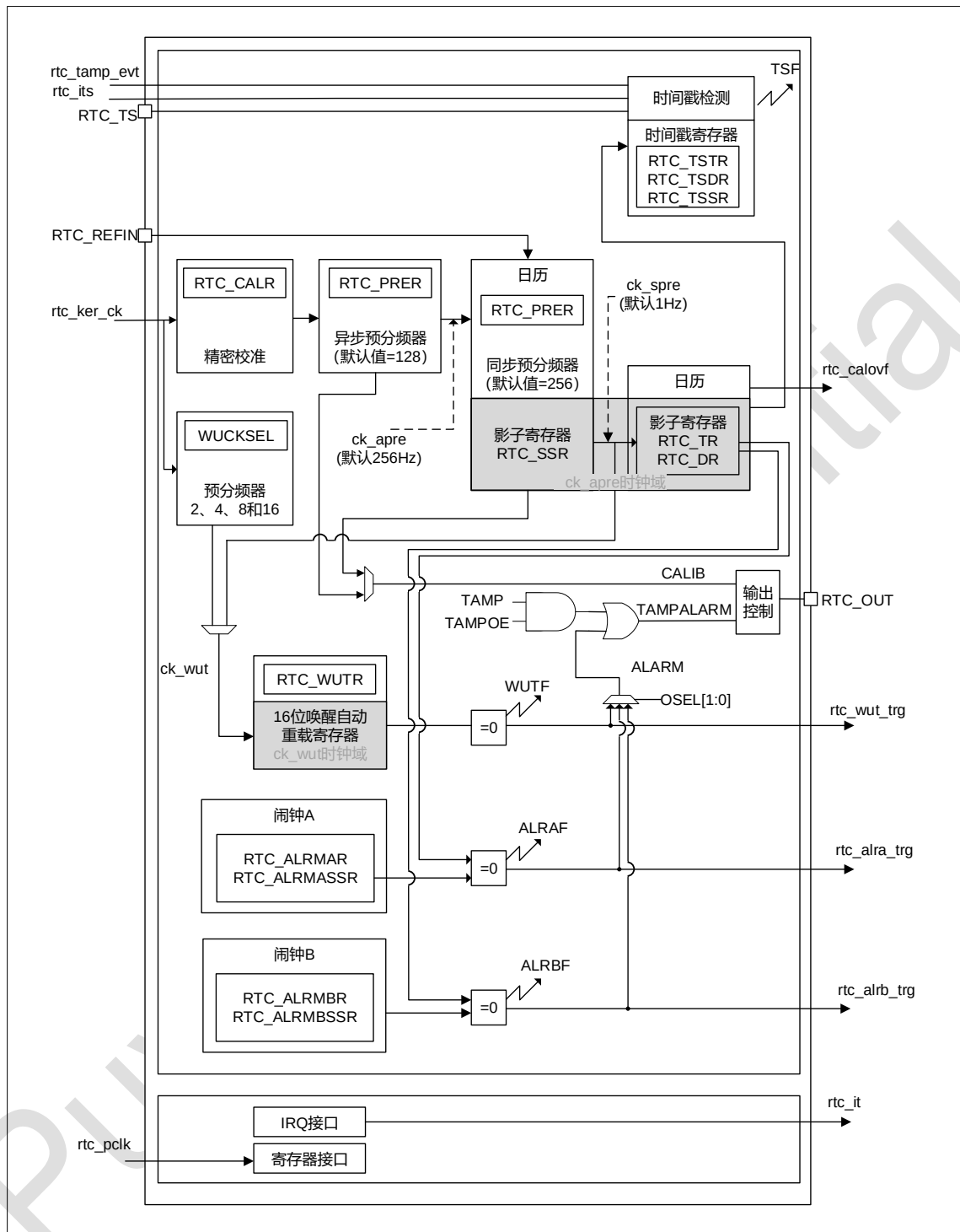


Figure 28-1 RTC block diagram

28.3.2. RTC pins and internal signals

Table 28-1 RTC input/output pins

Pin name	I/O	Description
RTC_TS	I	RTC timestamp input
RTC_REFIN	I	RTC 50 or 60 Hz reference clock input
RTC_OUT	Output	RTC output, with the following selectable options: --CALIB: 512 Hz or 1 Hz clock output (LSE frequency is 32.768 kHz). This output can be enabled by setting the COE bit in the RTC_CR register to 1. --TAMPALRM: This output is the logical OR result of the TAMP and ALARM outputs. The ALARM can be enabled by configuring the OSEL[1:0] bits in the RTC_CR register, allowing selection of Alarm A, Alarm B, or wake-up output. TAMP can be enabled by setting the TAMPOE bit in the RTC_CR register to 1 to select the tamper event output.

Table 28-2 RTC Internal Input/Output Signals

Pin name	I/O	Description
rtc_ker_ck	I	RTC core clock, also referred to as RTCCLK in this document.
rtc_pclk	I	RTC APB clock
rtc_its	I	RTC internal timestamp event
rtc_tamp_evt	I	Tamper event detected (internal or external) in the TAMP peripheral
rtc_it	Output	RTC interrupts
rtc_alra_trg	Output	RTC Alarm A event detection trigger
rtc_alrb_trg	Output	RTC Alarm B event detection trigger
rtc_wut_trg	Output	RTC wakeup timer event detection trigger
rtc_calovf	Output	RTC calendar overflow

28.3.3. RTC and TAMP controlled GPIO

VDDK domain IO PA[8] includes RTC and TAMP functions. These functions are directly controlled by RTC and TAMP, regardless of the GPIO AF function configuration.

RTC_OUT, RTC_TS, and TAMP_IN are mapped to the same pin (PA8). The RTC and TAMP functions mapped on PA8 are available in all low-power modes.

The output mechanism follows the priority order shown in the table below.

PA8 Pin function		OSEL[1:0] (ALARM output en- able)	TAMPOE (TAMPER output en- able)	COE bit (CALIB output en- able)	TAMPALRM_TYPE	TAMPALRM_PU	TAMPE (TAMP_IN input ena- ble)	TSE (RTC_TS input en- able)
TAMPALRM push-pull output		01 or 10 or 11	0	Irrele- vant	0	0	Irrelevant	Irrelevant
		00	1					
		01 or 10 or 11	1					
TAMPALRM open-drain output	No pull- up or pull- down	01 or 10 or 11	0	Irrele- vant	1	0	Irrelevant	Irrelevant
		00	1					
		01 or 10 or 11	1					
	Internal pull- up	01 or 10 or 11	0	Irrele- vant	1	1	Irrelevant	Irrelevant
		00	1					
		01 or 10 or 11	1					
CALIB output push- pull		00	0	1	Irrelevant	Irrelevant	Irrelevant	Irrelevant
TAMP_IN input float- ing		00	0	0	Irrelevant	Irrelevant	1	0
RTC_TS and TAMP_IN input float- ing		00	0	0	Irrelevant	Irrelevant	1	1
RTC_TS input floating		00	0	0	Irrelevant	Irrelevant	0	1
Wake-up pin or stand- ard GPIO		00	0	0	Irrelevant	Irrelevant	0	0

28.3.4. RTC clock and prescaler

By configuring the RCC_BDCR.RTCSEL register, the RTC clock source (RTCCLK) can be selected from the LSE clock, LSI clock, or HSE clock.

The programmable prescaler stages generate a 1 Hz clock for calendar updates. To minimize power consumption, the prescaler is divided into two programmable prescalers:

- A 7-bit asynchronous prescaler configured via the PREDIV_A bits in the RTC_PRER register.
- A 15-bit synchronous prescaler configured via the PREDIV_S bits in the RTC_PRER register.

Note: When using both prescalers, it is recommended to configure the asynchronous prescaler to a higher value to minimize power consumption.

To obtain a 1 Hz internal clock (ck_spre) using an LSE frequency of 32.768 kHz, set the asynchronous prescaler coefficient to 128 and the synchronous prescaler coefficient to 256.

The minimum division factor is 1 and the maximum division factor is 222.

This corresponds to a maximum input frequency of approximately 4 MHz.

fck_apre can be derived using the following formula:

$$f_{CK\_APRE} = \frac{f_{RTCCLK}}{PREDIV\_A + 1}$$

The ck_apre clock is used to clock the binary RTC_SSR sub-second down counter. When this counter reaches 0, the RTC_SSR is reloaded with the content of PREDIV_S.

fck_spre can be derived using the following formula:

$$f_{CK\_SPRE} = \frac{f_{RTCCLK}}{(PREDIV\_S + 1) \times (PREDIV\_A + 1)}$$

The ck_spre clock can be used both for calendar updates and as the timebase for the 16-bit wake-up auto-reload timer. For shorter timeout periods, the 16-bit wake-up auto-reload timer can also operate with the RTCCLK divided by a programmable 4-bit asynchronous prescaler.

28.3.5. Real-time clock and calendar

The RTC calendar time and date registers can be accessed via shadow registers synchronized with the PCLK (APB clock). They can also be accessed directly in order to avoid waiting for the synchronization duration.

- RTC_SSR corresponds to sub-seconds
- RTC_TR corresponds to time
- RTC_DR corresponds to date

Every RTCCLK cycle, the current calendar value is copied to the shadow registers, and the RSF bit in the RTC_ICSR register is set to 1. The copy is not performed in Stop and Standby mode. Upon exiting these modes, the shadow registers are updated within a maximum of 4 RTCCLK cycles.

When the application reads the calendar registers, it accesses the content of the shadow registers. You can also directly access the calendar registers by setting the BYPSHAD control bit in the RTC_CR register to 1. By default, this bit is cleared, and the user accesses the shadow registers.

When reading the RTC_SSR, RTC_TR, or RTC_DR registers in BYPSHAD=0 mode, the APB clock frequency (fAPB) must be at least 7 times the RTC clock frequency (fRTCCLK).

The shadow registers are reset by system reset.

28.3.6. Programmable alarm

The RTC unit provides two programmable alarms, Alarm A and Alarm B. The following description applies to Alarm A but is also valid for Alarm B.

The programmable alarm function can be enabled via the ALRAE bit in the RTC_CR register.

The ALRAF flag is set to 1 if the calendar sub-seconds, seconds, minutes, hours, date, or day matches the value programmed in the alarm registers RTC_ALRMASR and RTC_ALRMAR. Each calendar field can be individually selected via the MSKx bits in the RTC_ALRMAR register and the MASKSSx bits in the RTC_ALRMASR register.

The alarm interrupt can be enabled via the ALRAIE bit in the RTC_CR register.

Note: If the seconds field is selected (MSK1 bit in RTC_ALRMAR is reset), the synchronous prescaler division factor set in the RTC_PRER register must be at least 3 to ensure proper alarm operation.

Alarm A and Alarm B (if enabled via the OSEL[0:1] bits in the RTC_CR register) can be connected to the TAMPALRM output. The TAMPALRM output polarity can be configured via the POL bit in the RTC_CR register.

28.3.7. Periodic auto-wakeup

The periodic wakeup flag is generated by a 16-bit programmable auto-reload down-counter. The wakeup timer range can be extended to 17 bits.

This wakeup function can be enabled via the WUTE bit in the RTC_CR register.

The wakeup timer clock input `ck_wut` can be:

- RTC clock (RTCCLK) divided by 2, 4, 8, or 16.

When RTCCLK is LSE (32.768 kHz), the configurable wakeup interrupt period ranges from 122 μ s to 32 s with a resolution as low as 61 μ s.

- `ck_spre` (typically a 1 Hz internal clock)

When the `ck_spre` frequency is 1 Hz, the achievable wakeup time ranges from 1s to around 36h with a resolution of 1 second. This large programmable time range is divided in 2 parts:

– 1s to 18h when `WUCKSEL[2:1] = 10`

– Approximately 18h to 36h when `WUCKSEL[2:1] = 11`. In the latter case, 216 is added to the current value of the 16-bit counter. After the initialization sequence is completed, the timer starts counting down. When the wakeup function is enabled, the down-counting remains active in low-power modes. Additionally, when the timer reaches 0, the `WUTF` flag in the `RTC_SR` register is set to 1, and the wakeup counter is automatically reloaded with its reload value (`RTC_WUTR` register value).

The `WUTF` flag must be cleared by software afterwards.

When enabling the periodic wakeup interrupt by setting the `WUTIE` bit in the `RTC_CR` register to 1, it causes the device to exit low-power mode.

If the periodic wakeup flag is enabled through the `OSEL[1:0]` bits of the `RTC_CR` register, this flag can be connected to the `TAMPALRM` output. The `TAMPALRM` output polarity can be configured via the `POL` bit in the `RTC_CR` register.

System reset, as well as low-power modes (Sleep, Stop and Standby) have no influence on the wakeup timer.

28.3.8. RTC Initialization and Configuration

28.3.8.1. RTC register access

RTC registers are 32-bit registers. Except for read accesses to the calendar shadow registers when `BYPHAD=0`, the APB interface introduces 2 wait cycles when accessing RTC registers.

28.3.8.2. RTC register write protection

After a system reset, the RTC registers can be protected against unauthorized write access via the `DBP` bit in the power control peripheral. The `DBP` bit must be set to 1 to enable write access to RTC registers.

After an RTC domain reset, not all RTC registers are write-protected.

Write operations to protected RTC registers are enabled by writing a key to the write protection register `RTC_WPR`.

To unlock the write protection of protected RTC registers, the following steps must be performed:

- Write 0xCA to the `RTC_WPR` register.
- Write 0x53 to the `RTC_WPR` register.

Writing a wrong key reactivates the write protection.

The protection mechanism is not affected by system reset.

28.3.8.3. Calendar initialization and configuration

To program the initial time and date calendar values, including the time format and the prescaler configuration, the following sequence is required:

1. Set the `INIT` bit in the `RTC_ICSR` register to 1 to enter RTC initialization mode. In this mode, the calendar counter is stopped and its value can be updated.
2. Poll the `INITF` bit in the `RTC_ICSR` register. Initialization mode is entered when `INITF` is set to 1. This takes approximately 3 RTCCLK clock cycles (due to clock synchronization).
3. Program the two prescaler coefficients in the `RTC_PRER` register to generate a 1 Hz clock for the calendar counter.
4. Load the initial time and date values into the shadow registers (`RTC_TR` and `RTC_DR`), then configure the time format (12 or 24-hour) via the `FMT` bit in the `RTC_CR` register.
5. Exit initialization mode by clearing the `INIT` bit. Subsequently, the actual calendar counter value is automatically loaded, and counting resumes after 4 RTCCLK clock cycles.

When the initialization sequence is complete, the calendar starts counting.

Note: After a system reset, the application can read the `INITS` flag in the `RTC_ICSR` register to check if the calendar has been initialized. If this flag is 0, it indicates that the calendar has not been initialized since the system reset, and its year field remains at the RTC domain reset default value (0x00).

To read the calendar after initialization, software must first check if the `RSF` flag in the `RTC_ICSR` register is set to 1.

28.3.8.4. Daylight saving time

Daylight saving time can be managed through the SUB1H, ADD1H, and BKP bits of the RTC_CR register.

Using SUB1H or ADD1H, the software can subtract or add an hour in the calendar with just a single operation, eliminating the need to perform the entire initialization process.

Additionally, software can use the BKP bit to record whether this operation has been performed.

28.3.8.5. Programming the alarm

A similar procedure must be followed to program or update the programmable alarms. The procedure below is given for alarm A but can be translated in the same way for alarm B.

1. Clear the ALRAE bit in RTC_CR to disable Alarm A.
2. Program the Alarm A registers (RTC_ALRMASR/RTC_ALRMAR).
3. Set the ALRAE bit in the RTC_CR register to re-enable Alarm A.

Note: Due to clock synchronization, each change to the RTC_CR register takes effect after approximately 3 RTCCLK clock cycles.

28.3.8.6. Programming the wakeup timer

To configure or change the auto-reload value of the wakeup timer (WUT[15:0] in RTC_WUTR), follow this sequence:

1. Clear WUTE in RTC_CR to disable the wakeup timer.
2. Poll WUTWF in RTC_ICSR until this bit is set to 1, ensuring access to the wakeup auto-reload counter and WUCKSEL[2:0] bits. This step must be skipped in calendar initialization mode. This takes approximately 3 RTCCLK clock cycles (due to clock synchronization).
3. Program the wakeup auto-reload value WUT[15:0] and select the wakeup clock (WUCKSEL[2:0] bits in RTC_CR).

Set the WUTE bit in the RTC_CR register to re-enable the timer. The wakeup timer resumes decrementing. Due to clock synchronization, the WUTWF bit is also cleared after WUTE is cleared, but this takes up to 3 RTCCLK clock cycles.

28.3.9. Reading the Calendar

To correctly read the RTC calendar registers (RTC_SSR, RTC_TR, and RTC_DR) when the BYPSHAD control bit in RTC_CR is cleared, the APB clock frequency (fPCLK) must be at least seven times the RTC clock frequency (fRTCCLK). This ensures a secure behavior of the synchronization mechanism.

If the APB clock frequency is less than seven times the RTC clock frequency, software must read the calendar time and date registers twice. Only when the two RTC_TR read results match can the data be considered correct. Otherwise a third read access must be done. In any case, the APB1 clock frequency must not be lower than the RTC clock frequency.

Each time the values in the calendar registers are copied to the shadow registers RTC_SSR, RTC_TR, and RTC_DR, the RSF bit in the RTC_ICSR register is set to 1. This copy operation occurs every other RTCCLK cycle. To ensure these three values are from the same point in time, reading RTC_SSR or RTC_TR locks the values in the high-order calendar shadow registers until RTC_DR is read. To prevent the software from performing read access to the calendar at intervals shorter than 1 RTCCLK cycle: After the first calendar read, the RSF must be cleared by software, and the software must wait until RSF is set to 1 before reading the RTC_SSR, RTC_TR, and RTC_DR registers again.

After waking up from a low-power mode (Stop or Standby mode), the RSF must be cleared by software. Afterward, the software must wait until RSF is set to 1 again before reading the RTC_SSR, RTC_TR, and RTC_DR registers.

The RSF bit must be cleared after wake-up, not before entering low-power mode.

After a system reset, the software must wait until RSF is set to 1 before reading the RTC_SSR, RTC_TR, and RTC_DR registers. Indeed, a system reset resets the shadow registers to their default values.

After initialization, the software must wait until RSF is set to 1 before reading the RTC_SSR, RTC_TR, and RTC_DR registers. After synchronization, the software must wait until RSF is set to 1 before reading the RTC_SSR, RTC_TR, and RTC_DR registers.

When the BYPSHAD control bit in the RTC_CR register is set to 1 (bypassing the shadow registers)

Reading the calendar registers directly retrieves values from the calendar counter, eliminating the need to wait until the RSF bit is 1. This is especially useful after exiting from low-power modes (Stop or Standby), since the shadow registers are not updated during these modes.

When BYPSHAD is set to 1, if an RTCCLK edge occurs between two read accesses to the registers, the results from different registers may be inconsistent with each other. Additionally, if an RTCCLK edge occurs during a read operation, it may cause the value of one of the registers to be incorrect. The software must read all the registers twice, and then compare the results to confirm that the data is coherent and correct. Alternatively, the software can just compare the two results of the least-significant calendar register.

Note: When BYPSHAD=1, the instruction to read the calendar registers requires an additional APB cycle to complete.

28.3.10. Resetting the RTC

The calendar shadow registers (RTC_SSR, RTC_TR, and RTC_DR) and certain bits of the RTC status register (RTC_ICSR) are reset to their default values by all available system reset sources.

Conversely, the following registers are reset to their default values by the RTC domain reset and are unaffected by system resets: RTC current calendar registers, RTC control register (RTC_CR), prescaler register (RTC_PRER), RTC calibration register (RTC_CALR), RTC shift register (RTC_SHIFTR), RTC timestamp registers (RTC_TSSSR, RTC_TSTR, and RTC_TSDR), wake-up timer register (RTC_WUTR), and Alarm A and Alarm B registers (RTC_ALRMASR/RTC_ALRMAR and RTC_ALRMBSSR/RTC_ALRMBR).

Additionally, when the clock is provided by the LSE, if the reset source is not an RTC domain reset source (for details about RTC clock sources unaffected by system reset, refer to RCC), the RTC will remain operational during a system reset. When an RTC domain reset occurs, the RTC stops functioning, and all RTC registers are reset to their default values.

28.3.11. RTC Synchronization

The RTC can be synchronized with a high-precision remote clock. After reading the sub-second field (RTC_SSR or RTC_TSSSR), the precise deviation between the remote clock's time and the RTC can be calculated. Subsequently, the RTC_SHIFTR can be used to "shift" the RTC's clock by a fraction of a second, adjusting it to eliminate this deviation.

RTC_SSR contains the value of the synchronous prescaler counter. This allows calculating the RTC's accurate time with a resolution as fine as $1/(\text{PREDIV}_S + 1)$ seconds. Therefore, increasing the value of the synchronous prescaler (PREDIV_S[14:0]) can improve the resolution. Setting PREDIV_S to 0x7FFF achieves the maximum allowed resolution (30.52 μ s at a clock frequency of 32768 Hz).

However, increasing PREDIV_S requires reducing PREDIV_A to maintain the synchronous prescaler's output at 1 Hz.

This increases the output frequency of the asynchronous prescaler, consequently raising the RTC's dynamic power consumption.

The RTC can be fine-tuned using the RTC Shift Control Register (RTC_SHIFTR). Writing to RTC_SHIFTR with a resolution of $1/(\text{PREDIV}_S + 1)$ seconds can shift the clock (delay or advance) by up to 1 second. During this shift operation, the SUBFS[14:0] value is added to the synchronous prescaler counter SS[15:0], which delays the clock. If the ADD1S bit is set to 1 simultaneously, it adds one second while subtracting a fraction of a second, thereby advancing the clock.

Note: Before initiating a shift operation, the user must verify that SS[15] = 0 to ensure no overflow occurs.

When writing to the RTC_SHIFTR register to initiate a shift operation, the hardware sets the SHPF flag to 1, indicating a pending shift operation. This bit is cleared by hardware as soon as the shift operation has completed.

Note: This synchronization feature is incompatible with the reference clock detection function: firmware must not write to RTC_SHIFTR when REFCKON = 1.

28.3.12. RTC Reference Clock Detection

The RTC calendar update can be synchronized with the reference clock RTC_REFIN (typically the mains frequency, 50 Hz or 60 Hz).

The accuracy of the RTC_REFIN reference clock should be higher than that of the 32.768 kHz LSE clock. When RTC_REFIN detection is enabled (by setting the REFCKON bit in RTC_CR to 1), the calendar is still clocked by the LSE, while RTC_REFIN is used to compensate for the inaccurate calendar update frequency (1 Hz).

Each 1 Hz clock edge is compared with the nearest RTC_REFIN clock edge (if an edge is found within a given time window). In most cases, the two clock edges are properly aligned. When the 1 Hz clock deviates due to the inaccuracy of the LSE clock, the RTC slightly shifts the 1 Hz clock so that

subsequent 1 Hz clock edges can align. Thanks to this mechanism, the calendar becomes as precise as the reference clock.

The RTC uses a 256 Hz clock (ck_apre) generated by a 32.768 kHz quartz to detect the presence of a reference clock source. A detection is performed during a time window approximately at each calendar update (every 1 second). When the first reference clock edge is detected, this window equals 7 ck_apre cycles. Subsequent calendar updates use a smaller window of 3 ck_apre cycles.

Each time a reference clock is detected within the window, the asynchronous prescaler outputting the ck_apre clock is forced to reload. This operation has no effect when the reference clock is aligned with the 1 Hz clock, as the prescaler reloads at the same moment. When the clocks are misaligned, the reload operation fine-tunes subsequent 1 Hz clock edges to align them with the reference clock.

If the reference clock stops (no reference clock edge appears within 3 ck_apre windows), the calendar will continue updating based solely on the LSE clock. The RTC then waits for the reference clock using a large detection window (7 ck_apre cycles) centered on the ck_spre edge.

After enabling RTC_REFIN detection, PREDIV_A and PREDIV_S must be set to their default values:

- PREDIV_A = 0x007F
- PREDIV_S = 0x00FF

Note: RTC_REFIN clock detection is not available in Standby mode.

28.3.13. RTC Precision Digital Calibration

The RTC frequency can be digitally calibrated with a resolution of approximately 0.954 ppm, covering a range of -487.1 ppm to +488.5 ppm.

Frequency correction is performed using a series of fine adjustments (adding and/or removing individual RTCCLK pulses). These adjustments are distributed very evenly, so the RTC calibration works quite well, even when observed over short periods.

When the input frequency is 32768 Hz, the precision digital calibration cycle is approximately 220 RTCCLK pulses, which equals 32 seconds. This cycle is maintained by a 20-bit counter cal_cnt[19:0] clocked by RTCCLK.

The precision digital calibration register (RTC_CALR) specifies the number of RTCCLK clock cycles to be subtracted within the 32-second period:

- When bit CALM[0] is set to 1, only 1 pulse is subtracted within the 32-second period.
- When CALM[1] is set to 1, 2 cycles are subtracted.
- When CALM[2] is set to 1, 4 cycles are subtracted.
- Similarly, when CALM[8] is set to 1, 256 clock cycles are subtracted.

Note: CALM[8:0] (RTC_CALR) specifies the number of RTCCLK pulses to be subtracted within the 32-second period. When bit CALM[0] is set to 1, only 1 pulse is subtracted within the 32-second period (when cal_cnt[19:0] = 0x80000); when CALM[1] is set to 1, 2 cycles are subtracted (when cal_cnt = 0x40000 and 0xC0000); when CALM[2] is set to 1, 4 cycles are subtracted (when cal_cnt = 0x20000/0x60000/0xA0000/0xE0000); similarly, when CALM[8] is set to 1, 256 clock cycles are subtracted (when cal_cnt = 0xFF800).

With appropriate resolution, CALM can reduce the RTC frequency by up to 487.1 ppm, while CALP can be used to increase the frequency by 488.5 ppm. Setting CALP to 1 effectively inserts an additional RTCCLK pulse every 211 RTCCLK cycles, meaning 512 clock cycles are added per 32-second period.

When used in combination with CALM and CALP, a deviation ranging from -511 to +512 RTCCLK cycles can be applied within the 32-second period, corresponding to a calibration range of -487.1 ppm to +488.5 ppm with a resolution of approximately 0.954 ppm.

If the input frequency (FRTCCLK) is known, the effective calibrated frequency (FCAL) can be calculated using the following formula:

$$FCAL = FRTCCLK \times [1 + (CALP \times 512 - CALM) / (220 + CALM - CALP \times 512)]$$

28.3.13.1. Calibration under PREDIV_A < 3 conditions

When the asynchronous prescaler value (PREDIV_A bits in the RTC_PRER register) is less than 3, the CALP bit cannot be set to 1. If CALP is already set to 1 and the PREDIV_A value is less than 3, CALP is ignored, meaning calibration is performed assuming CALP equals 0.

To perform calibration under PREDIV_A < 3 conditions, the synchronous prescaler value (PREDIV_S) should be reduced to allow 8 additional RTCCLK clock cycles per second, meaning 256 clock cycles are added every 32 seconds. Thus, using only the CALM bits, it is possible to effectively add 255 to 256 clock pulses every 32 seconds (corresponding to a calibration range of 243.3 ppm to 244.1 ppm).

At the nominal RTCCLK frequency of 32768 Hz, when PREDIV_A equals 1 (division factor of 2), PREDIV_S should be set to 16379 instead of 16383 (4 less). The only other relevant case is when PREDIV_A equals 0, PREDIV_S should be set to 32759 instead of 32767 (8 less).

If PREDIV_S is reduced in this way, the effective frequency of the calibrated input clock is calculated using the following formula:

$$FCAL = FRTCCLK \times [1 + (256 - CALM) / (220 + CALM - 256)]$$

In this case, if RTCCLK is exactly 32768.00 Hz, when CALM[7:0] equals 0x100 (the median of the CALM range), it indicates the settings are correct.

28.3.13.2. RTC Calibration Verification

Measure the exact frequency of RTCCLK and calculate the correct CALM and CALP values to ensure RTC accuracy. Additionally, an optional 1 Hz output is provided for applications to measure and verify RTC accuracy.

If the exact frequency of the RTC is measured over a limited interval, it may result in a measurement error of up to 2 RTCCLK clock cycles during the measurement, depending on the alignment of the digital calibration cycle with the measurement period.

However, this measurement error can be eliminated if the measurement period is the same length as the calibration cycle period. In this case, the only error observed is the error due to the resolution of the digital calibration.

- By default, the calibration cycle is 32 seconds.

In this mode, measuring the accuracy of the 1 Hz output over the full 32 seconds ensures a measurement error within 0.477 ppm (0.5 RTCCLK cycles over 32 seconds, limited by calibration resolution).

- Set the CALW16 bit in the RTC_CALR register to 1 to force a 16-second calibration cycle.

In this case, RTC accuracy can be measured within 16 seconds, with a maximum error of 0.954 ppm (0.5 RTCCLK cycles over 16 seconds). However, due to reduced calibration resolution, the long-term RTC accuracy will also drop to 0.954 ppm: when CALW16 is set to 1, the CALM[0] bit will always remain 0.

- Set the CALW8 bit in the RTC_CALR register to 1 to force an 8-second calibration cycle.

In this case, RTC accuracy can be measured within 8 seconds, with a maximum error of 1.907 ppm (0.5 RTCCLK cycles over 8 seconds). The long-term RTC accuracy will also drop to 1.907 ppm: when CALW8 is set to 1, the CALM[1:0] bits will always remain 00.

28.3.13.3. Re-calibration on-the-fly

When RTC_ICSR/INITF=0, the calibration register (RTC_CALR) can be updated dynamically by following these steps:

1. Poll RTC_ICSR/RECALPF (recalibration pending flag).
2. If the flag is 0, you can write new values to RTC_CALR as needed. The RECALPF bit will then be automatically set to 1.
3. The new calibration settings will take effect within three ck_apre cycles after writing to RTC_CALR.

28.3.14. Timestamp function

Setting the TSE or ITSE bit in the RTC_CR register enables the timestamp.

When TSE is set to 1:

When a timestamp event is detected on the RTC_TS pin, the calendar is saved to the timestamp registers (RTC_TSSSR, RTC_TSTR, and RTC_TSDR).

When TAMPTS is set to 1:

When a tamper event is detected on the TAMP_INx pin, the calendar is saved to the timestamp registers (RTC_TSSSR, RTC_TSTR, and RTC_TSDR).

When ITSE is set to 1:

When an internal timestamp event is detected, the calendar is saved to the timestamp registers (RTC_TSSSR, RTC_TSTR, and RTC_TSDR). When a timestamp event occurs due to an internal or external event, the timestamp flag (TSF) in the RTC_SR register will be set to 1. For internal events, the ITSF flag in the RTC_SR register will also be set to 1.

An interrupt can be generated upon a timestamp event by setting the TSIE bit in the RTC_CR register.

If a new timestamp event is detected while the timestamp flag (TSF) is already set to 1, the timestamp overflow flag (TSOVF) will be set to 1, and the timestamp registers (RTC_TSTR and RTC_TSDR) will retain the result of the previous event.

Note: Due to the synchronization process, TSF will be set to 1 within up to four ck_apre cycles after the timestamp event.

There is no delay in the generation of TSOVF. This means that if two timestamp events occur consecutively, TSOVF may be "1" while TSF is "0". Therefore, it is recommended to poll TSOVF only after detecting that TSF is "1".

Note: If a timestamp event occurs immediately after the TSF bit is cleared, both the TSF and TSOVF bits will be set to 1. To avoid masking a timestamp event occurring at the same moment, the application must not write 0 into TSF bit unless it has already read it to 1.

Optionally, a tamper event can cause a timestamp to be recorded. Refer to the description of the TAMPTS control bit in the RTC control register (RTC_CR).

28.3.15. Calibration clock output

When the COE bit is set to 1 in the RTC_CR register, a reference clock is provided on the CALIB device output.

If the COSEL bit in the RTC_CR register is set to 0 and PREDIV_A = 0x7F, the CALIB frequency is $f_{RTCCLK}/64$.

This corresponds to a calibration output of 512 Hz when the RTCCLK frequency is 32.768 kHz. The CALIB duty cycle is irregular: there is slight jitter on the falling edge. It is therefore recommended to use rising edges.

If COSEL is set to 1 and "PREDIV_S+1" is a non-zero integer multiple of 256 (e.g., PREDIV_S[7:0] = 0xFF), the CALIB frequency is $f_{RTCCLK}/(256 * (PREDIV_A+1))$. This is equivalent to a 1 Hz calibration output when the RTCCLK frequency is 32.768 kHz, with the default prescaler values (PREDIV_A = 0x7F, PREDIV_S = 0xFF).

Note: When the CALIB output is selected, the RTC_OUT pin is automatically configured.

When the COSEL bit is cleared, the CALIB output is the 6th stage output of the asynchronous prescaler.

When the COSEL bit is set, the CALIB output is the 8th stage output of the synchronous prescaler.

28.3.16. Tamper and alarm output

The OSEL[1:0] control bits in the RTC_CR register are used to activate the alarm output TAMPALRM and select the output function. These functions can reflect the content of the corresponding flags in the RTC_SR register.

When the TAMPOE control bit in RTC_CR is set to 1, all external and internal tamper flags are logically ORed and then connected to the TAMPALRM output. If OSEL = 00, the TAMPALRM output only reflects the tamper flags. If OSEL ≠ 00, the signal on TAMPALRM provides the tamper flags and alarm A, B, or wake-up flags.

The polarity of the TAMPALRM output is determined by the POL control bit in RTC_CR, so that when POL is set to 1, the inverted value of the selected flag is output.

28.3.16.1. TAMPALRM output

The TAMPALRM pin can be configured as an open-drain or push-pull output using the TAMPALRM_TYPE control bit in the RTC_CR register. An internal pull-up can be applied in output mode using TAMPALRM_PU in RTC_CR.

Note: When the TAMPALRM output is enabled, it takes priority over CALIB on RTC_OUT.

When the TAMPALRM output is selected, the RTC_OUT pin is automatically configured. If TAMPALRM is configured as open-drain in the RTC, the RTC_OUT GPIO must be configured as an input.

28.4. RTC low-power modes

The following table shows the RTC operation in low-power modes:

Table 28-3 RTC low-power modes

Mode	Description
Sleep	No impact. RTC interrupts can wake up the sleep mode.
Stop	When the RTC clock source is selected as LSI or LSE, the RTC remains operational. RTC interrupts can wake up the stop mode.
Standby	When the RTC clock source is selected as LSI or LSE, the RTC remains operational. RTC interrupts can wake up standby mode.

The table below shows RTC pin functions in each mode:

Table 28-4 RTC Pin Functions

Function	Functional in all low-power modes except standby mode	Functional in standby mode
RTC_TS	Yes	Yes
RTC_REFIN	Yes	None
RTC_OUT	Yes	Yes

28.5. RTC interrupts

The table below shows RTC interrupt requests:

Table 28-5 RTC interrupt requests

Interrupt event	Event flag (1)	Enable control bit (2)	Interrupt clear method	Exit sleep mode	Exit stop and standby modes
Alarm A	ALRAF	ALRAIE	Write 1 to CALRAF	Yes	Yes (3)
Alarm B	ALRBF	ALRBIE	Write 1 to CALRBF	Yes	Yes (3)
Timestamp	TSF	TSIE	Write 1 in CTSF	Yes	Yes (3)
Wake-up timer interrupt	WUTF	WUTIE	Write 1 in CWUTF	Yes	Yes (3)

1. The event flags are located in the RTC_SR register.
2. The interrupt mask flags (generated by the logical AND operation of the event flag bits and enable control bits) are located in the RTC_MISR register.
3. Wake-up from Stop and Standby modes is only possible when the RTC clock source is LSE or LSI.

28.6. RTC registers

28.6.1. RTC Time Register (RTC_TR)

RTC_TR is the calendar time shadow register. This register must be written in initialization mode only.

This register is write protected.

Offset address: 0x00

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): 0x0000 0000 (when BYPSHAD=0, unaffected when BYPSHAD=1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PM	HT[1:0]		HU[3:0]			
									RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	MNT[2:0]			MNU[3:0]				Res.	ST[2:0]			SU[3:0]			
	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	Reserved
22	PM	RW	0	AM/PM notation. 0: AM or 24-hour format 1: PM
21:20	HT[1:0]	RW	0	Hour tens in BCD format
19:16	HU[3:0]	RW	0	Hour units in BCD format
15	Reserved	-	-	Reserved
14:12	MNT[2:0]	RW	0	Minute tens in BCD format
11:8	MNU[3:0]	RW	0	Minute units in BCD format
7	Reserved	-	-	Reserved
6:4	ST[2:0]	RW	0	Second tens in BCD format
3:0	SU[3:0]	RW	0	Second units in BCD format

28.6.2. RTC Date Register (RTC_DR)

The RTC_DR is the calendar date shadow register. This register must be written in initialization mode only.

This register is write protected.

Offset address: 0x04

Reset value (RTC domain reset): 0x0000 2101

Reset value (System reset): 0x00002101 (When BYPSHAD=0, unaffected when BYPSHAD=1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	YT[3:0]				YU[3:0]			
								RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDU[2:0]			MT	MU[3:0]				Res.	Res.	DT[1:0]		DU[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23: 20	YT[3:0]	RW	0	Tens digit of year (BCD format)
19:16	YU[3:0]	RW	0	Units digit of year (BCD format)
15:13	WDU[2:0]	RW	0x1	Day of week 000: Disabled

				001: Monday ... 111: Sunday
12	MT	RW	0	Tens digit of month (BCD format)
11:8	MU[3:0]	RW	0x1	Units digit of month (BCD format)
7:6	Reserved	-	-	Reserved
5:4	DT[1:0]	RW	0	Tens digit of date (BCD format)
3:0	DU[3:0]	RW	0x1	Units digit of date (BCD format)

28.6.3. RTC Subsecond Register (RTC_SSR)

Offset address: 0x08

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): 0x0000 0000 (when BYPSHAD=0, unaffected when BYPSHAD=1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	SS[15:0]	R	0	Subsecond value. This register is the value of the synchronous prescaler counter. Subsecond value = (PREDIV_S - SS) / (PREDIV_S + 1) Note: SS can be greater than PREDIV_S only after a shift operation is performed. In this case, the correct time/date is one second slower than indicated by RTC_TR/RTC_DR

28.6.4. RTC Initialization Control and Status Register (RTC_ICSR)

This register is write protected.

Address offset: 0x0C

Reset value (RTC domain reset): 0x0000 0007

Reset value (system reset): Unaffected (except for INIT, INITF, and RSF bits, which are cleared)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RECALPF
															R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INIT	INITF	RSF	INIT S	SHPF	WUTWF	ALRBWF	ALRAWF
								RW	R	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	RECALPF	R	0	Recalibration pending flag. When software performs a write operation to the RTC_CALR register, the RECALPF status flag is automatically set to 1, indicating that the RTC_CALR register is masked. This bit returns to 0 when the new calibration settings are applied.
15:8	Reserved	-	-	Reserved
7	INIT	RW	0	Initialization mode 0: Free-running mode 1: Initialization mode, used for programming the time (RTC_TR) and date registers (RTC_DR) as well as the prescaler register (RTC_PRER). The counter stops counting, and when INIT is reset, the counter starts counting from the new value.
6	INITF	R	0	Initialization flag When this bit is set to 1, the RTC is in initialization state, allowing updates to the event, date, and prescaler registers. 0: Calendar register updates are not allowed

				1: Calendar register updates are allowed
5	RSF	RC_W0	0	Register synchronization flag This bit is set by hardware each time the values of the calendar registers are copied into the shadow registers (RTC_SSRx, RTC_TRx, and RTC_DRx). This bit is cleared by hardware in initialization mode, when a shift operation is pending (SHPF = 1), or in bypass shadow register mode (BYPHAD = 1). This bit can also be cleared by software. It is cleared either by software or by hardware in initialization mode. 0: Calendar shadow registers not synchronized 1: Calendar shadow registers synchronized
4	INITS	R	0	Initialization status flag This bit is set by hardware when the calendar year field is not 0 (RTC domain reset state). 0: Calendar not initialized 1: Calendar initialized
3	SHPF	R	0	Shift operation pending This flag is set by hardware as soon as a shift operation is initiated by writing to the RTC_SHIFTR register. It is cleared by hardware when the corresponding shift operation has been executed. Writing to the SHPF bit has no effect. 0: No shift operation pending 1: A shift operation is pending
2	WUTWF	R	1	Wakeup timer write flag This bit is set by hardware when the WUT value can be modified after the WUTE bit in the RTC_CR register is cleared. After the WUTE bit in the RTC_CR register is set, hardware clears this bit after 3 RTCCLK cycles. 0: Wakeup timer configuration update not allowed 1: Wakeup timer configuration update allowed
1	ALRBWF	R	1	Alarm B write flag This bit is set by hardware when the Alarm B value can be modified after the ALRBE bit in the RTC_CR register is cleared. After setting the ALRBE bit in the RTC_CR register, the hardware clears this bit after 3 RTCCLK cycles. It is cleared by hardware in initialization mode. 0: Alarm B update not allowed 1: Alarm B update allowed
0	ALRAWF	R	1	Alarm A write flag. After setting the ALRAE bit to 0 in the RTC_CR register, this bit is set by hardware when the Alarm A value can be modified. After setting the ALRAE bit in the RTC_CR register, the hardware clears this bit after 3 RTCCLK cycles. It is cleared by hardware in initialization mode. 0: Alarm A update not allowed 1: Alarm A update allowed

28.6.5. RTC prescaler register (RTC_PRER)

This register must be written in initialization mode only. The initialization must be performed in two separate write accesses.

This register is write protected.

Offset address: 0x10

Reset value (RTC domain reset): 0x007F 00FF

Reset value (system reset): Unaffected

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

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	Reserved
22:16	PREDIV_A [22:16]	RW	0x7F	Asynchronous prescaler factor The formula for asynchronous divider factor is as follows: $ck\_apre\ frequency = RTCCLK\ frequency / (PREDIV\_A + 1)$
15	Reserved	-	-	Reserved
14:0	PREDIV_S [14:0]	RW	0x00FF	Synchronous prescaler factor The formula for synchronous divider factor is as follows: $ck\_spre\ frequency = ck\_apre\ frequency / (PREDIV\_S + 1)$ Note: During shift operation, the configured value of this register must be greater than 1.

28.6.6. RTC wakeup timer register (RTC_WUTR)

Write operations to this register are only allowed when the WUTWF bit in RTC_ICSR is set to 1.

This register is write protected.

Offset address: 0x14

Reset value (RTC domain reset): 0x0000 FFFF

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WUT[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	WUT[15:0]	RW	0xFFFF	Wakeup auto-reload value bits When the wakeup timer is enabled (WUTE set to 1), the WUTF flag is set to 1 once every (WUT[15:0] + 1) ck_wut cycles. The ck_wut period is selected via the WUCKSEL[2:0] bits in the RTC_CR register. When WUCKSEL[2] = 1, the wakeup timer becomes 17-bit, and WUCKSEL[1] is equivalent to WUT[16], which is the most significant bit to be reloaded into the timer. The first time WUTF is set to 1 occurs between WUT to (WUT+1) ck_wut cycles after WUTE is set to 1. Setting WUT[15:0] to 0x0000 is prohibited when WUCKSEL[2:0] = 011 (RTCCLK/2).

28.6.7. RTC control register (RTC_CR)

This register is write protected.

Address offset: 0x18

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	TAMPALRM_TYPE	TAMPALARM_PU	Res.	Res.	TAMP_OE	TAMP_TS	ITSE	COE	OSEL[1:0]		POL	COS EL	BKP	SUB1H	ADD1H
	RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSIE	WUTIE	ALRBIE	ALRAIE	TSIE	WUTE	ALRBE	ALRAE	Res.	FMT	BYP SHAD	REFCKON	TS EDGE	WUCKSEL[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30	TAMPALRM_TYPE	RW	0	TAMPALRM output type 0: TAMPALRM is push-pull output 1: TAMPALRM is open-drain output
29	TAMPALARM_PU	RW	0	TAMPALRM pull-up enable

				0: No pull-up applied on TAMPALRM output 1: Pull-up applied on TAMPALRM output
28:27	Reserved	-	-	Reserved
26	TAMPOE	RW	0	Tamper detection output to TAMPALRM enable 0: Tamper flag not connected to TAMPALRM 1: Tamper flag connected to TAMPALRM (combining signal from OSEL and polarity from POL).
25	TAMPTS	RW	0	Activate timestamp upon tamper detection event 0: Do not save RTC timestamp when tamper detection event occurs 1: Save RTC timestamp when tamper detection event occurs TAMPTS remains effective even when TSE = 0 in RTC_CR register. Timestamp flag is set after tamper flag, so if TAMPTS and TSIE are set to 1, it is recommended to disable tamper interrupt to avoid handling 2 interrupts.
24	ITSE	RW	0	Internal event timestamp enable 0: Disable internal timestamp event 1: Enable internal timestamp event
23	COE	RW	0	Calibration output enable This bit enables CALIB output 0: Disable calibration output 1: Enable calibration output
22:21	OSEL[1:0]	RW	0	Output selection These bits are used to select the flag to be connected to TAMPALRM output. 00: Disable output 01: Enable Alarm A output 10: Enable Alarm B output 11: Enable Wake-up output
20	POL	RW	0	Output polarity This bit is used to configure the polarity of the TAMPALRM output. 0: The pin is high when ALRAF/ALRBF/WUTF is set to 1 (depending on OSEL[1:0]) or when TAMPxF/ITAMPxF is set to 1 (when TAMPOE = 1). 1: The pin is low when ALRAF/ALRBF/WUTF is set to 1 (depending on OSEL[1:0]) or when TAMPxF/ITAMPxF is set to 1 (when TAMPOE = 1).
19	COSEL	RW	0	Calibration output selection When COE=1, this bit selects the signal output on CALIB. 0: Calibration output is 512 Hz 1: Calibration output is 1 Hz These frequencies are valid when RTCCLK is 32.768 kHz and the prescaler is at its default values (PREDIV_A = 127 and PREDIV_S = 255).
18	BKP	RW	0	Backup This bit can be written by the user to memorize whether the daylight saving time change has been performed or not.
17	SUB1H	W	0	Subtract 1 hour (winter time change) When this bit is set to 1 in modes other than initialization mode, the calendar time is decreased by 1 hour if the current hour is not 0. This bit is always read as 0. Setting this bit to 1 has no effect when the current hour is 0. 0: No effect 1: Decrease the current time by 1 hour. This can be used for winter time change.
16	ADD1H	W	0	Add 1 hour (summer time change) When this bit is set to 1 in modes other than initialization mode, the calendar time is increased by 1 hour. This bit is always read as 0. 0: No effect 1: Increase the current time by 1 hour. This can be used for summer time change

15	TSIE	RW	0	Timestamp interrupt enable 0: Timestamp interrupt disabled 1: Timestamp interrupt enabled
14	WUTIE	RW	0	Wakeup timer interrupt enable 0: Disable wakeup timer interrupt 1: Enable wakeup timer interrupt
13	ALRBIE	RW	0	Alarm B interrupt enable 0: Disable alarm B interrupt 1: Enable alarm B interrupt
12	ALRAIE	RW	0	Alarm A interrupt enable 0: Disable alarm A interrupt 1: Enable alarm A interrupt
11	TSE	RW	0	Timestamp enable 0: Disable timestamp 1: Enable timestamp
10	WUTE	RW	0	Wakeup timer enable 0: Disable wakeup timer 1: Enable wakeup timer Note: When wakeup timer is disabled, it must wait until WUTWF=1 before re-enabling
9	ALRBE	RW	0	Alarm B enable 0: Disable alarm B 1: Enable alarm B
8	ALRAE	RW	0	Alarm A enable 0: Disable Alarm A 1: Enable Alarm A
7	Reserved	-	-	Reserved
6	FMT	RW	0	Hour format 0: 24-hour/day format 1: AM/PM hour format
5	BYPSHAD	RW	0	Bypass shadow registers 0: Calendar values (when read from RTC_SSR, RTC_TR, and RTC_DR) are taken from the shadow registers, which are updated every two RTCCLK cycles. 1: Calendar values (when read from RTC_SSR, RTC_TR, and RTC_DR) are taken directly from the calendar counters. Note: If the APB1 clock frequency is lower than 7 times the RTCCLK frequency, BYPSHAD must be set to 1.
4	REFCKON	RW	0	RTC_REFIN reference clock detection enable (50 Hz or 60 Hz) 0: Disable RTC_REFIN detection 1: Enable RTC_REFIN detection Note: PREDIV_S must be 0x00FF.
3	TSEDGE	RW	0	Timestamp event active edge 0: RTC_TS input rising edge generates timestamp event 1: RTC_TS input falling edge generates timestamp event When TSEDGE is changed, TSE must be reset to avoid accidentally setting TSF to 1.
2:0	WUCKSEL[2:0]	RW	0	ck_wut wakeup clock selection 000: Select RTC/16 clock 001: Select RTC/8 clock 010: Select RTC/4 clock 011: Select RTC/2 clock 10x: Select ck_spre clock (typically 1 Hz) 11x: Select ck_spre clock (typically 1 Hz) and increment WUT counter value by 216

Note: Bits 6 and 4 of this register can only be written in initialization mode (RTC_ICSR/INITF = 1).
WUT = Wakeup unit counter value. When WUCKSEL[2:1] = 11, WUT = (0x0000 to 0xFFFF) + 0x10000 (incremented value).

Writing to bits 2 to 0 of this register is only allowed when RTC_CR WUTE bit = 0 and RTC_ICSR WUTWF bit = 1.

Note: Changing the hour during calendar hour increment is not supported, as it may mask the calendar hour increment.

Changes to ADD1H and SUB1H take effect in the next second.

28.6.8. RTC Write Protection Register (RTC_WPR)

Offset address: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	KEY[7:0]							
								W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	KEY[7:0]	W	0	Write protection key This byte is written by software. Reading this byte always returns 0x00.

28.6.9. RTC Calibration Register (RTC_CALR)

This register is write protected.

Address offset: 0x28

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CALP	CALW8	CALW16	Res.	Res.	Res.	Res.	CALM[8:0]								
RW	RW	RW					RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	CALP	RW	0	Increase the RTC frequency by 488.5 ppm 0: No RTCCLK pulse is added. 1: One RTCCLK pulse is effectively inserted every 211 pulses (increasing the frequency by 488.5 ppm). This feature should be used in combination with CALM, which reduces the calendar frequency at high resolution. If the input frequency is 32768 Hz, the number of added RTCCLK pulses in a 32-second window is calculated as: $(512 \times \text{CALP}) - \text{CALM}$.
14	CALW8	RW	0	Use 8-second calibration period When CALW8 is set to 1, the 8-second calibration period is selected. Note: When CALW8 = 1, CALM[1:0] will always remain 00.
13	CALW16	RW	0	Use 16-second calibration period When CALW16 is set to 1, the 16-second calibration period is selected. If CALW8 = 1, this bit must not be set to 1. Note: When CALW16 = 1, CALM[0] will always remain 0.
12:9	Reserved	-	-	Reserved
8:0	CALM[8:0]	RW	0	Negative calibration Reduce the calendar frequency by masking CALM pulses within 220 RTCCLK pulses (32 seconds if the input frequency is 32768 Hz). The resolution is 0.9537 ppm. To increase the calendar frequency, this function should be used in combination with CALP.

28.6.10. RTC shift control register (RTC_SHIFTR)

This register is write protected.

Address offset: 0x2C

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD1S	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
W															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SUBFS[14:0]														
	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31	ADD1S	W	0	Add one second 0: No effect 1: Add one second to the clock/calendar This bit is write-only and always reads as 0. When a shift operation is pending (SHPF = 1 in RTC_ICSR), writing to this bit has no effect. This function should be used with SUBFS (see description below) to effectively add a sub-second value to the clock in an atomic operation mechanism.
30:15	Reserved	-	-	Reserved
14:0	SUBFS [14:0]	W	0	Subtract sub-second value This bit is write-only and always reads as 0. When a shift operation is pending (SHPF = 1 in RTC_ICSR), writing to this bit has no effect. The value written to SUBFS will be added to the synchronous prescaler counter. Since this counter counts down, this operation effectively subtracts from (delays) the clock by: Delay (seconds) = SUBFS / (PREDIV_S + 1) When the ADD1S function is used with SUBFS, it effectively adds a sub-second value to the clock (advancing the clock), causing the clock to advance by the following time: Advance (seconds) = (1 - (SUBFS / (PREDIV_S + 1))). Note: Writing to SUBFS will clear RSF. The software will then wait until RSF = 1 to confirm that the shadow register has been updated with the shifted time.

28.6.11. RTC timestamp time register (RTC_TSTR)

The contents of this register are valid only when TSF in RTC_SR is set to 1. Clear this register when the TSF bit is reset.

Offset address: 0x30

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PM	HT[1:0]		HU[3:0]			
									R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	MNT[2:0]			MNU[3:0]				Res.	ST[2:0]			SU[3:0]			
	R	R	R	R	R	R	R		R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	Reserved
22	PM	R	0	AM/PM notation. 0: AM or 24-hour format 1: PM
21:20	HT[1:0]	R	0	Hour tens in BCD format
19:16	HU[3:0]	R	0	Hour units in BCD format
15	Reserved	-	-	Reserved
14:12	MNT[2:0]	R	0	Minute tens in BCD format
11:8	MNU[3:0]	R	0	Minute units in BCD format
7	Reserved	-	-	Reserved
6:4	ST[2:0]	R	0	Second tens in BCD format
3:0	SU[3:0]	R	0	Second units in BCD format

28.6.12. RTC timestamp date register (RTC_TSDR)

The contents of this register are valid only when TSF in RTC_SR is set to 1. Clear this register when the TSF bit is reset.

Offset Address: 0x34

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDU[2:0]			MT	MU[3:0]				Res.	Res.	DT[1:0]		DU[3:0]			
R	R	R	R	R	R	R	R			R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:13	WDU[2:0]	R	0x0	Day of week 000: Disabled 001: Monday ... 111: Sunday
12	MT	R	0	Tens digit of month (BCD format)
11:8	MU[3:0]	R	0x0	Units digit of month (BCD format)
7:6	Reserved	-	-	Reserved
5:4	DT[1:0]	R	0	Date tens digit (BCD format)
3:0	DU[3:0]	R	0x0	Units digit of date (BCD format)

28.6.13. RTC timestamp sub-second register (RTC_TSSSR)

The contents of this register are valid only when TSF in RTC_SR is set to 1. Clear this register when the TSF bit is reset.

Offset Address: 0x38

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	SS[15:0]	R	0	Subsecond value. When a timestamp event occurs, this register holds the value of the synchronous prescaler counter.

28.6.14. RTC alarm A register (RTC_ALRMAR)

This register can only be written when ALRAWF in RTC_ICSR is set to 1 or in initialization mode.

This register is write protected.

Offset address: 0x40

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSK4	WDSEL	DT[1:0]		DU[3:0]				MSK3	PM	HT[1:0]		HU[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	MSK4	RW	0	Alarm A date mask 0: Alarm A is set if date/day matches 1: Date/day is ignored in Alarm A comparison
30	WDSEL	RW	0	Day of week selection 0: DU[3:0] represents the date units digit 1: DU[3:0] represents the day of week DT[1:0] are don't care bits
29:28	DT[1:0]	RW	0	Tens digit of date (BCD format)
27:24	DU[3:0]	RW	0	Units digit of date (BCD format)
23	MSK3	RW	0	Alarm A hour mask 0: Set Alarm A to 1 if the hour matches 1: Hour is ignored in Alarm A comparison
22	PM	RW	0	AM/PM notation. 0: AM or 24-hour format 1: PM

21:20	HT[1:0]	RW	0	Hour tens in BCD format
19:16	HU[3:0]	RW	0	Hour units in BCD format
15	MSK2	RW	0	Alarm A minute mask 0: Set Alarm A to 1 if the minute matches 1: Minute is ignored in Alarm A comparison
14:12	MNT[2:0]	RW	0	Minute tens in BCD format
11:8	MNU[3:0]	RW	0	Minute units in BCD format
7	MSK1	RW	0	Alarm A second mask 0: Set Alarm A to 1 if the second matches 1: Second is ignored in Alarm A comparison
6:4	ST[2:0]	RW	0	Second tens in BCD format
3:0	SU[3:0]	RW	0	Second units in BCD format

28.6.15. RTC Alarm A Subsecond Register (RTC_ALRMASRR)

This register can only be written when ALRAWF in RTC_ICSR is set to 1 or in initialization mode.

This register is write protected.

Offset Address: 0x44

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

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

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:24	MASKSS[3:0]	RW	0	Mask the most significant bits starting from this bit 0: Do not compare subseconds for Alarm A Set alarm when the second field increments (assuming all other fields match) 1: SS[14:1] are ignored in Alarm A comparison Only compare SS[0]. 2: In Alarm A comparison, SS[14:2] are don't-care bits. Only compare SS[1:0]. 3: In Alarm A comparison, SS[14:3] are don't-care bits. Only compare SS[2:0]. ... 12: In Alarm A comparison, SS[14:12] are don't-care bits. Compare SS[11:0]. 13: In Alarm A comparison, SS[14:13] are don't-care bits. Compare SS[12:0]. 14: In Alarm A comparison, SS[14] is a don't-care bit. Compare SS[13:0]. 15: All 15 SS bits are compared, and all must match to activate the alarm. The overflow bit (bit 15) of the synchronous counter is never compared. This bit is non-zero only after a shift operation is performed. Note: The overflow bit (bit 15) of the synchronous counter is never compared. This bit is non-zero only after a shift operation is performed.
23:15	Reserved	-	-	Reserved
14:0	SS[14:0]	RW	0	Subsecond value. This value is compared with the content of the synchronous prescaler counter to determine whether to activate Alarm A. Only compare bits 0 to MASKSS-1.

28.6.16. RTC Alarm B Register (RTC_ALRMBR)

This register can only be written when ALRBWF in RTC_ICSR is set to 1 or in initialization mode.

This register is write protected.

Offset Address: 0x48

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSK4	WDSEL	DT[1:0]		DU[3:0]				MSK3	PM	HT[1:0]		HU[3:0]			

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0
MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]				
RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	MSK4	RW	0	Alarm B date mask 0: Alarm B is set to 1 if date/day matches 1: Date/day is irrelevant in Alarm B comparison
30	WDSEL	RW	0	Day of week selection 0: DU[3:0] represents the date units digit 1: DU[3:0] represents the day of week DT[1:0] are don't care bits
29:28	DT[1:0]	RW	0	Tens digit of date (BCD format)
27:24	DU[3:0]	RW	0	Units digit of date (BCD format)
23	MSK3	RW	0	Alarm B hour mask 0: Alarm B is set to 1 if hour matches 1: Hour is irrelevant in Alarm B comparison
22	PM	RW	0	AM/PM notation. 0: AM or 24-hour format 1: PM
21:20	HT[1:0]	RW	0	Hour tens in BCD format
19:16	HU[3:0]	RW	0	Hour units in BCD format
15	MSK2	RW	0	Alarm B minute mask 0: Alarm B is set to 1 if minute matches 1: Minute is irrelevant in Alarm B comparison
14:12	MNT[2:0]	RW	0	Minute tens in BCD format
11:8	MNU[3:0]	RW	0	Minute units in BCD format
7	MSK1	RW	0	Alarm B second mask 0: Alarm B is set to 1 if second matches 1: Second is irrelevant in Alarm B comparison
6:4	ST[2:0]	RW	0	Second tens in BCD format
3:0	SU[3:0]	RW	0	Second units in BCD format

28.6.17. RTC Alarm B Sub-Second Register (RTC_ALRMBSSR)

This register can only be written when ALBAWF in RTC_ICSR is set to 1 or in initialization mode.

This register is write protected.

Offset address: 0x4C

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

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

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:24	MASKSS[3:0]	RW	0	Mask the most significant bits starting from this bit 0: Do not compare sub-seconds for Alarm B. Set alarm when the second field increments (assuming all other fields match) 1: In Alarm B comparison, SS[14:1] are don't-care bits. Only compare SS[0]. 2: In Alarm B comparison, SS[14:2] are don't-care bits. Only compare SS[1:0]. 3: In Alarm B comparison, SS[14:3] are don't-care bits. Only compare SS[2:0]. ... 12: In Alarm B comparison, SS[14:12] are don't-care bits. Compare SS[11:0]. 13: In Alarm B comparison, SS[14:13] are don't-care bits. Compare SS[12:0]. 14: In Alarm B comparison, SS[14] is a don't-care bit. Compare SS[13:0]. 15: All 15 SS bits are compared, and all must match to

				activate the alarm. The overflow bit (bit 15) of the synchronous counter is never compared. This bit is non-zero only after a shift operation is performed. Note: The overflow bit (bit 15) of the synchronous counter is never compared. This bit is non-zero only after a shift operation is performed.
23:15	Reserved	-	-	Reserved
14:0	SS[14:0]	RW	0	Sub-second value. This value is compared with the content of the synchronous prescaler counter to determine whether to activate alarm B. Only compare bits 0 to MASKSS-1.

28.6.18. RTC status register (RTC_SR)

This register is write protected.

Address offset: 0x50

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITSF	TSOVF	TSF	WUTF	ALRBF	ALRAF
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	ITSF	R	0	Internal timestamp flag This flag is set by hardware when a timestamp on the internal event occurs.
4	TSOVF	R	0	Timestamp overflow flag. This flag is set by hardware when a timestamp interrupt occurs while TSF is already set to 1. It is recommended to check and clear the TSOVF bit only after the TSF bit has been cleared. Otherwise, if a timestamp event occurs just before clearing the TSF bit, the overflow event might be missed.
3	TSF	R	0	Timestamp flag This flag is set by hardware when a timestamp event occurs. If the ITSF flag is already set, TSF must be cleared together with ITSF.
2	WUTF	R	0	Wakeup timer flag. When the wakeup auto-reload counter counts down to 0, this flag is set to 1 by hardware. Software must clear this flag at least 1.5 RTCCLK cycles before WUTF is set to 1 again.
1	ALRBF	R	0	Alarm B flag This flag is set to 1 by hardware when the time/date registers (RTC_TR and RTC_DR) match the Alarm B register (RTC_ALRMBR).
0	ALRAF	R	0	Alarm A flag. This flag is set to 1 by hardware when the time/date registers (RTC_TR and RTC_DR) match the Alarm A register (RTC_ALRMAR).

28.6.19. RTC Masked Interrupt Status Register (RTC_MISR)

This register is write protected.

Address offset: 0x54

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

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

Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ITSM F	TSOVM F	TSM F	WUTM F	ALRBM F	ALRAM F
											R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	ITSMF	R	0	Internal timestamp mask flag Set by hardware when an internal timestamp event occurs and ITSE is 1, triggering an interrupt.
4	TSOVMF	R	0	Timestamp overflow mask flag. This flag is set to 1 by hardware when a timestamp event occurs while TSMF is already set to 1. It is recommended to check and clear the TSOVF bit only after the TSF bit has been cleared. Otherwise, if a timestamp event occurs just before clearing the TSF bit, the overflow event might be missed.
3	TSMF	R	0	Timestamp Mask Flag When a timestamp event occurs and TSE is 1, the hardware sets this flag to 1. If the ITSMF flag is already set to 1, TSMF must be cleared together with ITSMF.
2	WUTMF	R	0	Wakeup Timer Mask Flag When the wakeup auto-reload counter counts down to 0 and WUTIE is 1, the hardware sets this flag to 1. Software must clear this flag at least 1.5 RTCCLK cycles before WUTF is set to 1 again.
1	ALRBMF	R	0	Alarm B Mask Flag When the time/date registers (RTC_TR and RTC_DR) match the Alarm B register (RTC_ALRMBR) and ALRBIE is 1, the hardware sets this flag to 1.
0	ALRAMF	R	0	Alarm A Mask Flag When the time/date registers (RTC_TR and RTC_DR) match the Alarm A register (RTC_ALRMAR) and ALRAIE is 1, the hardware sets this flag to 1.

28.6.20. RTC Status Clear Register (RTC_SCR)

Address offset: 0x5C

Reset value (RTC domain reset): 0x0000 0000

Reset value (system reset): Unaffected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CITSF	CTSOVF	CTSF	CWUTF	CALRBF	CALRAF
										W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	CITSF	W	0	Clear Internal Timestamp Flag Clear the ITSF bit by writing 1.
4	CTSOVF	W	0	Clear the timestamp overflow flag. Clear the TSOVF bit by writing 1. It is recommended to check and clear the TSOVF bit only after the TSF bit has been cleared. Otherwise, if a timestamp event occurs just before clearing the TSF bit, the overflow event might be missed.
3	CTSF	W	0	Clear the timestamp flag. Clear the TSF bit by writing 1. If the ITSF flag is set to 1, both TSF and ITSF must be cleared by setting CRSF and CITSF to 1.
2	CWUTF	W	0	Clear the wakeup timer flag. Clear the WUTF bit by writing 1.

1	CALRBF	W	0	Clear the alarm B flag. Clear the ALRBF bit by writing 1.
0	CALRAF	W	0	Clear the alarm A flag. Clear the ALRAF bit by writing 1.

Puya Confidential

29. Tamper and backup registers (TAMP&BKP)

29.1. TAMP introduction

Backup registers are 8 32-bit registers used to store 32 bytes of user application data. They can be used to store sensitive data as their content is protected by an tamper detection circuit. One tamper pin and one internal tamper can be used for tamper detection. The external tamper pins can be configured for edge detection, or level detection with or without filtering.

When the system is woken up from standby mode, the backup registers are not erased.

Backup registers are reset by backup domain power-on reset and RTC domain soft reset (BPOR+BDRST), and their contents are erased upon detection of a tamper event.

29.2. TAMP main features

- Supports 32-byte data backup registers
- Status/control register for managing anti-intrusion detection and having interrupt function
- 1 external intrusion detection event
- 1 internal intrusion event
- Any intrusion detection can generate an RTC timestamp event
- Any intrusion detection can erase the backup registers
- Backup registers are protected by FLASH RDP (Read Protection). When RDP level changes from 1 to 0, all data in backup registers will be erased

29.3. TAMP functional description

29.3.1. TAMP block diagram

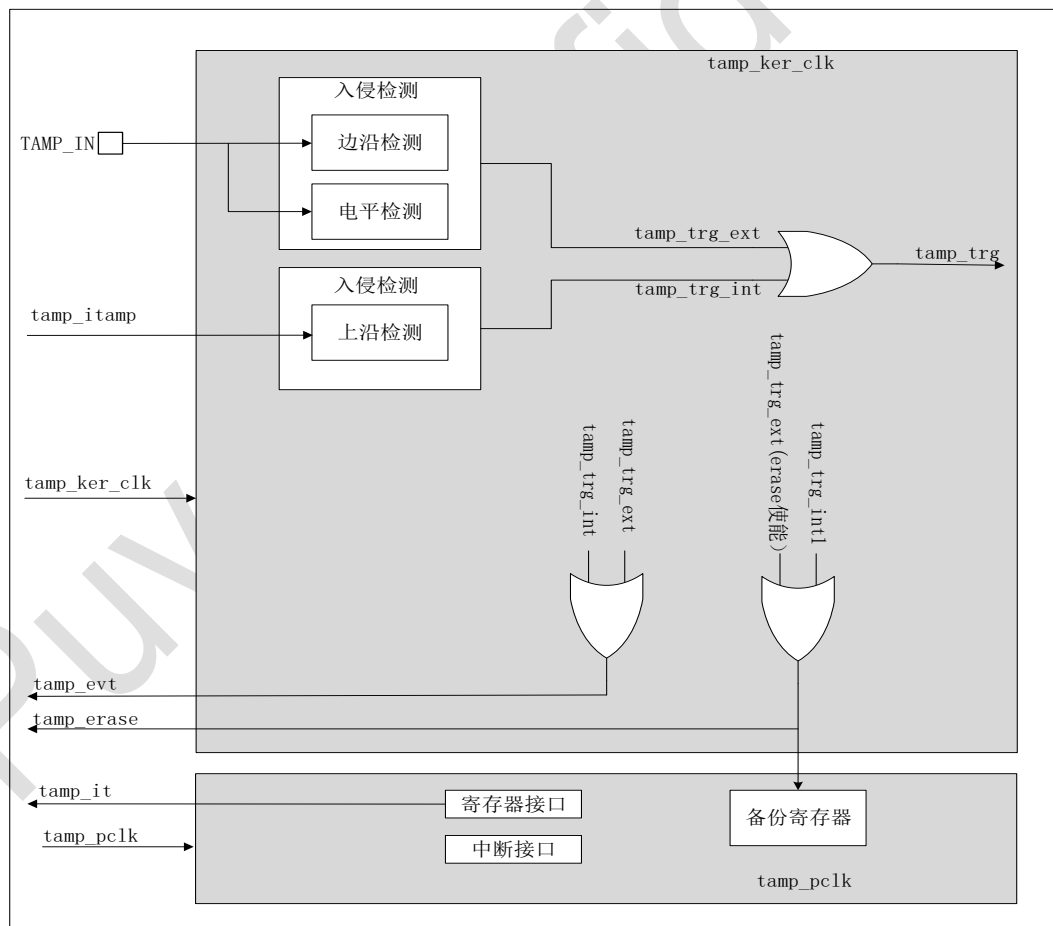


Figure 29-1 TAMP block diagram

29.3.2. TAMP pin and interface description

Table 29-1 TAMP signal description

Names	Signal type	Description
TAMP_IN	I	TAMP external trigger pin (PA8)
tamp_itamp	I	TAMP internal trigger pin
tamp_ker_clk	I	TAMP core clock, connected to rtc_ker_ck (RTCCLK)
tamp_pclk	I	TAMP APB clock
tamp_evt	Output	Intrusion event detected, used to generate RTC timestamp
tamp_erase	Output	Intrusion event detected, used to erase confidential information: backup registers
tamp_it	Output	TAMP interrupts
tamp_trg	Output	TAMP trigger output, from external trigger source "OR" internal trigger source

29.3.3. TAMP register write protection

After a system reset, access to TAMP registers is prohibited to prevent potential unintended write operations. To enable access to TAMP registers, perform the following operations:

- Enable the clock for the power management module by setting the PWREN bit in the RCC_APBENR1 register
- Set the DBP bit in the power control register PWR_CR1 to enable access to TAMP registers.

29.3.4. Tamper detection

The tamper detection can be configured for the following purposes:

- Erase backup registers (default configuration)
- Generate an interrupt, capable of waking up from Stop and Standby modes
- Generate hardware trigger for LPTIM

29.3.4.1. TAMP backup registers

Backup registers (TAMP_BKPxR) are not reset by a system reset or when the device wakes up from Standby mode.

Backup registers are reset upon a tamper detection event, except when the TAMPxNOER bit is set to 1 or the TAMPxMSK bit in the TAMP_CR2 register is set to 1.

Note: Backup registers are also erased when the Flash read protection level is changed from Level 1 to Level 0.

29.3.4.2. Tamper detection initialization

Each input can be enabled by setting the corresponding TAMPxE bit in the TAMP_CR1 register.

Each TAMP_INx tamper detection input is associated with the TAMPxF flag in the TAMP_SR register.

When TAMPxMSK is cleared:

The TAMPxF flag will be enabled after a tamper event occurs on the pin, with the following delay:

- When TAMPFLT is not 0x0 (level detection with filtering), the delay is the number of RTCCLK cycles configured for TAMPFLT
- When TAMPFLT = 0x0 (edge detection), there is no delay

During this period, as long as TAMPxF is set to 1, new tamper events on the same pin cannot be detected until the flag is cleared by software.

When TAMPxMSK is set to 1:

During the aforementioned delay and for an additional 3 RTCCLK cycles, new tamper events on the same pin cannot be detected.

An interrupt can be generated upon a tamper detection event (when TAMPxF is set to 1) by setting the TAMPxIE bit in the TAMP_IER register. When the corresponding TAMPxMSK is set to 1, setting TAMPxIE to 1 is not allowed.

29.3.4.3. Setting TAMPxIE is not allowed when the corresponding TAMPxMSK is set.

LPTIM can use tamper event detection as a trigger input.

When clearing the TAMPxMSK bit in the TAMP_CR2 register, the TAMPxF flag must be cleared by software to detect new tamper events on the same pin.

When the TAMPxMSK bit is set to 1, the TAMPxF flag is masked and remains cleared in the TAMP_SR register. This configuration allows LPTIM to trigger automatically in Stop mode without requiring a system wakeup to clear TAMPxF. In this case, the backup registers are not cleared.

29.3.4.4. Timestamp on tamper event

When TAMPTS in RTC_CR is set to 1, any tamper event will cause a timestamp event to occur. In this case, the TSF or TSOVF bit in RTC_SR will be set to 1, just like a normal timestamp event. When TSF or TSOVF in RTC_SR is set to 1, the affected tamper flag TAMPxF in TAMP_SR will also be set to 1.

29.3.4.5. Edge detection on tamper inputs (passive mode)

If the TAMPFLT bit is set to 00, a tamper detection event will be generated on the TAMP_INx pin when a rising or falling edge appears on the corresponding TAMPxTRG bit.

Note: When using edge detection, it is recommended to check the tamper pin level via software (by reading the GPIO register) immediately after enabling tamper detection and before writing sensitive values to the backup registers, to ensure valid edges occur only after tamper event detection is enabled.

When TAMPFLT = 00 and TAMPxTRG = 0 (rising edge detection), if the tamper input is already high before enabling tamper detection, the tamper event can be detected by hardware.

After detecting and clearing a tamper event, TAMP_INx should be disabled before reprogramming the backup registers (TAMP_BKPxR), and then re-enabled (TAMPxEN set to 1). This prevents the application from writing to the backup registers while the TAMP_INx input value still indicates tamper detection. This is equivalent to level detection on the TAMP_INx input.

Note: Tamper detection remains active when the V_{CC} power is off. To avoid accidental reset of backup registers, the pin mapped to TAMPx should be externally connected to the correct level.

29.3.4.6. Level detection with filtering on tamper inputs (passive mode)

Level detection with filtering can be performed by setting TAMPFLT to a non-zero value. A tamper detection event is generated when the level specified by the TAMPxTRG bit is continuously sampled 2, 4, or 8 times (depending on the TAMPFLT value).

29.4. Low-power modes

Table 29-2 Impact of Low-Power Modes on TAMP

Mode	Description
Sleep	No effect. TAMP interrupt (when enabled) will exit sleep mode.
Stop	No impact on all functions except for level detection mode with filtering, which remains active only when the clock source is LSE or LSI. Tamper event will exit stop mode.
Standby	No impact on all functions except for level detection mode with filtering, which remains active only when the clock source is LSE or LSI. Tamper event will exit standby mode.

29.5. TAMP interrupts

The interrupt channel is set in the interrupt status register. The interrupt output is also activated.

Table 29-3 Interrupt Requests

Interrupt event	Event flag	Enable control bit	Interrupt Clear	Exiting Sleep mode	Exit stop and standby modes
External tamper x	TAMP_SR.TAMP1F	TAMP1IE	Write 1 to CTAMP1F	Yes	Note 1 applies
Internal tamper	TAMP_SR.ITAMP1F	ITAMP1IE	Write 1 to CITAMP1F	Yes	Note 1 applies

Note 1: In passive tamper mode with filtered level detection, wake-up from stop and standby modes is only possible when the TAMP clock source is LSE or LSI.

29.6. TAMP registers

29.6.1. TAMP control register 1 (TAMP_CR1)

Address offset: 0x00

RTC domain reset value: 0x0000_0000

System reset: not affected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITAMP1EN

															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1EN
															RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	ITAMP1EN	RW	0	Internal tamper 1 enable: RTC calendar overflow 0: Disable internal tamper 1 1: Enable internal tamper 1 A tamper event is generated when the RTC calendar reaches its maximum value. The calendar is then frozen and cannot overflow.
15:1	Reserved	-	-	Reserved
0	TAMP1EN	RW	0	Tamper detection enable for TAMP_IN1 0: Disable intrusion detection for TAMP_IN. 1: Enable intrusion detection for TAMP_IN.

29.6.2. TAMP control register 2 (TAMP_CR2)

Address offset: 0x04

RTC domain reset value: 0x0000_0000

System reset: not affected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1TR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1MSK
							RW								RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1NOER
															RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	TAMP1TRG	RW	0	Valid level for external intrusion 1 input (disabled operation mode) If TAMPFLT ≠ 00: 0: Intrusion 1 input held low triggers an intrusion detection event. 1: Intrusion 1 input held high triggers an intrusion detection event. If TAMPFLT = 00: 0: Rising edge and high level of intrusion 1 input trigger an intrusion detection event. 1: Falling edge and low level of intrusion 1 input trigger an intrusion detection event.
23:17	Reserved	-	-	Reserved
16	TAMP1MSK	RW	0	Intrusion 1 mask 0: Intrusion 1 event generates a trigger event, and TAMP1F must be cleared by software to allow the next intrusion event detection. 1: Intrusion 1 event generates a trigger event. Mask TAMP1F and clear it internally by hardware. The backup registers are not erased. When TAMP1MSK is set to 1, intrusion 1 interrupt must not be enabled.
15:1	Reserved	-	-	Reserved
0	TAMP1NOER	RW	0	TAMP_IN1 intrusion does not erase backup registers 0: Intrusion 1 event erases backup registers. 1: Intrusion 1 event does not erase backup registers.

29.6.3. TAMP filter control register (TAMP_FLTCR)

Address offset: 0x0C

RTC domain reset value: 0x0000_0000

System reset: not affected

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

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	Reserved
4:3	TAMPFLT[1:0]	RW	0	TAMP_IN filter count These bits determine the number of consecutive samples required at the specified level (TAMP1TRG) to activate an intrusion event. TAMPFLT is effective for each TAMP_IN input. 0x0: Activates a tamper event on the active edge transition of the TAMP_IN input (no internal pull-up on TAMP_IN input). 0x1: Activates a tamper event after 2 consecutive samples at the active level. 0x2: Activates a tamper event after 4 consecutive samples at the active level. 0x3: Activates a tamper event after 8 consecutive samples at the active level.
2:0	TAMPFREQ[2:0]	RW	0	Tamper sampling frequency These bits determine the sampling frequency for each TAMP_INx input. 0x0: RTCCLK/32768 (1 Hz when RTCCLK = 32768 Hz) 0x1: RTCCLK/16384 (2 Hz when RTCCLK = 32768 Hz) 0x2: RTCCLK/8192 (4 Hz when RTCCLK = 32768 Hz) 0x3: RTCCLK/4096 (8 Hz when RTCCLK = 32768 Hz) 0x4: RTCCLK/2048 (16 Hz when RTCCLK = 32768 Hz) 0x5: RTCCLK/1024 (32 Hz when RTCCLK = 32768 Hz) 0x6: RTCCLK/512 (64 Hz when RTCCLK = 32768 Hz) 0x7: RTCCLK/256 (128 Hz when RTCCLK = 32768 Hz)

29.6.4. Tamper Interrupt Enable Register (TAMP_IER)

Address offset: 0x2C

RTC domain reset value: 0x0000_0000

System reset: not affected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITAMP1IE
															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1IE
															RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	ITAMP1IE	RW	0	Internal tamper 1 interrupt enable 0: Disable tamper 1 interrupt. 1: Enable tamper 1 interrupt.
15:1	Reserved	-	-	Reserved
0	TAMP1IE	RW	0	External tamper 1 interrupt enable 0: Disable tamper 1 interrupt. 1: Enable tamper 1 interrupt.

29.6.5. Tamper Status Register (TAMP_SR)

Address offset: 0x30

RTC domain reset value: 0x0000_0000

System reset: not affected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITAMP1F
															R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1F
															R

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	ITAMP1F	R	0	RTC calendar overflow tamper detection flag 0: No RTC calendar overflow detected 1: RTC calendar overflow detected
15:1	Reserved	-	-	Reserved
0	TAMP1F	R	0	External tamper 1 detection flag 0: No external tamper 1 detected 1: External tamper 1 detected

29.6.6. Tamper Masked Interrupt Status Register (TAMP_MISR)

Address offset: 0x34

RTC domain reset value: 0x0000_0000

System reset: not affected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITAMP1MF
															R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1MF
															R

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	ITAMP1MF	R	0	RTC calendar overflow tamper detection interrupt 0: No RTC calendar overflow interrupt detected 1: RTC calendar overflow interrupt detected When an RTC calendar overflow tamper detection flag is generated and the corresponding interrupt is enabled, the hardware sets this register.
15:1	Reserved	-	-	Reserved
0	TAMP1MF	R	0	External tamper 1 detection interrupt 0: No external tamper 1 interrupt detected 1: External tamper 1 interrupt detected When an external tamper 1 detection flag is generated and the corresponding interrupt is enabled, the hardware sets this register.

29.6.7. TAMP status clear register (TAMP_SCR)

Address offset: 0x3C

RTC domain reset value: 0x0000_0000

System reset: not affected

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CITAMP1F
															W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CTAMP1F
															W

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	CITAMP1F	W	0	Clear RTC calendar overflow tamper detection flag in TAMP_SR
15:1	Reserved	-	-	Reserved
0	CTAMP1F	W	0	Clear external tamper 1 detection flag in TAMP_SR

29.6.8. TAMP backup x register (TAMP_BKPxR)

Address offset: 0x100+0x04*x, (x=0~7)

RTC domain reset value: 0x0000_0000

System reset: not affected

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

Bit	Name	R/W	Reset Value	Function
31:0	BKP[31:0]	RW	0	The application can write or read data to and from these registers. When V_{CC} is turned off, these registers are powered by V_{DDK}, so they are not reset during a system reset, and the register contents remain valid when the device operates in low-power mode. In the default configuration this register is reset on a tamper detection event. It is forced to reset value as long as there is at least one internal or external tamper flag being set. This register is also reset when read protection (RDP) is disabled.

30. Internal integrated circuit interface (I2C)

30.1. I2C introduction

The I2C (Inter-Integrated Circuit) bus interface connects communication between the microcontroller and the serial I2C bus. It provides multimaster capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing. It supports Standard-mode (Sm), Fast-mode (Fm) and Fast-mode plus (Fm+).

It is also compatible with SMBus (System Management Bus) and PMBus (Power Management Bus). For specific needs, DMA can be used to reduce the burden on the CPU.

The I2C interface can be configured to support wakeup from Stop and Sleep low-power modes (address match).

This product includes 2 I2Cs, and the functions of both I2Cs are identical.

30.2. I2C key features

- Slave and master modes
- Support different communication speeds
 - Standard mode (Sm): up to 100 kHz
 - Fast mode (Fm): up to 400 kHz
 - Fast Mode Plus (Fm+): up to 1 MHz
- As Master
 - Generate clock
 - Generate Start and Stop bits
- As Slave
 - Programmable I2C address detection
 - Dual-address capability that responds to two secondary addresses
 - Stop bit detection
- 7-bit/10-bit addressing mode
- General call
- Status flag
 - Transmit/receive mode flags
 - Byte transfer complete flag
 - I2C busy flag bit
- I2S error flag
 - Master arbitration loss
 - ACK failure after address/data transfer
 - Start/Stop bit error
 - Overload/Underload (Clock stretching disabled)
- Optional clock stretching function
- Single-byte buffer with DMA capability
- Software reset
- Analog noise filter function
- Configurable PEC (Packet Error Checking) generation and verification
 - PEC value can be transmitted as last byte in Tx mode
 - PEC error checking for last received byte
- Support SMBus
 - SCL low timeout delay of 25 ms
 - Master accumulated SCL low extension time of 10 ms
 - Slave accumulated SCL low extension time of 25 ms
 - Hardware PEC generation/verification with ACK control
 - Supports Address Resolution Protocol (ARP)
 - Timeout and idle condition detection
- Supports PMBus
- Supports wakeup from low-power mode on address match
 - Supports configurable timeout period

30.3. I2C functional description

30.3.1. I2C block diagram

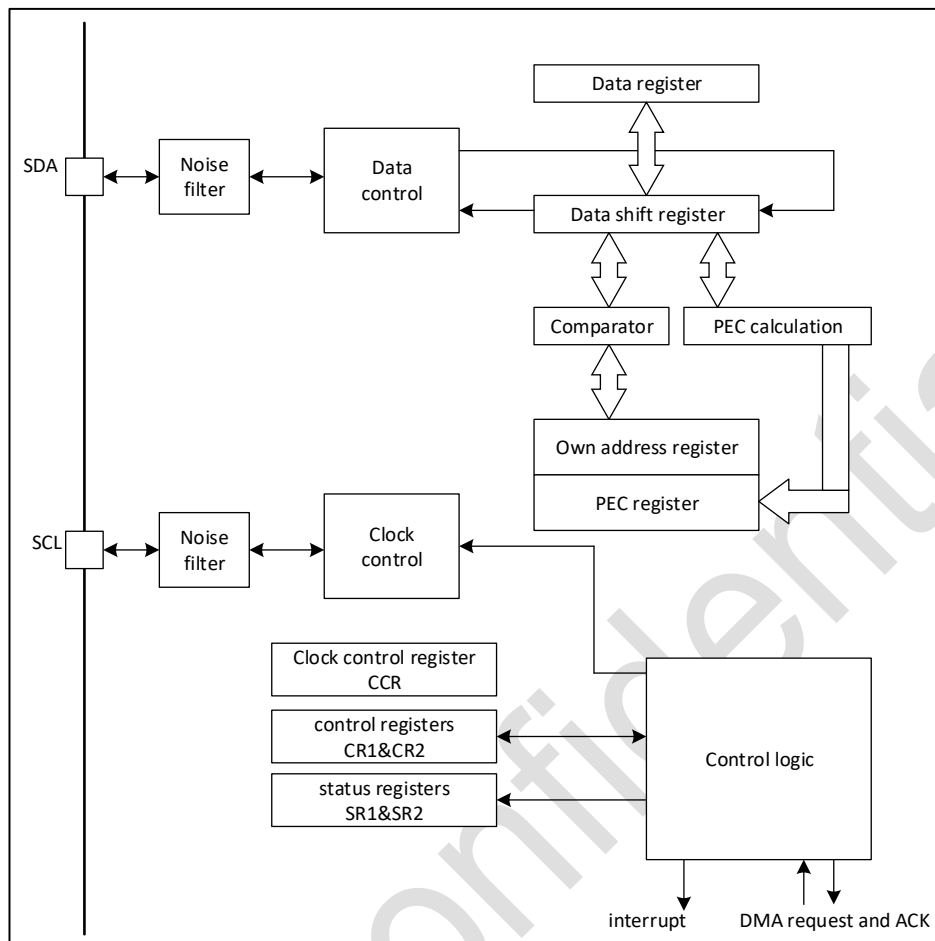


Figure 30-1 I2C block diagram

30.3.2. Mode selection

The interface can operate in one of the four following modes:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

Default operation in slave mode. The interface automatically switches from slave mode to master mode after generating a start condition.

30.3.2.1. Communication flow

In Master mode, the I2C interface initiates a data transfer and generates the clock signal. Serial data transmission always begins with a start bit and ends with a stop bit. Start and stop bits are generated under software control in master mode.

As a slave, the I2C interface can recognize its own address (7-bit or 10-bit) and the broadcast call address. The General Call address detection may be enabled or disabled by software.

Data and addresses are transmitted in 8-bit (byte) units, MSB first. One or two bytes following the start bit represent the address (1 byte for 7-bit address mode, 2 bytes for 10-bit address mode). The address is always transmitted in Master mode.

A 9th clock pulse follows the 8 clock cycles of a byte transfer, during which the receiver must send an acknowledged bit to the transmitter. Refer to the following figure. Refer to the figure below.

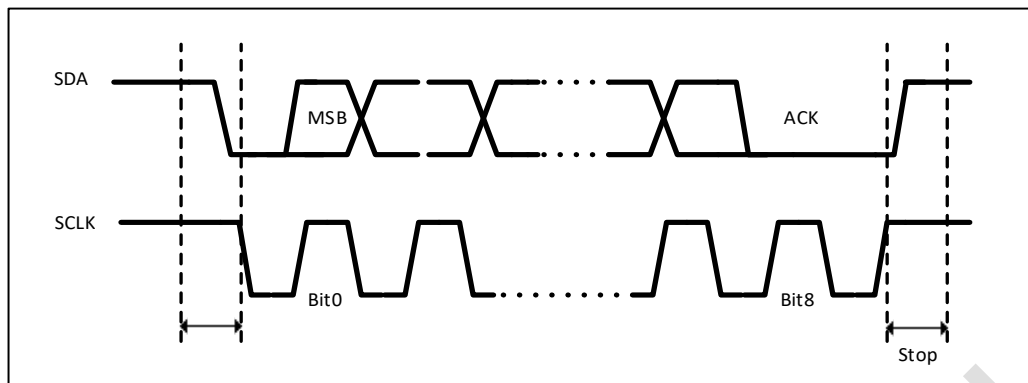


Figure 30-2 I2C bus protocol

Software can enable or disable the acknowledge bit (ACK). Software can also select the I2C interface address (7-bit/10-bit dual addressing and/or broadcast call address).

30.3.3. I2C initialization

30.3.3.1. Enabling and disabling the peripheral

The I2C clock module must first be enabled via the I2CxEN bit in the RCC_APBENR1 register, and then the I2C module is enabled by setting the PE bit in I2C_CR1 to 1.

30.3.3.2. Analog noise filter

Before enabling the I2C peripheral by setting the PE bit in I2C_CR1 register, the user must configure the noise filters, if needed. By default, an analog noise filter is present on the SDA and SCL inputs. This analog filter complies with the I2C specification, which requires filtering noise of a certain pulse width in Fast mode and Fast mode Plus (filter width is specified in the datasheet). Users can enable this analog filter by setting the SYSCFG_I0CFG.Px_EIIC bit to 1.

30.3.3.3. I2C timings

To meet the accurate data hold and setup times in both master and slave modes, I2C timing configuration is required. This is achieved by writing to the I2C_CCR and I2C_TRISE registers.

30.3.4. I2C slave mode

By default, the I2C interface operates in Slave mode. Switching from slave mode to master mode requires generating a start condition.

The peripheral input clock must be programmed in the I2C_CR2 register in order to generate correct timings. The peripheral input clock frequency must be at least:

2 MHz in Sm mode

4 MHz in Fm mode

In Fast mode Plus: 16 MHz

As soon as a start condition is detected, the address is received from the SDA line and sent to the shift register. Then it is compared with the address of the interface (OAR1) or the General Call address (if ENGCG = 1).

If the additional address is enabled (I2C_OAR2.ENDUAL=1), the address is compared with the chip's OAR2 address, and the address in the OAR2 register can be masked by OA2MSK[2:0], allowing up to 7 OA2 addresses to be masked. Therefore for OA2MSK configured from 1 to 6, only OA2[7:2], OA2[7:3], OA2[7:4], OA2[7:5], OA2[7:6] or OA2[7] are compared with the received address. As long as OA2MSK is not equal to 0, the OA2 address comparator excludes I2C reserved addresses (0000 XXX and 1111 XXX), which will not receive an acknowledgment. If OA2MSK=7, all received 7-bit addresses are acknowledged (except reserved addresses).

Header or address not matched:

The interface ignores it and waits for another Start condition.

Address matched:

The I2C interface generates the following timing and states:

- If I2C_CR1.ACK is set to '1' by software, an acknowledgment pulse is generated.
- Hardware sets the I2C_SR1.ADDR bit, and if the ITEVTEN bit is set, an interrupt is generated.
- In slave mode, the I2C_SR2.TRA bit indicates whether the current mode is receiver or transmitter.

30.3.4.1. Slave transmission

After receiving the address and clearing the I2C_SR1.ADDR bit, the slave sends data (bytes) from the I2C_DR register to the SDA via the internal shift register (if the LSB of the address byte is 1).

The slave pulls SCL low until the I2C_SR1.ADDR bit is cleared and the data to be sent is written to the I2C_DR register (refer to EV1, EV3).

When the acknowledge pulse is received: The TxE bit is set by hardware with an interrupt if the ITEVFEN and the ITBUFEN bits are set.

If TxE is set and some data were not written in the I2C_DR register before the end of the next data transmission, the BTF bit is set. The interface waits until BTF is cleared by a read to I2C_SR1 followed by a write to the I2C_DR register, stretching SCL low.

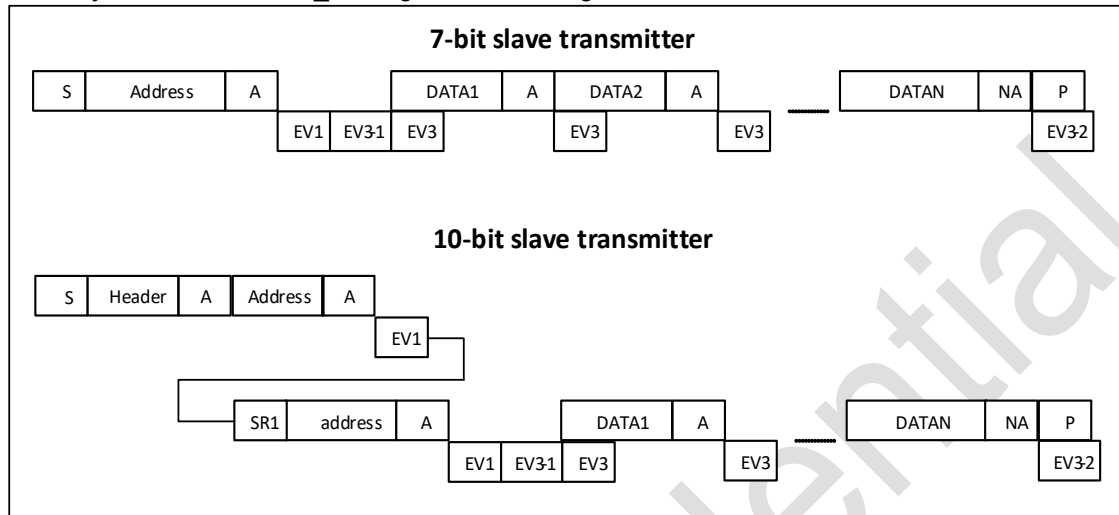


Figure 30-3 Slave transmission sequence diagram

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, NA= Non-acknowledge, EVx=Event(with interrupt if ITEVFEN=1)

EV1: ADDR=1, clear the ADDR bit by first reading the I2C_SR1 register and then reading the I2C_SR2 register

EV3-1: TxE=1, shift register empty, data register empty, write DATA1 to I2C_DR register

EV3: TxE=1, shift register not empty, data register empty, write (DATA2) to I2C_DR register to clear TxE

EV3-2: AF = 1; The software writes 0 to the AF bit to clear the bit

30.3.4.2. Slave receiver

Following the address reception and after clearing ADDR, the slave receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

- An acknowledge pulse if the ACK bit is set
- The RxNE bit is set by hardware. An interrupt is generated if the ITEVTEN and ITBUFEN bit is set.

If RxNE is set and the DR register is not read before the reception of new data is completed, the BTF bit is set. The slave will keep SCL low until BTF is cleared (by reading I2C_SR1 followed by reading I2C_DR register). (See figure below).

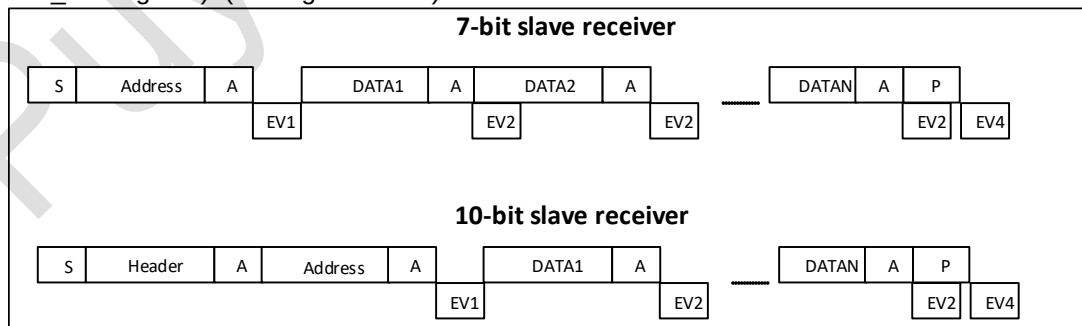


Figure 30-4 Slave reception sequence diagram

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledged, EVx= Event (with interrupt if ITEVFEN=1)

EV1: ADDR=1, cleared by reading I2C_SR1 followed by reading I2C_SR2

EV2: RxNE=1, read I2C_DR register to clear this bit

EV4: STOPF=1, cleared by reading I2C_SR1 register followed by writing to the I2C_CR1 register.

Note:

- 1) The EV1 event stretches SCL low until the end of the corresponding software sequence.
 - 2) The EV2 software sequence must be completed before the current byte transmission is finished.
- After checking the contents of the I2C_SR1 register, the user should perform a complete clear sequence operation for each flag found to be set. For example, the ADDR and STOPF flags require the following operation sequence:

-- If ADDR=1, first read I2C_SR1, then read I2C_SR2; if STOPF=1, first read I2C_SR1, then write to I2C_CR1.

The purpose is to make sure that both ADDR and STOPF flags are cleared if both are found set.

30.3.4.3. Disable slave communication

After transmitting the last data byte, the master generates a stop bit. When the slave detects this stop bit:

- The STOPF bit is set and generates an interrupt if the ITEVFEN bit is set.
- The STOPF bit is cleared by a read of the I2C_SR1 register followed by a write to the I2C_CR1 register. (See EV4 in figure above)

30.3.5. I2C master mode

In Master mode, the I2C interface initiates a data transfer and generates the clock signal. Serial data transmission always starts with a start bit and ends with a stop bit.

Master mode is selected as soon as the Start condition is generated on the bus with a START bit.

The following is the required sequence in master mode:

- Program the peripheral input clock in I2C_CR2 register in order to generate correct timings
- Configure clock control register
- Configure rise time register
- Program the I2C_CR1 register to enable the peripheral
- Set the START bit in the I2C_CR1 register to generate a Start condition

The peripheral input clock frequency must be at least:

- 2 MHz in Sm mode
- 4 MHz in Fm mode
- In Fast mode Plus: 16 MHz

30.3.5.1. SCL master clock generation

The I2C_CCR register generates the high and low levels of SCL using rising or falling edges. As a slave may stretch the SCL line, the peripheral checks the SCL input from the bus at the end of the time programmed in TRISE register after rising edge generation.

- If the SCL line is low, it means that a slave is stretching the bus, and the high level counter stops until the SCL line is detected high. This allows to guarantee the minimum HIGH period of the SCL clock parameter.
- If the SCL line is high, the high level counter keeps on counting.

In practice, even if the slave does not stretch SCK, the feedback loop from the generation of the SCL rising edge to its detection takes some time. This loop time is related to the SCL rise time (SCL VIH data detection), analog noise filtering of the SCL input path, and internal chip synchronization of SCL using the APB clock. By configuring the TRISE register, the SCL frequency remains stable regardless of the SCL rise time.

30.3.5.2. Start bit

When I2C_SR2.BUSY=0, setting I2C_CR1.START=1 will cause the I2C interface to generate a start condition and switch to master mode (I2C_SR2.MSL is set).

Note: Setting the START bit in master mode will generate a repeated start condition (ReStart) by hardware after the current byte transmission is completed.

Once the start bit is issued:

- The I2C_SR1.SB bit is set by hardware, and an interrupt will be generated if the ITEVTEN bit is set.
- Then the master waits for a read of the I2C_SR1 register followed by a write in the I2C_DR register with the Slave address. (EV5)

30.3.5.3. Slave address transmission

The slave address is sent to the SDA line through the internal shift register.

- In 10-bit address mode, sending a header sequence generates the following events:
 - The ADD10 bit is set by hardware, and an interrupt is generated if the ITEVTEN bit is set.
 - The software then reads the I2C_SR1 register, followed by the I2C_SR2 register.
 - The ADDR bit is set by hardware and an interrupt is generated if the ITEVFEN bit is set.
 - The software then reads the I2C_SR1 register, followed by the I2C_SR2 register.

- In 7-bit addressing mode, one address byte is sent.

As soon as the address byte is sent,

- The ADDR bit is set by hardware and an interrupt is generated if the ITEVFEN bit is set.
- The software then reads the I2C_SR1 register, followed by the I2C_SR2 register.
- Based on the least significant bit of the sent slave address, the software decides whether to enter transmitter mode or receiver mode.

- In 7-bit addressing mode,

- To enter transmitter mode, the master sets the least significant bit to '0' when sending the slave address.
- To enter receiver mode, the master sets the least significant bit to '1' when sending the slave address.

- In 10-bit addressing mode,

- To enter transmitter mode, the master first sends a header byte (11110xx0), followed by the slave address with the least significant bit set to 0 (the xx value in the header byte corresponds to the highest 2 bits of the 10-bit address).
- To enter receiver mode, the master first sends a header byte (11110xx0), followed by the slave address with the least significant bit set to 0. Then, a start condition is reissued, followed by the header byte (11110xx1) (the xx value in the header byte corresponds to the highest 2 bits of the 10-bit address).

- The TRA bit indicates whether the master is in receiver mode or transmitter mode.

30.3.5.4. Master transmitter

After sending the address and clearing the ADDR bit, the master sends the byte from the I2C_DR register to the SDA line through the internal shift register.

The master waits until the first data byte is written to the I2C_DR register (see EV8_1).

When the acknowledge pulse is received, the TxE bit is set by hardware and an interrupt is generated if the ITEVFEN and ITBUFEN bits are set.

If TxE is set and no new data byte is written to the I2C_DR register before the previous data transmission ends, BTF is set by hardware. Before clearing BTF (by first reading I2C_SR1 and then writing to the I2C_DR register), the I2C interface will keep SCL low.

Close communication

After writing the last byte in the I2C_DR register, generate a stop condition by setting the STOP bit (see EV8_2 in the figure), then the I2C interface will automatically return to slave mode (MSL bit cleared).

Note: Stop condition should be programmed during EV8_2 event, when either TxE or BTF is set.

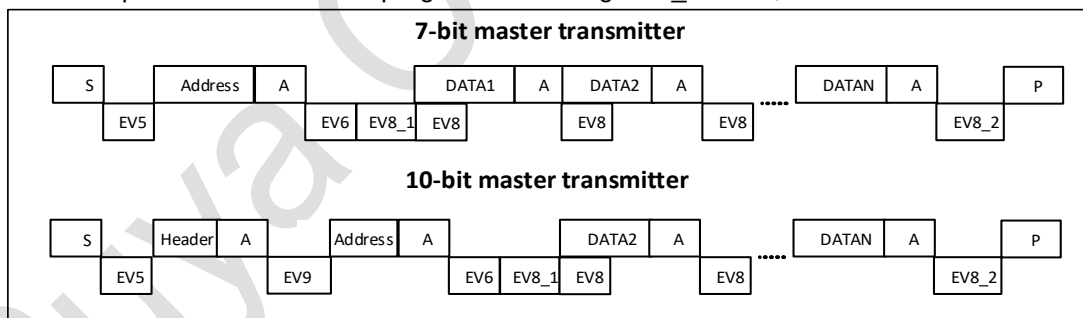


Figure 30-5 Master transmission sequence diagram

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5: SB=1, clear this bit by reading I2C_SR1 and then writing data to the I2C_DR register

EV6: ADDR=1, clear this bit by first reading I2C_SR1 and then reading I2C_SR2

EV8_1: TxE=1, shift register empty, data register empty, write DATA1 to the I2C_DR register

EV8: TxE=1, shift register not empty, data register empty, write DATA2 to the I2C_DR register, this bit is cleared

EV8_2: TxE=1, BTF=1, write to the stop bit register (STOP), when the hardware issues the stop bit, TxE and BTF are cleared

EV9: ADD10=1, read I2C_SR1 and then write to the I2C_DR register to clear this event.

Note:

- 1) The EV5, EV6, EV8_1 and EV8_2 events stretch SCL low until the end of the corresponding software sequence.

- 2) The EV8 software sequence must complete before the end of the current byte transfer. The software sequence in EV8 cannot be completed before the end of the current byte transmission. It is recommended to use BTF instead of TxNE, but this will slow down communication.

30.3.5.5. Master receiver

After sending the address and clearing ADDR, the I2C interface enters master receive mode. In this mode, the I2C interface receives data bytes from the SDA line and sends them to the I2C_DR register via the internal shift register. After each byte the interface generates in sequence:

- An acknowledge pulse if the ACK bit is set
- The RxNE bit is set and an interrupt is generated if the ITEVFEN and ITBUFEN bits are set.
- If the RxNE bit is set and the data in the I2C_DR register is not read before the end of receiving new data, the hardware will set BTF=1, and the I2C interface will keep SCL low until BTF is cleared; reading I2C_SR1 and then reading the I2C_DR register will clear the BTF bit.

30.3.5.6. Close communication

Method 1: When I2C is set as the highest priority interrupt in the application

The master sends a NACK for the last byte received from the slave. After receiving this NACK, the slave releases the control of the SCL and SDA lines. The master can then send a stop/repeated start bit.

- To generate the non-acknowledge pulse after the last received data byte, the ACK bit must be cleared just after reading the second last data byte (after second last RxNE event).
- To generate a stop/repeated start bit, the software must set the stop/start bit after reading the second-to-last data byte (after the second-to-last RxNE event).
- When receiving only one byte, disable the acknowledgment and stop condition generation bits immediately after EV6 (after clearing ADDR in EV6_1). After generating the stop bit, the I2C interface automatically returns to slave mode (MSL bit is cleared).

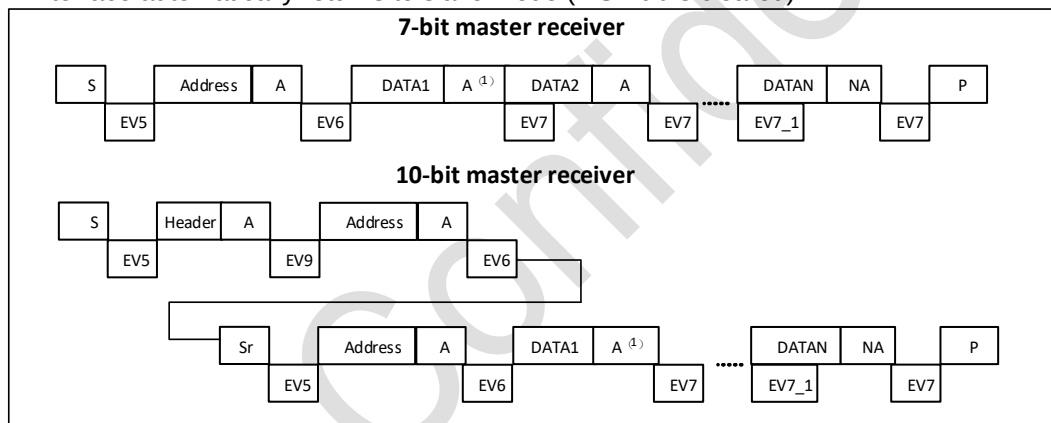


Figure 30-6 Method 1: Transmission sequence during master reception

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5: SB=1, read I2C_SR1, then write I2C_DR register, this bit is cleared

EV6: ADDR=1, cleared by reading I2C_SR1 followed by reading I2C_SR2

EV6_1: no associated flag event, used for 1 byte reception only.

EV7: RxNE=1, read I2C_DR register, this bit is cleared

EV7_1: RxNE=1, read I2C_DR register, write ACK=0 and set STOP

EV9: ADD10=1, read I2C_SR1 and then write to the I2C_DR register to clear this event.

Note:

- 1) If a single byte is received, it is NA.
- 2) The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
- 3) The EV7 software sequence must complete before the end of the current byte transfer. In EV7, the software sequence cannot be completed before the current byte transfer is finished. It is recommended to use BTF instead of RxNE, but this will slow down communication.
- 4) The EV6_1 or EV7_1 software sequence must complete before the ACK pulse of the current byte transfer.

Method 2: I2C interrupt is not the highest priority in the application, or polling mode is used

With this method, DATA N-2 is not read, so after DATA N-1, the communication is stretched (both RxNE and BTF are set). Then, before reading DATA N-2 from the I2C_DR register, clear the ACK bit to ensure it is cleared before the ACK of DATA N. After this, after reading DATA N-2, set the stop/start bit and read DATA N-1. After RxNE is set, read DATA N.

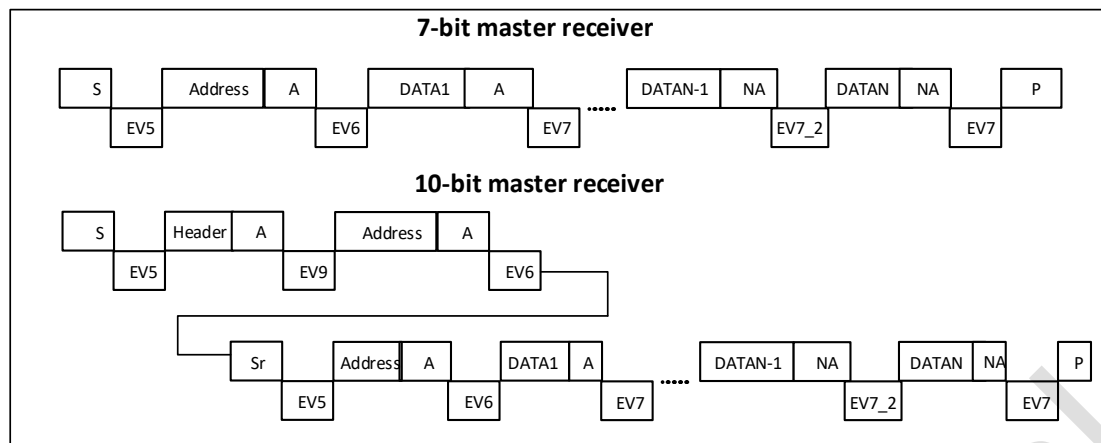


Figure 30-7 Method 2: Master receive sequence N>2

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5: SB=1, first read I2C_SR1 register, then write I2C_DR register, clear this bit

EV6: ADDR, first read I2C_SR1, then read I2C_SR2, clear this bit

EV7: RxNE=1, read I2C_DR register to clear this bit

EV7_2: BTF=1, DATA N-2 is in the I2C_DR register, DATA N-1 is in the shift register, write ACK=0, read DATA N-2 from the I2C_DR register. Set STOP, read DATA N-1

EV9: ADD10=1, read I2C_SR1 and then write to the I2C_DR register to clear this event.

Note:

- 1) The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
- 2) The EV7 software sequence must complete before the end of the current byte transfer. In EV7, the software sequence cannot be completed before the current byte transfer is finished. It is recommended to use BTF instead of RxNE, but this will slow down communication.

■ When 3 bytes remain to be read:

- RxNE=1 => No operation (DATA N-2 not read).
- Receive DATA N-1
- BTF=1, shift and data registers full: I2C_DR register contains DATA N-2, shift register contains DATA N-1 => SCL pulled low: no more data to be received on the bus
- Clear ACK bit
- Read DATA N-2 from I2C_DR register => This initiates the reception of DATA N by the shift register
- DATA N reception completed (reply with NACK)
- Program START/STOP bit
- Read DATA N-1
- RxNE=1
- Read DATA N

The procedure described above is valid for N>2. The cases where a single byte or two bytes are to be received should be handled differently, as described below:

■ Case of two bytes to be received:

- Software sets POS and ACK bits
- Wait for hardware to set ADDR
- Software clears ADDR bit
- Software clears ACK bit
- Wait for hardware to set BTF
- Software writes STOP bit
- Software reads I2C_DR twice

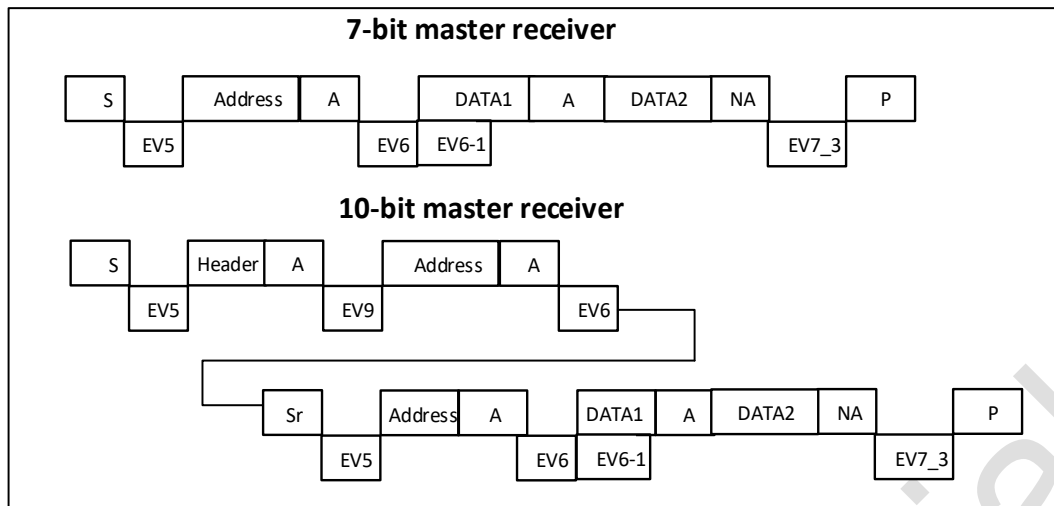


Figure 30-8 Method 2: Transfer sequence when master receives, N=2

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5: SB=1, first read I2C_SR1 register, then write I2C_DR register, clear this bit

EV6: ADDR=1, cleared by reading I2C_SR1 followed by reading I2C_SR2

EV6_1: No associated flag event. The acknowledge disable should be done just after EV6, that is after ADDR is cleared.

EV7_3: BTF=1, write STOP=1, then read I2C_DR twice (DATA1 and DATA2)

EV9: ADD10=1, read I2C_SR1 and then write to the I2C_DR register to clear this event.

Note:

- 1) The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
 - 2) The EV6_1 software sequence must complete before the ACK pulse of the current byte transfer.
- Software operations for single-byte reception case
- In the ADDR event, clear the ACK bit
 - Clear ADDR
 - Program STOP/START bit
 - Read the data after the RxNE flag is set.

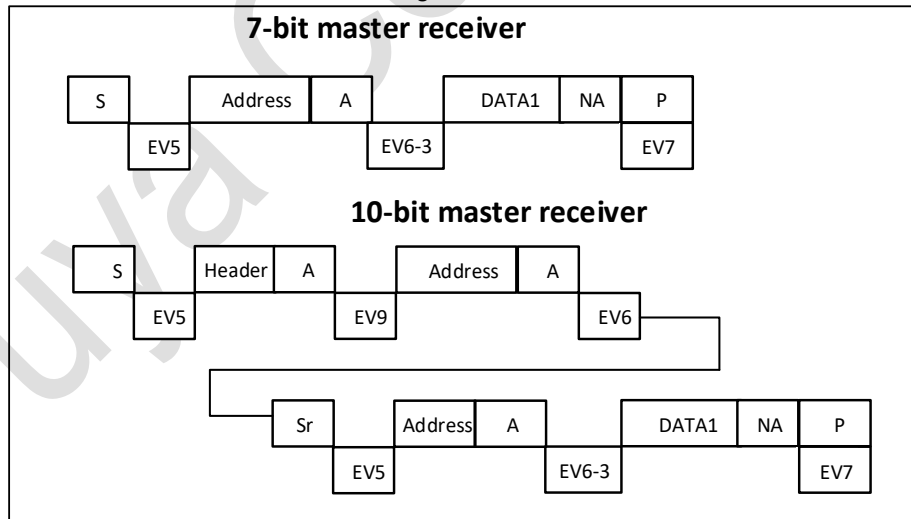


Figure 30-9 Method 2: Transfer sequence when master receives, N=1

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5: SB=1, first read I2C_SR1 register, then write I2C_DR register, clear this bit

EV6_3: ADDR = 1, program ACK = 0. Clear ADDR by reading I2C_SR1 register followed by reading I2C_SR2 register. Program STOP =1 just after ADDR is cleared.

EV7: RxNE=1, read I2C_DR register to clear this bit

EV9: ADD10=1, read I2C_SR1 and then write to the I2C_DR register to clear this event.

Note:

EV5, EV6, EV8_1, and EV8_2 events stretch the low level of the SCL until the end of execution of the corresponding software sequence.

30.3.6. Error condition

30.3.6.1. Bus error (BERR)

A bus error occurs when the I2C interface detects an external stop or start bit during the transmission of an address or data byte. In this case:

- The BERR bit is set, and if the ITERREN bit is enabled, an interrupt is generated.
- In Slave mode: data are discarded, and the lines are released by hardware:
 - If it is an erroneous start bit, the slave treats it as a repeated start bit and waits for an address or stop bit.
 - If it is an erroneous stop bit, the slave operates as if it were a normal stop bit, and the hardware releases the bus.
- In Master mode: the lines are not released and the state of the current transmission is not affected. It is up to the software to abort or not the current transmission.

30.3.6.2. Acknowledge Error (AF)

An acknowledge error occurs when the interface detects a not-acknowledge (NACK) bit. In this case:

- The AF bit is set, and an interrupt is generated if the ITERREN bit is set
- A transmitter which receives a NACK must reset the communication:
 - If Slave: lines are released by hardware.
 - If in master mode, the software must generate a stop bit or a repeated start bit.

30.3.6.3. Arbitration lost (ARLO)

This error occurs when the I2C interface detects an arbitration lost condition. In this case:

- The ARLO bit is set by hardware (and an interrupt is generated if the ITERREN bit is set)
- The I2C Interface goes automatically back to slave mode (the MSL bit is cleared). When the I2C interface loses arbitration, it cannot respond to its slave address in the same transmission, but it can respond after the master that wins the bus sends a repeated start bit.
- Lines are released by hardware

30.3.6.4. Overrun/Underrun Error (OVR)

In slave mode, if clock stretching is disabled, an overrun error occurs when the I2C interface is receiving data and has received a byte (RxNE=1), but the previous byte in the I2C_DR register has not been read.

In this case:

- The last received byte is lost.
- In case of an overrun error, the software should clear the RxNE bit, and the transmitter should resend the last byte.

In slave mode, if clock stretching is disabled, an underrun error occurs when the I2C interface is transmitting data and a new byte has not been written to the I2C_DR register (TxNE=1) before the clock for the next byte arrives. In this case:

- The previous byte in the I2C_DR register will be transmitted again.
- The user should ensure that the receiver discards the repeated data in case of an underrun. The transmitter should update the I2C_DR register within the time specified by the I2C bus standard.

When transmitting the first byte, the I2C_DR register must be written after clearing ADDR and before the first SCL rising edge; if this is not possible, the receiver should discard the first data.

30.3.7. SDA/SCL Line Control

- If clock stretching is enabled:
 - Transmit mode: If TxNE=1 and BTF=1, the I2C interface holds the clock line low before transmission to wait for the software to read I2C_SR1 and then write data into the I2C_DR register (both I2C_DR and the shift register are empty).
 - Receive mode: If RxNE=1 and BTF=1, the I2C interface holds the clock line low after receiving a data byte to wait for the software to read I2C_SR1 and then read the I2C_DR register (both DR and the shift register are full).
- If clock stretching is disabled in Slave mode:
 - If RxNE=1 and I2C_DR has not been read before receiving the next byte, an overflow occurs. The last received byte is lost.
 - If TxNE=1 and no new data is written to I2C_DR before the next byte must be transmitted, an underflow occurs. The same byte will be sent again.
 - Hardware does not implement control for write conflicts.

30.3.8. DMA request

The DMA request (when enabled) is used only for data transmission. DMA requests are generated by data register becoming empty in transmission and data register becoming full in reception. The DMA must be initialized and enabled before the I2C data transfer. The DMAEN bit must be set in the I2C_CR2 register before the ADDR event.

In master or slave mode, when the clock stretching feature is enabled, the DMAEN bit can be set during the ADDR event before clearing ADDR. The DMA request must be served before the end of the current byte transfer. When the DMA data transfer length reaches the value set by DMA, the DMA controller sends an EOT (End of Transfer) to I2C and generates a transfer complete interrupt (if the interrupt enable bit is active):

- Master transmission: In the EOT interrupt service routine, disable the DMA request, then wait for the BTF event before setting the STOP bit.
- Master reception: When the number of data bytes to be received is greater than or equal to 2, the DMA controller sends a hardware signal EOT_1, which corresponds to DMA transfer (number of bytes - 1). If, in the I2C_CR2 register, the LAST bit is set, I2C automatically sends a NACK after the next byte following EOT_1. The user can generate a Stop condition in the DMA transfer complete interrupt routine if enabled.

30.3.8.1. DMA transmission

DMA mode can be enabled for transmission by setting the DMAEN bit in the I2C_CR2 register. Data will be loaded from a Memory area configured using the DMA peripheral to the I2C_DR register whenever the TxE bit is set. To allocate a DMA channel for I2C, follow these steps (x is the channel number):

- 1) Configure the DMAx_MAP register in SYSCFG_CFGR3 or SYSCFG_CFGR4 to select the DMA channel.
- 2) Set the I2C_DR register address in the DMA_CPARx register. The data will be moved to this address from the memory after each TxE event.
- 3) Set the memory address in the DMA_CMARx register. The data will be loaded into I2C_DR from this memory after each TxE event.
- 4) Set the desired number of bytes to transfer in the DMA_CNDTRx register. After each TxE event, this value will be decremented.
- 5) Configure the channel priority using the PL[0:1] bits in the DMA_CCRx register.
- 6) Set the DIR bit in the DMA_CCRx register, and configure the interrupt request to be issued when half or all of the transfer is completed, as required by the application.
- 7) Activate the channel by setting the EN bit in the DMA_CCRx register.

When the data transfer quantity set in the DMA controller is completed, the DMA controller sends an EOT/EOT_1 signal to the I2C interface, indicating the end of the transfer. The DMA generates an interrupt, if enabled, on the DMA stream interrupt vector.

Note: Do not enable the ITBUFEN bit in the I2C_CR2 register if DMA is used for transmission.

30.3.8.2. DMA reception

DMA mode can be enabled for reception by setting the DMAEN bit in the I2C_CR2 register. Each time a data byte is received, the DMA transfers the data from the I2C_DR register to the configured memory area (refer to the DMA documentation). To set up a DMA channel for I2C reception, follow these steps (x is the channel number):

- 1) Configure the DMAx_MAP register in SYSCFG_CFGR3 or SYSCFG_CFGR4 to select the DMA channel.
- 2) Set the address of the I2C_DR register in the DMA_CPARx register. The data will be moved from this address to the memory after each RxNE event.
- 3) Set the memory area address in the DMA_CMARx register. The data will be loaded from the I2C_DR register to this memory area after each RxNE event.
- 4) Set the desired number of bytes to transfer in the DMA_CNDTRx register. After each RxNE event, this value will be decremented.
- 5) Configure channel priority using the PL[0:1] bits in the DMA_CCRx register.
- 6) Clear the DIR bit in the DMA_CCRx register, and set the interrupt request to occur when either half or all of the data transfer is completed, according to application requirements.
- 7) Set the EN bit in the DMA_CCRx register to activate the channel.

When the number of data transfers set in the DMA controller is completed, the DMA controller sends an EOT/EOT_1 signal to the I2C interface indicating the end of the transfer. The DMA generates an interrupt, if enabled, on the DMA stream interrupt vector.

Note: Do not enable the ITBUFEN bit in the I2C_CR2 register if DMA is used for reception.

30.3.9. Packet Error Checking

Packet Error Checking (PEC) is used to improve communication reliability by calculating each bit of serial data using the following CRC-8 polynomial:

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

- PEC calculation is activated by the ENPEC bit in the I2C_CR1 register. PEC uses the CRC-8 algorithm to calculate all message bytes, including the address and read/write bits.
 - During transmission: Set the PEC transmission bit in the I2C_CR1 register at the last TxEvent, and the PEC will be sent after the last byte.
 - During reception: Set the PEC bit in the I2C_CR1 register after the last RxNE event. If the next received byte does not match the internally calculated PEC, the receiver sends a NACK. If acting as a master receiver, a NACK will be sent after the PEC regardless of the verification result. The PEC bit must be set before receiving an ACK pulse for the current byte.
- The PECERR error flag/interrupt is available in the I2C_SR1 register.
- If both DMA and PEC are activated:
 - During transmission: When the I2C interface receives an EOT signal from the DMA controller, it automatically sends the PEC after the last byte.
 - During reception: When the I2C interface receives an EOT_1 signal from the DMA, it automatically treats the next byte as the PEC and verifies it. A DMA request is generated after receiving the PEC.
- To allow intermediate PEC transmission, there is a control bit (LAST bit) in the I2C_CR2 register to determine whether it is truly the last DMA transfer. If it is indeed the last DMA request for the master receiver, a NACK is automatically sent after receiving the last byte.
- PEC calculation becomes invalid upon arbitration loss.

30.3.10. System Management Bus (SMBus)

The System Management Bus (SMBus) is a two-wire interface that allows devices to communicate with each other or with the rest of the system. It is based on I2C principles of operation. SMBus provides a control bus for system and power management related tasks.

This peripheral is compatible with the SMBus specification (<http://smbus.org>).

The System Management Bus specification involves three types of devices:

- Slave devices, which receive or respond to commands.
- Master devices, which issue commands, generate clocks, and terminate transmissions.
- Host, a dedicated master device that provides the main interface to the system CPU. The host must have master-slave functionality and must support the SMBus host notification protocol. Only one host is allowed in a system.

This peripheral can be configured as a master, slave, or host.

The I2C module of this project supports SMBus/PMbus functionality.

Differences between SMBus and I2C:

Table 30-1 Differences between SMBus and I2C

SMBus	I2C
Maximum speed 100 kHz	Maximum transmission speed ~1MHz
Minimum clock speed 10 kHz	No minimum clock speed
25 ms clock low timeout	No timeout
Logic levels are fixed	Logic levels are VDDD dependent
Different address types (reserved, dynamic, etc.)	7-bit, 10-bit and general call slave address types
Different bus protocols (fast commands, handling calls, etc.)	No bus protocols

Similarities between SMBus and I2C:

- 2-wire bus protocol (1 clock, 1 data) + optional SMBus alert line
- Master-slave communication, with the host providing the clock
- SMBus data format similar to I2C 7-bit addressing format

30.3.10.1. Bus protocols

There are 11 available command protocols for any given device. A device can choose any one of these 11 protocols or use all 11 for communication. These 11 protocols are Quick Command, Send Byte, Receive Byte, Write Byte, Write Word, Read Byte, Read Word, Process Call, Block Read, Block Write, and Block Write-Block Read Process Call.

These protocols should be implemented by the user software.

For detailed information about these protocols, refer to the SMBus specification (<http://smbus.org>).

30.3.10.2. Address resolution protocol (ARP)

SMBus slave address conflicts can be resolved by dynamically assigning a new unique address to each slave. The Address Resolution Protocol (ARP) has the following features:

- Uses the standard SMBus physical layer arbitration mechanism to assign addresses;
- The assigned address remains unchanged while the device is powered, and the device is allowed to retain its address after power-off.
- After address assignment, there is no additional SMBus packet overhead (meaning accessing devices with assigned addresses takes the same time as accessing fixed-address devices);
- Any SMBus host can traverse the bus.

30.3.10.3. SMBus alert mode

SMBus Alert is an optional signal with an interrupt line, used for devices that wish to expand their control capabilities at the cost of one pin. SMBALERT is a wired-AND signal, just like SCL and SDA signals. SMBALERT is used in conjunction with the SMBus general call address. Messages related to SMBus are 2 bytes in size. A slave-only device can signal the host via SMBALERT that it wishes to communicate by setting the ALERT bit in the I2C_CR1 register. The host handles this interrupt and accesses all SMBALERT devices using the Alert Response Address ARA (address value 0001100x). Only devices that pull SMBALERT low can respond to ARA. This status is indicated by the SMBALERT status flag in the I2C_SR1 register. The host performs a modified receive byte operation. The 7-bit device address provided by the slave transmitter is placed in the 7 most significant bits of the byte, with the eighth bit being '0' or '1'.

If multiple devices pull SMBALERT low, the highest-priority device (smallest address) will win communication rights during address transmission through standard arbitration. After acknowledging the slave address the device must disengage its SMBALERT pull-down. If the host still sees SMBALERT low when the message transfer is complete, it knows to read the ARA again.

Hosts that do not implement the SMBALERT signal can access ARA periodically.

For more details on SMBus Alert mode, refer to the SMBus Specification Version 2.0 (<http://smbus.org/specs/>).

30.3.10.4. Timeout error

There are differences in the timing specifications between I2C and SMBus. SMBus defines a clock low timeout of 25ms. SMBus specifies tLOW:SEXT as the cumulative clock low extension time for slaves. SMBus specifies TLOW:MEXT as the cumulative clock low extension time for the host. For more timeout details, refer to the SMBus Specification Version 2.0 (<http://smbus.org/specs/>).

The status flags TIMEOUT or Tlow error in I2C_SR1 indicate the status of this feature.

Enable timeout detection by setting the TIMOUTEN and TEXTEN bits in the I2C_TIMEOUTR register to 1. The timer must detect timeout conditions before the maximum time specified in the SMBus specification.

tTIMEOUT check

In order to enable the tTIMEOUT check, the 12-bit TIMEOUTA[11:0] bits must be programmed with the timer reload value in order to check the tTIMEOUT parameter. The TIDLE bit must be configured to '0' in order to detect the SCL low level timeout.

Then the timer is enabled by setting the TIMOUTEN in the I2C_TIMEOUTR register.

If the SCL low duration exceeds $(\text{TIMEOUTA} + 1) \times 2048 \times \text{tPCLK}$, the TIMEOUT flag in the I2C_SR1 register will be set to 1.

(max tTIMEOUT = 25 ms).

Note: When the TIMOUTEN bit is set to 1, changing the configuration of TIMEOUTA[11:0] bits and TIDLE bit is not allowed.

tLOW: SEXT and tLOW: MEXT checks

The TIMEOUTB timer must be configured based on whether the peripheral is set as a master or slave device, to verify tLOW:SEXT for slave devices and tLOW:MEXT for master devices. As the standard specifies only a maximum, the user can choose the same value for the both.

Then the timer is enabled by setting the TEXTEN bit in the I2C_TIMEOUTR register.

If the cumulative time the SMBus peripheral extends SCL exceeds $(\text{TIMEOUTB} + 1) \times 2048 \times \text{tPCLK}$, the TIMEOUT flag in the I2C_SR1 register will be set to 1.

Bus idle detection

In order to enable the tIDLE check, the 12-bit TIMEOUTA[11:0] field must be programmed with the timer reload value in order to obtain the tIDLE parameter. The TIDLE bit must be configured to '1' in order to detect both SCL and SDA high level timeout.

Then the timer is enabled by setting the TIMOUTEN in the I2C_TIMEOUTR register.

If the high-level duration of the SCL and SDA lines exceeds $(\text{TIMEOUTA}+1) \times 4 \times \text{tPCLK}$, the TIMEOUT flag in the I2C_SR1 register will be set to 1.

Note: When TIMEOUTEN is set to 1, changes to the TIMEOUTA and TIDLE configurations are not allowed.

30.3.10.5. How to use the interface in SMBus mode

To switch from I2C mode to SMBus mode, the following steps should be performed:

- Set the SMBUS bit in the I2C_CR1 register;
- Configure the SMBTYPE and ENARP bits in the I2C_CR1 register according to application requirements.

If the device is to be configured as a master, refer to the "I2C Master Mode" section for steps to generate the start condition. Otherwise, refer to the "I2C Slave Mode" section.

The software program must handle various SMBus protocols.

- If ENARP=1 and SMBTYPE=0, use the SMBus device default address.
- If ENARP=1 and SMBTYPE=1, use the SMBus host header field.
- If SMBALERT=1, use the SMBus alert response address.

30.3.10.6. PMbus

PMBus is based on SMBus, and the transmission logic is exactly the same as SMBus. The difference is that PMbus defines some functions related to power management (done by software).

30.3.10.7. SMBus: I2C_TIMEOUTR register configuration example

This section is relevant only when SMBus feature is supported.

Configure the maximum duration of **tTIMEOUT to 25 ms**.

Table 30-2 Example TIMEOUTA settings for different PCLK frequencies (maximum tTIMEOUT = 25 ms)

fPCLK	TIMEOUTA[11:0] bits	TIDLE bit	TIMEOUTEN bit	tTIMEOUT
8 MHz	0x61	0	1	$98 \times 2048 \times 125 \text{ ns} = 25 \text{ ms}$
16 MHz	0xC3	0	1	$196 \times 2048 \times 62.5 \text{ ns} = 25 \text{ ms}$
32 MHz	0x186	0	1	$391 \times 2048 \times 31.25 \text{ ns} = 25 \text{ ms}$
48 MHz	0x249	0	1	$586 \times 2048 \times 20.08 \text{ ns} = 25 \text{ ms}$

Configure the maximum duration of **tLOW:SEXT and tLOW:MEXT to 8 ms**

Table 30-3 Example TIMEOUTB settings at different PCLK frequencies

fPCLK	TIMEOUTB[11:0] bits	TEXTEN bit	tLOW: EXT
8 MHz	0x1F	1	$32 \times 2048 \times 125 \text{ ns} = 8 \text{ ms}$
16 MHz	0x3F	1	$64 \times 2048 \times 62.5 \text{ ns} = 8 \text{ ms}$
48 MHz	0xBB	1	$188 \times 2048 \times 20.08 \text{ ns} = 8 \text{ ms}$

Configure the maximum duration of **tIDLE to 50 μ s**

Table 30-4 Example TIMEOUTA settings for different PCLK frequencies (maximum tIDLE = 50 μ s)

fPCLK	TIMEOUTB[11:0] bits	TIDLE bit	TIMEOUTEN bit	TIDLE
8 MHz	0x63	1	1	$100 \times 4 \times 125 \text{ ns} = 50 \text{ us}$
16 MHz	0xC7	1	1	$64 \times 4 \times 62.5 \text{ ns} = 50 \text{ us}$
48 MHz	0x257	1	1	$600 \times 4 \times 20.08 \text{ ns} = 50 \text{ us}$

SMBus timeout specifications

Table 30-5 Timeout specifications

Symbol	Parameter	Limits		Unit
		Min	Max	
tTIMEOUT	Detect clock low timeout	25	35	ms
tLOW:SEXT (1)	Cumulative Clock Low Extension Time (Slave)	-	25	ms
tLOW:MEXT (2)	Cumulative Clock Low Extension Time (Master)	-	10	ms
tIDLE	SCL and SDA high timeout	-	50	μ s

- 1) tLOW:SEXT is a cumulative time period during which the clock signal can be stretched by a given slave device from the initial start to stop of a message. Other slave devices or the master device may also stretch the clock, causing the total extended time of clock low level to exceed tLOW:SEXT. Therefore, when measuring this parameter, the device should be the only one addressed by the full-speed master device.
- 2) tLOW:MEXT is a cumulative time period during which the master device can stretch the clock signal within each byte of a message (defined as START to ACK, ACK to ACK, or ACK to STOP). The slave device or other master device may also extend the clock, which in turn causes

the total time of the clock low to exceed tLOW: MEXT (for a given byte). Therefore, the full-speed master only addresses one slave when measuring this parameter.

30.4. I2C interrupts

Table 30-6 I2C interrupt requests

Interrupt event	Event flag	Enable control bit
Start bit transmitted (Master)	SB	ITEVTEN
Address transmitted (Master) or address matched (Slave)	ADDR	
10-bit address header sequence transmitted (Master)	ADD10	
Stop received (Slave)	STOPF	
Data byte transfer finished	BTF	
Receive buffer not empty	RxNE	ITEVTEN and ITBUFEN
Transmit buffer empty	TxE	
Bus error (BERR)	BERR	ITERREN
Arbitration lost (Master)	ARLO	
Acknowledge failure	AF	
Overflow/Underflow	OVR	
PEC error	PECERR	
Timeout/TLOW error	TIMEOUT	
SMBus Alert	SMBALERT	

30.5. I2C Debug Mode

When the microcontroller enters debug mode, the SMBUS timeout control either continues normal operation or stops, depending on the DBG_I2Cx_SMBUS_TIMEOUT configuration bit in the DBGMCU module.

30.6. I2C Low Power

When the MCU is in Stop low-power mode, it can receive an address via I2C. If the address matches, the MCU can be awakened; otherwise, it remains in the previous low-power state (taking I2C-only wake-up as an example).

- 1) To achieve the above functionality, I2C_CR1's PE and WUPEN must be enabled before entering Stop low-power mode, and the I2C clock must be enabled in RCC.
- 2) The Stop low-power mode wake-up process also supports a configurable timeout period, which can be set via I2C_CR1's WKUP_CNT and WKUP_DIV. If ITERREN is enabled, the MCU is awakened after a timeout occurs; if ITERREN is not enabled, the previous state is maintained after a timeout.

30.7. I2C IO Configuration

For IOs used as I2C, they must be configured in the GPIO registers as alternate function, open-drain output, pull-up mode, and the corresponding bits in SYSCFG_IOCFG.P*_EIIIC registers must be enabled.

30.8. I2C registers

30.8.1. I2C Control Register 1 (I2C_CR1)

Offset address: 0x00

Reset value : 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Re s.	Res.	Re s.	Res.	Re s.	Res.	Res.	Res.	Res.	Res.	Res.	WKUP_CNT [1: 0]		WKUP_DIV [1: 0]	
												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWRS T	Re s.	ALER T	PE C	PO S	AC K	STO P	STAR T	NO STRETC H	ENG C	ENPE C	ENAR P	SMBTY PE	WUPE N	SMBU S	PE
RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19:18	WKUP_CNT [1: 0]	RW	2'b10	Low-power wake-up timeout count

				00: 2 01: 8 10: 32 11: 128 Note: The timeout duration is calculated as $(WKUP\_DIV/fPCLK)*WKUP\_CNT$ Example: If WKUP_DIV is set to 2'b11 (1024), fPCLK is 8M, and WKUP_CNT is set to 2'b00 (2), the timeout duration is $(1024/8)*2=256\mu s$ Note: When fPCLK is 72M, the timeout duration ranges from 3.5 μs to 1817.6 μs ; When fPCLK is 4M, the timeout duration ranges from 64 μs to 32768 μs
17:16	WKUP_DIV [1: 0]	RW	2'b0	PCLK division factor. The divided clock is used to count the timeout duration for waking up from all low-power stop modes. 00: 128 01: 256 10: 512 11: 1024
15	SWRST	RW	0	Software reset When set, the I2C is under reset state. Before resetting this bit, make sure the I2C lines are released and the bus is free. 0: I2C Peripheral not under reset 1: I2C Peripheral under reset state Note: This bit can be used to reinitialize the I2C in case of errors or lock states. For example, when the BUSY bit is 1 and no stop condition is detected on the bus.
14	Reserved	-	-	Reserved
13	ALERT	RW	0	SMBus Alert 0: Release the SMBA pin to high level and disable the alert response to address header: 0001100x followed by NACK; 1: Release the SMBA pin to high level and enable the alert response to address header: 0001100x followed by ACK; When PE = 0, it is cleared by the hardware. Note: If SMBus functionality is not supported, this register is fixed at 0.
12	PEC	RW	0	Packet error checking This bit can be set or cleared by software. Hardware can clear it after PEC transmission, upon detecting a start condition, stop condition, or when PE=0; 0: No PEC transfer 1: PEC transfer (in Tx or Rx mode) Note: PEC calculation is corrupted by an arbitration loss.
11	POS	RW	0	ACK/PEC position (for data reception) This bit is set and cleared by software and cleared by hardware when PE=0. 0: ACK bit controls the (N)ACK of the current byte being received in the shift register. The PEC bit indicates that current byte in shift register is a PEC. 1: ACK bit controls the (N)ACK of the next byte which will be received in the shift register. The PEC bit indicates that the next byte in the shift register is a PEC. Note: The POS bit is used when the procedure for reception of 2 bytes is followed. It must be configured before data reception starts. To NACK the 2nd byte, the ACK bit must be cleared just after ADDR is cleared.

				To detect the PEC of the second byte, the PEC bit must be set during the ADDR extension event after configuring the POS bit.
10	ACK	RW	0	Acknowledge enable This bit is set and cleared by software and cleared by hardware when PE=0. 0: No acknowledge returned 1: Acknowledge returned after a byte is received. (matched address or data)
9	STOP	RW	0	Generate stop condition The software can set/clear this register, or it is cleared by hardware when a stop bit is detected; it is set by hardware when a timeout error is detected. In Master Mode: 0: No stop bit generated 1: Generate a stop bit after the current byte transmission is completed or after the current start bit is issued In slave mode: 0: No stop bit generated 1: Release the SCL and SDA lines after the current byte transmission is completed
8	START	RW	0	Start bit generation The software can set/clear this register, or it is cleared by hardware when a start bit is issued or when PE=0. In Master mode: 0: No start bit generated 1: Generate a start bit/repeated start bit In Slave mode: 0: No start bit generated 1: Generate a start bit when the bus is idle (and automatically switch to master mode by hardware)
7	NOSTRETCH	RW	0	Clock stretching disabled (slave) This bit is used to disable clock stretching in slave mode when ADDR or BTF flag is set, until it is reset by software. 0: Clock stretching enabled 1: Clock stretching disabled
6	ENGCG	RW	0	Broadcast call enable 0: General call disabled. . Address 00h is NACKed. 1: General call enabled. Address 00h is ACKed.
5	ENPEC	RW	0	PEC enabled 0: PEC calculation disabled 1: PEC calculation enabled
4	ENARP	RW	0	ARP enabled 0: Prohibit ARP 1: Enable ARP SMBus device default address recognized if SMBTYPE=0; SMBus host address recognized if SMBTYPE=1. Note: If SMBus functionality is not supported, this register is fixed at 0.
3	SMBTYPE	RW	0	SMBus type 0: SMBus device 1: SMBus host Note: If SMBus functionality is not supported, this register is fixed at 0.
2	WUPEN	RW	0	Stop low-power mode wake-up enable (slave) 0: Stop low-power mode wake-up disabled 1: Stop low-power mode wake-up enabled Note: If the stop low-power mode wake-up function is not supported, this bit is reserved and forced to zero by hardware.
1	SMBUS	RW	0	SMBus mode 0: I2C mode 1: SMBus mode Note: If the register does not support SMBus functionality, it is fixed at 0.

0	PE	RW	0	I2C module enable 0: Disabled 1: Enabled The corresponding I/O pins must be configured for alternate function. Note: If this bit is reset while a communication is on going, the peripheral is disabled at the end of the current communication, when back to IDLE state. All bit resets due to PE=0 occur at the end of the communication. In master mode, this bit must never be cleared before the end of communication.

Note: When the STOP/START/PEC bits are set, software should not perform any write operations to I2C_CR1 until the hardware clears these bits; otherwise, the STOP/START/PEC bits may be set a second time.

30.8.2. I2C Control Register 2 (I2C_CR2)

Offset address: 0x04

Reset value : 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN	Res.	FREQ[6:0]						
			RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	LAST	RW	0	DMA last transfer 0: Next DMA EOT is not the last transfer 1: Next DMA EOT is the last transfer Note: This bit is used in master receive mode to generate a NACK during the last data reception.
11	DMAEN	RW	0	DMA request enable. 0: Disable DMA request 1: DMA request enabled when TxNE=1 or RxNE=1.
10	ITBUFEN	RW	0	Buffer interrupt enable 0: TxNE = 1 or RxNE = 1 does not generate any interrupt. 1: Generate an interrupt when TxNE=1 or RxNE=1 (regardless of DMAEN value)
9	ITEVTEN	RW	0	Event interrupt enable 0: Disabled 1: Event interrupt enabled This interrupt is generated when: ■ SB=1 (master mode) ■ ADDR = 1 (Master/Slave) ■ ADD10=1 (master mode) ■ STOPF=1 (slave mode) ■ BTF = 1 with no TxNE or RxNE event ■ TxNE event to 1 if ITBUFEN = 1 ■ RxNE event to 1 if ITBUFEN = 1
8	ITERREN	RW	0	Error interrupt enable. 0: Error interrupt disabled; 1: Enable error interrupt; This interrupt is generated when: ■ BERR = 1 ■ ARLO = 1 ■ AF = 1 ■ OVR = 1 ■ PECERR = 1 ■ TIMEOUT = 1 ■ SMBALERT=1
7	Reserved	-	-	Reserved
6:0	FREQ	RW	0	I2C clock frequency This register must be configured with a supported APB clock frequency value to generate data setup and hold times compatible with the I2C protocol.

				The minimum configurable frequency is 4MHz (100k), 8MHz (400k), and 16MHz (1MHz). The maximum frequency is the highest APB clock frequency of the chip. 0000000: Disabled 0000001: Disabled 0000010: Disabled 0000011: Disabled 0000100: 4 MHz 0100100: 36 MHz 0110000: 48 MHz 1001000: 72 MHz Greater than 1001000: Disabled.

30.8.3. I2C own address register 1 (I2C_OAR1)

Offset address: 0x08

Reset value : 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDMODE	Res.	Res.	Res.	Res.	Res.	ADD[9:8]		ADD[7:1]							ADD0
RW						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	ADDMODE	RW	0	Addressing mode (slave mode). 0: 7 bit slave address (does not respond to 10 bit addresses); 1: 10 bit slave address (does not respond to 7 bit addresses); Note: If the register does not support 10-bit addressing, it is fixed at 0.
14:10	Reserved	-	-	Reserved
9:8	ADD[9:8]	RW	0	Interface address. 7-bit addressing mode: don't care 10-bit addressing mode: bits 9:8 of address Note: If the register does not support 10-bit addressing, it is fixed at 0.
7:1	ADD[7:1]	RW	0	Bits 7:1 of interface address
0	ADD0	RW	0	Interface address. 7-bit addressing mode: don't care Bit 0 of the 10-bit address mode address. Note: If the register does not support 10-bit addressing, it is fixed at 0.

30.8.4. I2C own address register 2 (I2C_OAR2)

Note: This register is fixed to 0 if the dual-byte address feature is not supported.

Offset address: 0x0C

Reset value : 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	OA2MSK [2:0]			ADD2 [7:1]							ENDUAL
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10:8	OA2MSK [2:0]	RW	0	ADD2 mask configuration. 000: No masking; 001: Mask ADD2[1], only compare ADD2[7:2]; 010: Mask ADD2[2:1], only compare ADD2[7:3]; 011: Mask ADD2[3:1], only compare ADD2[7:4];

				100: Mask ADD2[4:1], only compare ADD2[7:5]; 101: Mask ADD2[5:1], only compare ADD2[7:6]; 110: Mask ADD2[6:1], only compare ADD2[7]; 111: Mask ADD2[7:1], no comparison, respond to all received 7-bit addresses (except reserved bits). Note: When OA2MSK is not 0, even if the comparison matches, no response will be given to reserved I2C addresses (0b0000xxx and 0b1111xxx).
7:1	ADD2 [7:1]	RW	0	Bits 7:1 of interface address bits 7:1 of address in dual addressing mode
0	ENDUAL	RW	0	Dual addressing mode enable 0: In 7-bit address mode, only OAR1 is recognized; 1: In the 7-bit address mode, both OAR1 and OAR2 are recognized.

30.8.5. I2C Data register (I2C_DR)

Offset address: 0x10

Reset value : 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	DR[7:0]	RW	0	8-bit data register. Internally, the chip has two separate buffers sharing one address, used for storing received data (RX_DR) and placing data to be transmitted to the bus (TX_DR). Transmitter mode: Byte transmission starts automatically when a byte is written in the DR (TX_DR actually) register. A continuous transmit stream can be maintained if the next data to be transmitted is put in DR once the transmission is started (TxE=1). Receiver mode: The received byte is stored in the DR register (actually RX_DR) (RxNE=1). A continuous transmit stream can be maintained if DR is read before the next data byte is received (RxNE=1). Notes: 1) In slave mode, the address is not stored in the data register DR. 2) Write collision is not managed (DR can be written if TxE=0). 3) If an ARLO event occurs during the processing of the ACK pulse, the received byte will not be stored in the data register and thus cannot be read.

30.8.6. I2C status register (I2C_SR1).

Offset address: 0x14

Reset value : 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMBALERT	TIMEOUT	Res.	PECERR	OVRR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	ADD10	BTFF	ADDR	SB
RC_W0	RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	R	R		R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	SMBALERT	RC_W0	0	SMBus alert SMBus host mode: 0: No SMBus reminder; 1: Generates an SMBus alert event on the SMBA pin; SMBus slave mode:

				0: No response to the address header sequence on SMBA; 1: Receives a response to the address header sequence on SMBA until the SMBA pin goes low. Cleared by software writing 0, or by hardware when PE=0. Note: If SMBUS functionality is not supported, this register is fixed at 0.
14	TIMEOUT	RC_W0	0	Timeout or Tlow error. 0: No timeout error; 1: SCL has remained low for 25ms (timeout); or the master's cumulative clock stretching time for low level exceeds 10ms (Tlow:mext); or the slave's cumulative clock stretching time for low level exceeds 25ms (Tlow:sext); or Tidle timeout occurs; or timeout is reached when waking from stop low-power mode. When set in slave mode: The slave resets communication, and the hardware releases the bus; When set in master mode: The hardware generates a stop bit. Cleared by software writing 0, or by hardware when PE=0. Note: This function is only effective in SMBUS mode and when waking from stop low-power mode in I2C mode.
13	Reserved	-	-	Reserved
12	PECERR	RC_W0	0	PEC Error in reception 0: no PEC error: receiver returns ACK after PEC reception (if ACK=1) 1: PEC error: receiver returns NACK after PEC reception (whatever ACK) Cleared by software writing 0, or by hardware when PE=0.
11	OVR	RC_W0	0	Overflow/underflow flag. 0: No overflow/underflow; 1: Overflow/underflow has occurred. Set by hardware in slave mode when NOSTRETCH=1 and: In reception when a new byte is received (including ACK pulse) and the DR register has not been read yet. New received byte is lost. In transmission when a new byte should be sent and the DR register has not been written yet. The same byte is sent twice. Cleared by software writing 0, or by hardware when PE=0. Note: If the DR write occurs very close to SCL rising edge, the sent data is unspecified and a hold timing error occurs.
10	AF	RC_W0	0	Acknowledge failure 0: No acknowledge failure 1: Acknowledge failure Set by hardware when no acknowledge is returned. Cleared by software writing 0, or by hardware when PE=0.
9	ARLO	RC_W0	0	Arbitration loss (master mode). 0: No Arbitration Lost detected 1: Arbitration Lost detected When the interface loses control to another master on the bus, the hardware sets this register. Cleared by software writing 0, or by hardware when PE=0. After an ARLO event the interface switches back automatically to Slave mode (MSL=0). Note: In SMBUS mode, arbitration for data in slave mode only occurs during data interpretation or ACK transmission intervals (excluding address ACK).
8	BERR	RC_W0	0	Bus error 0: No start or stop bit error; 1: Start or stop bit error.

				When the interface detects an incorrect start or stop bit, the hardware sets this bit to 1. Cleared by software writing 0, or by hardware when PE=0.
7	TxE	R	0	Data register empty (transmitters) 0: Data register not empty 1: Data register empty Set when DR is empty in transmission. TxE is not set during address phase. Writing data to the DR register clears this bit, or it is automatically cleared by hardware after generating a start or stop bit, or when PE=0. TxE is not set if either a NACK is received, or if next byte to be transmitted is PEC (PEC=1) Note: TxE is not cleared by writing the first data being transmitted, or by writing data when BTF is set, as in both cases the data register is still empty.
6	RxNE	R	0	Data register not empty (receivers) 0: Data register empty 1: Data register not empty Set when data register is not empty in receiver mode. RxNE is not set during address phase. Cleared by software reading or writing the DR register or by hardware when PE=0. Note: RxNE is not cleared by reading data when BTF is set, as the data register is still full.
5	Reserved	-	-	Reserved
4	STOPF	R	0	Stop detection (slave mode) 0: No Stop condition detected 1: Stop bit detected. After an ACK (if ACK=1), when the slave detects a stop bit on the bus, the hardware sets this bit to 1. Cleared by software reading the I2C_SR1 register followed by a write in the I2C_CR1 register, or by hardware when PE=0. Note: The STOPF bit is not set after a NACK reception.
3	ADD10	R	0	10-bit address header sequence has been transmitted (master mode). 0: No ADD10 event occurred. 1: The master has sent out the first address byte. In 10-bit addressing mode, this bit is set by hardware when the master has sent out the first byte. This bit is cleared by writing to the I2C_DR register after software reads the I2C_SR1 register; or cleared by hardware when PE=0. Note 1: This register is not set after receiving a NACK. Note 2: If the 10-bit address function is not supported, this register is fixed at 0.
2	BTF	R	0	Byte transfer complete flag 0: Data byte transfer not done 1: Byte transfer complete This register is set by hardware under the following conditions (in slave mode when NOSTRETCH=0; in master mode, regardless of NOSTRETCH): — In reception when a new byte is received (including ACK pulse) and DR has not been read yet (RxNE=1). — In transmission when a new byte should be sent and DR has not been written yet (TxE=1). This bit is cleared by reading or writing to the data register after software reads the I2C_SR1 register; or by sending a START or STOP bit; or cleared by hardware when PE=0. Note: The BTF bit is not set after receiving a NACK. If the next byte to be transmitted is PEC (I2C_SR2.TRA=1, I2C_CR1.PEC=1), the BTF bit will not be set.
1	ADDR	R	0	Address has been sent (master mode)/address matched (slave mode).

				This bit is cleared by reading the I2C_SR2 register after software reads the I2C_SR1 register; or cleared by hardware when PE=0. Address matched (slave): 0: Address mismatched or not received. 1: Received address matched. This bit is set by hardware when the received slave address matches the OAR register, or a general call occurs, or an SMBus device receives the default address, or an SMBus host recognizes an SMBus alert. Note: In slave mode, a complete clearing sequence is recommended—after ADDR is set, first read the I2C_SR1 register, then read the I2C_SR2 register. Address sent (Master): 0: No end of address transmission 1: End of address transmission —In 10-bit addressing mode, set after receiving the ACK for the second byte of the address; —In 7-bit addressing, set after receiving the ACK byte. Note: ADDR is not set after a NACK reception.
0	SB	R	0	Start bit (Master mode) 0: No Start condition 1: Start condition generated. —Set when a START bit is transmitted. —This bit is cleared by writing to the I2C_DR register after software reads the I2C_SR1 register; or cleared by hardware when PE=0.

30.8.7. I2C status register 2 (I2C_SR2)

Offset address: 0x18

Reset value : 0x0000_0000

Note: Even if the ADDR flag is set after reading the I2C_SR1 register, reading the I2C_SR2 register after reading I2C_SR1 will clear the ADDR flag. Consequently, I2C_SR2 must be read only when ADDR is found set in I2C_SR1 or when the STOPF bit is cleared.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res.	Res.	Res.	Res.	Res	Res	Res.	Res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PEC[7:0]								DUAL F	SMBHOS T	SMBDEFAUL T	GENCAL L	Res	TR A	BUS Y	MS L
R	R	R	R	R	R	R	R	R	R	R	R		R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:8	PEC[7:0]	R	0	Packet error checking register. This register contains the internal PEC when ENPEC=1.
7	DUALF	R	0	Dual address flag (slave mode). 0: Received address matched with OAR1; 1: Received address matched with OAR2. The hardware clears this register when a stop bit or a repeated start bit is generated, or when PE=0. Note: If dual address functionality is not supported, this register is fixed at 0.
6	SMBHOST	R	0	SMBus host header sequence received flag (slave mode). 0: No SMBus Host address; 1: SMBus host address received when SMBTYPE=1 and ENARP=1. The hardware clears this register when a stop bit or a repeated start bit is generated, or when PE=0. Note: If SMBUS functionality is not supported, this register is fixed at 0.
5	SMBDEFAULT	R	0	SMBus slave default address (slave mode). 0: No SMBus device default address; 1: SMBus device default address received when ENARP=1. The hardware clears this register when a stop bit or a repeated start bit is generated, or when PE=0.

				Note: If SMBUS functionality is not supported, this register is fixed at 0.
4	GENCALL	R	0	General call address (slave mode). 0: No General Call; 1: General Call Address received when ENGC=1. The hardware clears this register when a stop bit or a repeated start bit is generated, or when PE=0.
3	Reserved	-	-	Reserved
2	TRA	R	0	Transmit/receive flags 0: Data bytes received 1: Data bytes transmitted This bit is set depending on the R/W bit of the address byte, at the end of total address phase. The hardware clears this register when a stop bit is detected (STOPF=1), or a repeated start bit, or bus arbitration loss (ARLO=1), or when PE=0.
1	BUSY	R	0	Bus busy 0: No communication on the bus 1: Communication ongoing on the bus Set by hardware on detection of SDA or SCL low Cleared by hardware when a stop bit is detected. This register indicates the ongoing bus communication and is still updated when the interface is disabled (PE=0).
0	MSL	R	0	Master/slave 0: Slave mode 1: Host mode — Set by hardware when the interface is in master mode (SB=1); — Cleared by hardware when a stop bit is detected on the bus (STOPF=1), arbitration loss (ARLO=1), or when PE=0.

30.8.8. I2C clock control register (I2C_CCR)

Offset address: 0x1C

Reset value : 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F/S	DUTY	F +	Res.	CCR[11:0]											
RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	F/S	RW	0	I2C master mode selection 0: Sm mode I2C 1: Fm mode I2C
14	DUTY	RW	0	Fm mode duty cycle 0: Fm mode tlow/thigh = 2 1: Fm mode tlow/thigh = 16/9
13	F +	RW	0	I2C Fm+ master mode selection. 0: Standard mode or fast mode, the specific selection is determined by bit15; 1: Fast mode enhancement; Note: When configuring this register to 1, the value of bit15 is irrelevant;
12	Reserved	-	-	Reserved
11:0	CCR[11:0]	RW	0	Clock control division coefficient in fast/standard mode (master mode). This division coefficient is used to set the SCL clock in master mode. - Standard mode or SMBus mode: - Thigh=CCR x Tpclk - Tlow =CCR x Tpclk - Fast mode: - DUTY=0: - Thigh=CCR x Tpclk

				- $T_{low} = 2 \times CCR \times T_{pclk}$ - DUTY=1: - $T_{high} = 9 \times CCR \times T_{pclk}$ - $T_{low} = 16 \times CCR \times T_{pclk}$ ■ When fast mode enhancement is enabled: - DUTY=0: - $T_{high} = 3 \times CCR \times T_{pclk}$ - $T_{low} = 5 \times CCR \times T_{pclk}$ - DUTY=1 (to achieve 1 MHz): - $T_{high} = 2 \times CCR \times T_{pclk}$ - $T_{low} = 3 \times CCR \times T_{pclk}$ Notes: ■ 1. The minimum allowed value is 0x04, except in DUTY mode where the minimum allowed value is 0x01. $T_{high} = t_r(SCL) + t_w(SCLH)$ $T_{low} = t_r(SCL) + t_w(SCLL)$ ■ These delays have no filters ■ This register can only be configured when PE=0 ■ fCK should be an integer multiple of 10 MHz to correctly generate a 400 kHz waveform

Note: bit15 and bit13 combine to define the extended mode as shown in the table below:

F +	F/S	Mode
0	0	Standard mode
0	1	Fast mode
1	0	Fast mode enhancement
1	1	Fast mode enhancement

30.8.9. I2C TRISE register (I2C_TRISE)

Offset address: 0x20

Reset value : 0x0000_0082

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	THOLDDATA_SEL	THOLDDATA[4:0]				TRISE[6:0]							
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	THOLDDATA_SEL	RW	0	Data retention time selection 0: Default hardware calculation data retention time 1: Configure data retention time through THLDDATA
11:7	THLDDATA[4:0]	RW	1	Maximum data hold time in Standard Mode/Fast Mode/Fast Mode Plus (transmit mode). These bits define the minimum hold time to ensure data retention in data transmit mode. Example: The maximum allowed SDA fall time in Standard Mode is 300ns. If FREQ[6:0] in I2C_CR2 register equals 0x08 with $T_{pclk} = 125$ ns, configure TRISE as 0x03 ($300ns/125ns = 2.4 + 1$). If the result is not an integer, write the integer part to THLDDATA to ensure proper configuration.
6:0	TRISE	RW	0x2	Maximum rise time in Standard/Fast Mode (master mode). These bits should provide the maximum duration of the SCL feedback loop in master mode. The purpose is to keep a stable SCL frequency whatever the SCL rising edge duration. These bits must be programmed with the maximum SCL rise time given in the I2C bus specification, incremented by 1. For instance: in Sm mode, the maximum allowed SCL rise time is 1000 ns. If FREQ[6:0] in I2C_CR2 register equals 0x08 with $T_{PCLK} = 125ns$, configure TRISE as 0x09 ($1000ns/125ns = 8 + 1 = 9$). The filter value can also be added to TRISE[5:0].

				If the result is not an integer, TRISE[5:0] must be programmed with the integer part, in order to respect the tHIGH parameter. Note: TRISE[5:0] must be configured only when the I2C is disabled (PE = 0).

30.8.10. I2C TIMEOUT Register (I2C_TIMEOUTR)

Offset address: 0x24

Reset value : 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEXTEN	Res.	Res.	Res.	TIMEOUTB [11: 0]											
RW				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMEOUTEN	Res.	Res.	TIDLE	TIMEOUTA [11: 0]											
RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	TEXTEN	RW	0	Clock signal extension timeout enable 0: Disable clock signal extension timeout detection. 1: Enable clock stretching timeout detection. A timeout error (TIMEOUT=1) is detected when the accumulated SCL stretching time exceeds LOW:EXT on the I2C interface.
30:28	Reserved	-	-	Reserved
27:16	TIMEOUTB [11: 0]	RW	0	Bus timeout B This field is used to configure the cumulative clock extension timeout: In master mode, the accumulated clock low stretching time (tLOW:MEXT) of the master device is detected. In slave mode, the accumulated clock low stretching time (tLOW:SEXT) of the slave device is detected. $tLOW:EXT = (TIMEOUTB + 1) \times 2048 \times tPCLK$ Note: These bits can only be written when TEXTEN = 0
15	TIMEOUTEN	RW	0	Clock timeout enable 0: Disable SCL timeout detection. 1: Enable SCL timeout detection: A timeout error (TIMEOUT=1) will be detected when the low-level duration of SCL exceeds tTIMEOUT (TIDLE=0), or when the high-level duration of SCL exceeds tIDLE (TIDLE=1).
14:13	Reserved	-	-	Reserved
12	TIDLE	RW	0	Idle clock timeout detection 0: TIMEOUTA is used to detect SCL low-level timeout 1: TIMEOUTA is used to detect SCL and SDA high-level timeout (bus idle condition) Note: This bit can only be written when TIMOUTEN=0.
11: 0	TIMEOUTA [11: 0]	RW	0	TIMEOUTA[11:0]: Bus Timeout A This field is used to configure: – SCL low-level timeout condition tTIMEOUT (when TIDLE=0), $tTIMEOUT = (TIMEOUTA + 1) \times 2048 \times tPCLK$ – Bus idle condition, i.e., SCL and SDA high-level (when TIDLE=1), $tIDLE = (TIMEOUTA + 1) \times 4 \times tPCLK$ Note: These bits can be written only when TIMOUTEN=0.

31. Universal Synchronous/Asynchronous Receiver/Transmitter (USART)

This project implements 2 USART modules with identical functionality. Both USART1 and USART2 support LIN, S7816, and IrDA.

31.1. USART Introduction

The Universal Synchronous/Asynchronous Receiver/Transmitter (USART) enables flexible full-duplex data exchange with external devices, meeting the requirements of industrial-standard NRZ asynchronous serial data formats. The USART utilizes a fractional baud rate generator to provide a wide range of baud rate options.

USART not only supports synchronous unidirectional communication and half-duplex single-wire communication but also LIN (Local Interconnect Network), smart card protocol, IrDA (Infrared Data Association), SIR ENDEC specifications, modem operations (CTS/RTS), and multi-processor communication.

High-speed data communication can be achieved by using the DMA method of the multi-buffer configuration.

31.2. USART main features

- Full-duplex asynchronous communication
- NRZ standard format
- Configurable 16x or 8x oversampling to achieve the optimal balance between speed tolerance and clock tolerance
- Programmable baud rate shared by transmission and reception, up to 4.5Mbit/s (72MHz, 16x oversampling)
- Auto baud rate detection
- Programmable data length of 8 or 9 bits
- Programmable data sequence, shifting MSB or LSB first
- Configurable stop bits (0.5/1/1.5/2 bits)
- Synchronous master/slave mode and clock output/input for synchronous communication
- Single-wire Half-duplex communications
- Receive/transmit bytes by DMA buffer
- Independent transmit and receive enable bits
- Tx/Rx pin configurable swap
- RS232 hardware flow control
- Transfer detection flag
 - Receive buffer full
 - Transmit buffer empty
 - End of transmission flags
- Parity control
 - Transmit parity bit
 - Check the received data byte
- LIN master mode sync break transmission and LIN slave mode break detection
 - Generates 13-bit break character and detects 10/11-bit stop bits when USART is configured for LIN
- Supports IrDA SIR encoder/decoder with 3/16-bit duration in normal mode
- Smart card mode function
 - Smart card interface supports ISO7816-3 standard protocols, including T=0 and T=1 asynchronous protocols
 - Supports 0.5 and 1.5 stop bits in smart card operation mode
 - Configurable NACK signal width (1/1.5/2 ETU)
 - Support transmission error retransmit
 - Supports EGT setting (0~256) with timeout interrupt
- Flagged interrupt sources
 - Transmit data register empty
 - CTS change
 - Transmission complete
 - Receive register not empty
 - Overrun error
 - Bus idle detected

- Framing error
- Noise operation
- Error detection
- Receive timeout
- End of block
- LIN break
- Multiprocessor communication
 - enter into mute mode if address match does not occur.
 - Wakeup from mute mode: via idle detection or address mark detection

Puya Confidential

31.3. USART functional description

31.3.1. USART block diagram

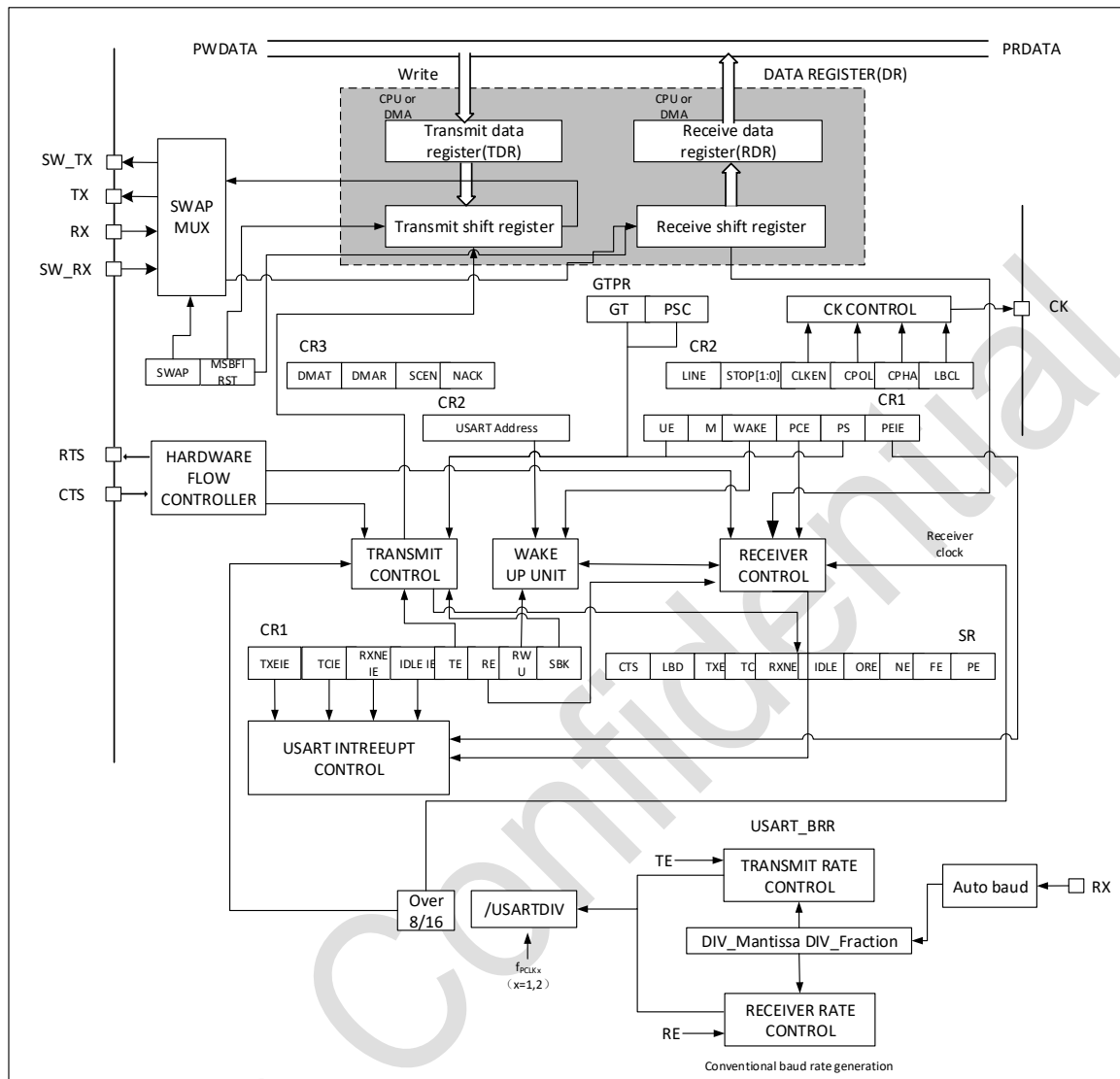


Figure 31-1 USART block diagram

31.3.2. USART signals

31.3.2.1. Bidirectional communication

USART bidirectional communication requires at least two pins: Receive Data Input (RX) and Transmit Data Output (TX)

- RX: Serial data input for reception

Uses oversampling to distinguish data from noise for data recovery.

- TX: Serial data output for transmission

When the transmitter is disabled, the output pin returns to its IO port configuration. When the transmitter is enabled and nothing is to be transmitted, the TX pin is at high level. In single-wire and smartcard mode, this IO is used to transmit and receive the data.

31.3.2.2. RS232 Hardware flow control mode

The following pins are required in RS232 hardware flow control mode:

- CTS (Clear To Send)

When driven high, this signal blocks the data transmission at the end of the current transfer.

- RTS (Request To Send)

When low, this signal indicates the USART is ready to receive data.

31.3.2.3. Synchronous master mode and smartcard mode

The following pins are required in synchronous master mode and smartcard mode:

- CK

This pin serves as clock output in synchronous master mode and smartcard mode, and as clock input in synchronous slave mode.

In synchronous master mode, this pin outputs the transmitter data clock for synchronous transmission in host mode (no clock pulses on start and stop bits, software can be configured to send clock pulses to the last data bit). Data can be received synchronously on the RX pin. This mechanism can be used to control peripherals with shift registers (e.g., LCD drivers). The clock phase and polarity are software programmable.

In smartcard mode, CK outputs the clock to the smartcard.

■ NSS

This pin acts as Slave Select input in Synchronous slave mode.

Note: NSS and CTS share the same pin.

31.3.3. USART Character Description

The word length can be configured as 8-bit or 9-bit data by programming the M bit in the USART_CR1 register. The TX pin is in low state during the start bit. It is in high state during the stop bit.

An idle character is considered a complete data frame consisting entirely of '1's (the stop bit level is also '1').

A break character is considered as receiving all '0's during a frame period (including the stop bit period, which is also '0'). At the end of the break frame, the transmitter inserts 1 or 2 stop bits ('1') in response to the start bit.

Transmission and reception are driven by a common baud rate generator, which generates the transmit clock and receive clock when the transmitter and receiver enable bits are set, respectively.

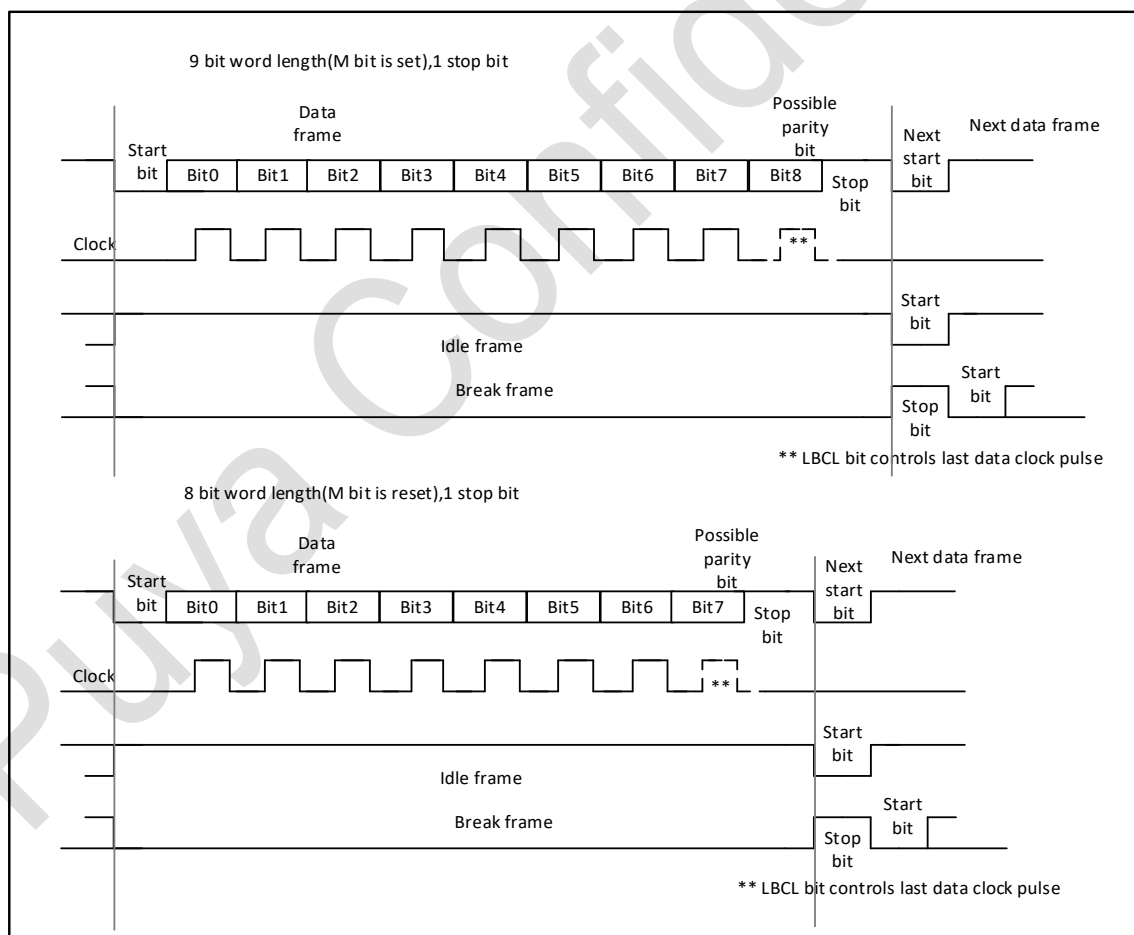


Figure 31-2 word length programming

31.3.4. USART transmitter

The transmitter sends 8-bit or 9-bit data words based on the status of the USART_CR1.M register bit. When the transmit enable bit (TE) is set, the data in the transmit shift register is output on the TX pin and the corresponding clock pulse is output on the CK pin.

31.3.4.1. Character transmission

During a USART transmission, data shifts out the least significant bit first on the TX pin. In this mode, the USART_DR register consists of a buffer (TDR) between the internal bus and the transmit shift register.

Each character is preceded by a start bit with a one-cycle low level; characters are terminated by a configurable number of stop bits. USART supports multiple stop bit configurations: 0.5, 1, 1.5, and 2 stop bits.

Note:

The TE bit should not be reset during transmission of data. Resetting the TE bit during the transmission will corrupt the data on the TX pin as the baud rate counters will get frozen. The current data being transmitted will be lost.

An idle frame will be sent after the TE bit is enabled.

31.3.4.2. Configure stop bits

The number of stop bits to be transmitted with every character can be programmed in Control register 2, bits 13, 12.

- 1) 1 stop bit: The default value for the number of stop bits.
- 2) 2 stop bits: Can be used in normal USART mode, single-wire mode, and modem mode.
- 3) 0.5 stop bits: Used when receiving data in smartcard mode.
- 4) 1.5 stop bits: Used when sending and receiving data in smart card mode.

Idle frames include stop bits.

Break frame is 10 bits of low level followed by a stop bit (when USART_CR1.M=0); or 11 bits of low level followed by a stop bit (when USART_CR1.M=1). It is not possible to transmit longer broken frames (longer than 10 or 11 bits in length).

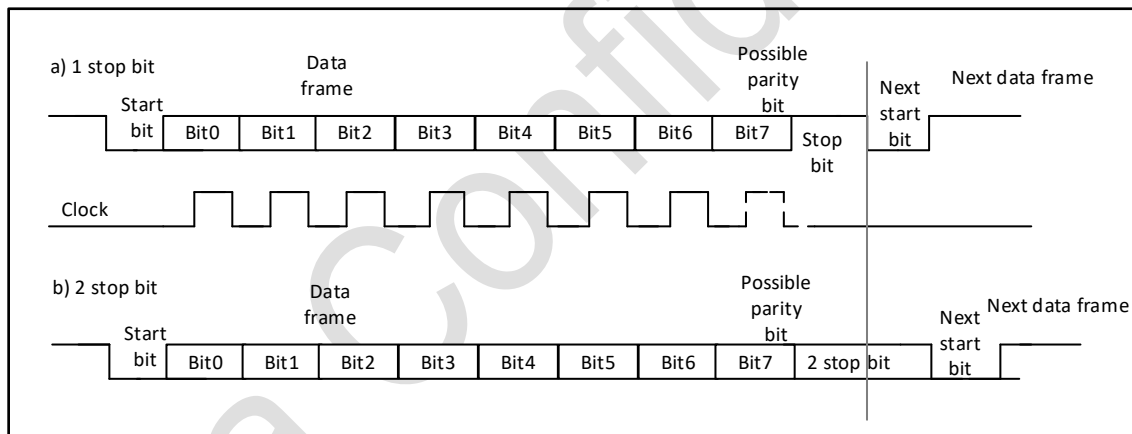


Figure 31-3 Configurable stop bits

Procedure:

- 1) Enable the USART by writing the UE bit in USART_CR1 register to 1.
- 2) Program the M bit in USART_CR1 to define the word length.
- 3) Program the number of stop bits in the STOP register of USART_CR2.
- 4) Select DMA enable (DMAT) in USART_CR3 if multibuffer communication must take place. Configure the DMA registers as described in "Multibuffer Communication".
- 5) Select the desired baud rate using the USART_BRR register.
- 6) Set the TE bit in USART_CR1 to send an idle frame as first transmission.
- 7) Write the data to send in the USART_DR register (this clears the TXE bit). Repeat this for each data to be transmitted in case of single buffer.
- 8) After writing the last data into the USART_DR register, wait until TC=1. This indicates that the transmission of the last frame is complete. Before disabling the USART or entering stop mode (e.g., entering low-power mode), ensure the transmission is complete to avoid corrupting the last transmission.

31.3.4.3. Single-byte communication

Writing to the data register clears the TXE bit. The TXE bit is set by hardware, indicating:

- The data has been moved from TDR to the shift register and the data transmission has started.
- The TDR register is empty.

■ The next data can be written in the USART_DR register without overwriting the previous data.

If the TXEIE bit is set, the TXE flag will generate an interrupt.

If the USART is currently transmitting data, writing to the USART_DR register stores the data in the TDR buffer and copies it into the shift register at the end of the current transmission.

When no transmission is taking place, a write instruction to the USART_DR register places the data directly in the shift register, the data transmission starts, and the TXE bit is immediately set.

If a frame is transmitted (after the stop bit) and the TXE bit is set, the TC bit goes high. An interrupt is generated if the TCIE bit is set in the USART_CR1 register.

After writing the last data word to the USART_DR register, wait for TC=1 before disabling the USART module or setting the microcontroller to low-power mode (see the figure below for details).

Software clears the TC bit using the following procedure:

1. A read from the USART_SR register;
2. A write to the USART_DR register.

Note: The TC bit can also be cleared by writing '0' to it via software. This clearing sequence is recommended only for Multibuffer communication.

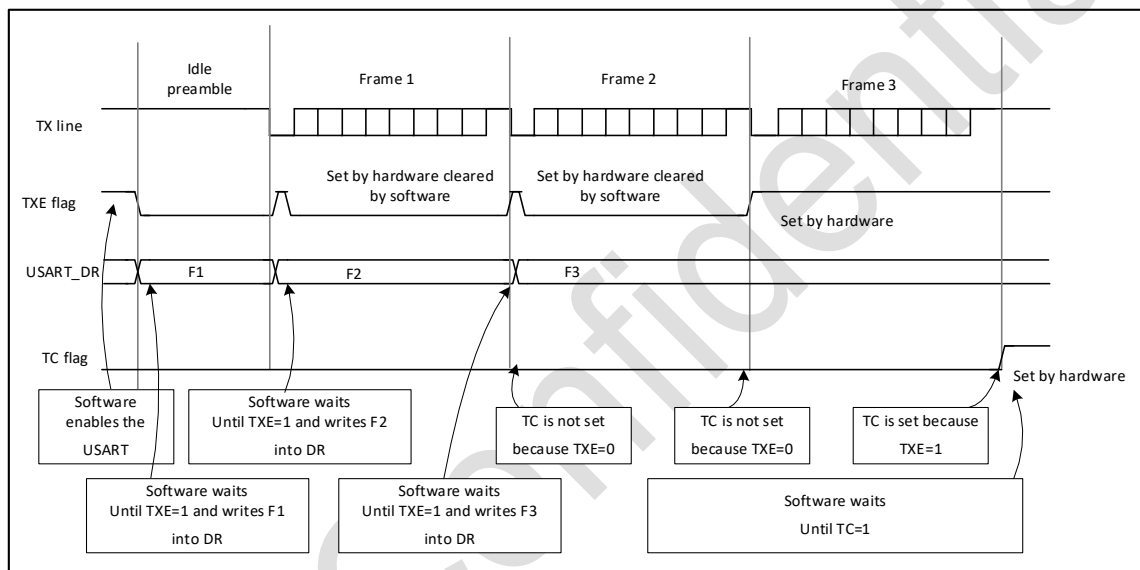


Figure 31-4 TC/TXE behavior during transmission

31.3.4.4. Disconnect character

Setting SBK sends a break character frame. The break frame length depends on the Mbit. If SBK = 1 is set, a disconnect character will be sent on the TX line after the current data transmission is completed. The SBK is reset by hardware upon completion of the disconnect character transmission (at the disconnect of the stop bit of the symbol). The USART inserts a logic '1' at the end of the last break frame to ensure recognition of the next frame's start bit.

Note: If the SBK bit is reset by software before starting to transmit the break frame, the break character will not be sent. To send two consecutive break frames, the SBK bit should be set after the stop bit of the previous break character.

31.3.4.5. Idle characters

Setting TE will cause the USART to send an idle frame before the first data frame.

31.3.5. USART Receiver

The USART can receive data words of either 8 or 9 bits depending on the M bit in the USART_CR1 register.

31.3.5.1. Start bit detection

In the USART, the start bit is detected when a specific sequence of samples is recognized. The sequence is 1 1 1 0 X 0 X 0 X 0 0 0

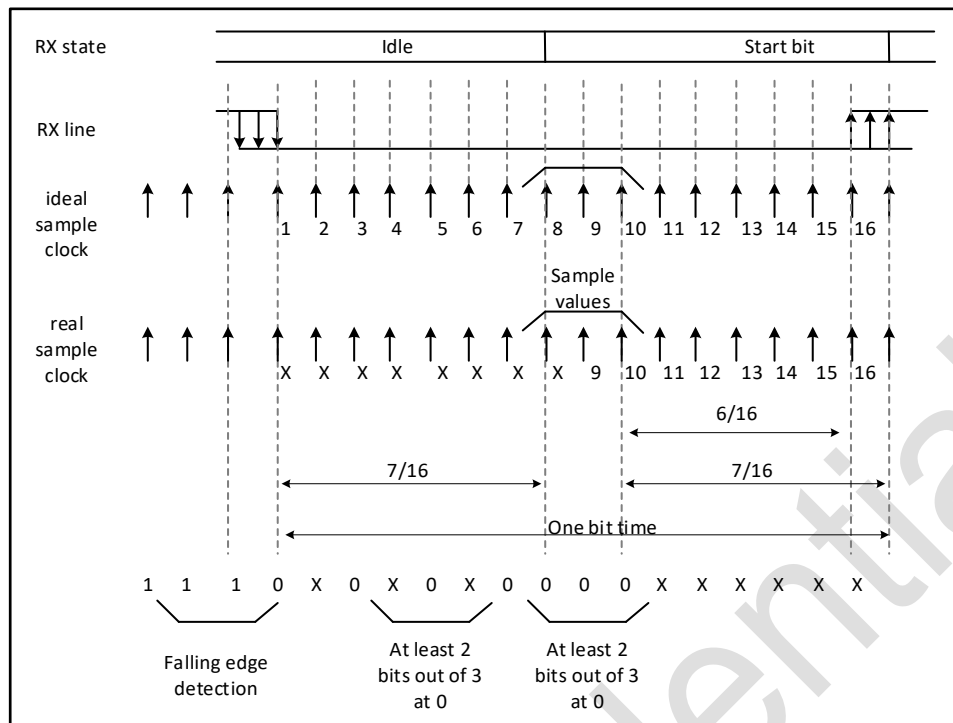


Figure 31-5 Start Bit Detection

If the sequence is incomplete, the receiving end will exit the start bit detection and return to the idle state (without setting the flag bit) waiting for the falling edge. If all three sample points are '0' (first sampling at bits 3, 5, 7, and second sampling at bits 8, 9, 10), the start bit is confirmed, and the RXNE flag is set. If RXNEIE=1, an interrupt is generated.

If during two samplings, only two out of three sample points are '0' (sampling points at bits 3, 5, 7 or bits 8, 9, 10), the start bit is still valid, but the NE noise flag is set. If this condition is not met, the start detection aborts and the receiver returns to the idle state (no flag is set).

If during one sampling, only two out of three sample points are '0' (sampling points at bits 3, 5, 7 or bits 8, 9, 10), the start bit is still valid, but the NE noise flag is set.

31.3.5.2. Character reception

During a USART reception, data shifts in least significant bit first through the RX pin. In this mode, the USART_DR register consists of a buffer (RDR) between the internal bus and the received shift register.

Procedure:

1. Enable the USART by writing the UE bit in USART_CR1 register to 1.
2. Program the M bit in USART_CR1 to define the word length.
3. Program the number of stop bits in the STOP register of USART_CR2.
4. For multi-buffer communication, select the DMA enable bit (DMAR) in USART_CR3. Configure the DMA register as explained in multibuffer communication.
5. Select the desired baud rate using the baud rate register USART_BRR
6. Set the RE bit USART_CR1. Activate the receiver to begin detecting the start bit.

When a character is received:

- The RXNE bit is set, indicating that the contents of the shift register have been transferred to the RDR. This means the data has been received and can be read (including any associated error flags).
- An interrupt is generated if the RXNEIE bit is set.
- The error flags can be set if a frame error, noise or an overrun error has been detected during reception.
- In multi-buffer communication, RXNE is set after each byte is received. It is cleared when the DMA reads the Receive data Register.
- In single-buffer mode, reading the USART_DR register by software clears the RXNE bit. The RXNE flag can also be cleared by writing 0 to it. The RXNE bit must be cleared before the end of the reception of the next character to avoid an overrun error.

Note: The RE bit should not be reset while receiving data. If the RE bit is disabled during reception,

the reception of the current byte will be aborted.

31.3.5.3. Disconnect character

When a break character frame is received, the USART processes it as a frame error.

31.3.5.4. Idle characters

When an idle frame is detected, it is processed similarly to a normal data frame, but an interrupt is generated if the IDLEIE bit is set.

31.3.5.5. Overrun error

An overrun error occurs when a character is received when RXNE has not been reset. Data can not be transferred from the shift register to the RDR register until the RXNE bit is cleared. The RXNE flag is set after each byte is received.

When the RXNE flag is set, an overflow error occurs if the next data is received or the previous DMA request has not been processed.

When an overrun error occurs:

- The ORE bit is set.
- The RDR content will not be lost. The previous data is available when a read to USART_DR is performed.
- The shift register will be overwritten. After that point, any data received during overrun is lost.
- An interrupt is generated if RXNEIE or EIE is set.
- The ORE bit is reset by a read to the USART_SR register followed by a USART_DR register read operation.

Note: The ORE bit, when set, indicates that at least 1 data has been lost. There are two possibilities:

- If RXNE=1, then the last valid data is stored in the receive register RDR and can be read.
- If RXNE=0, this means the last valid data has been read, and there is no data left to read in RDR. This may occur when new (i.e. missing) data is received while the last valid data is read in the RDR. This may also occur when new data is received during a read sequence (between a USART_SR register read access and a USART_DR read access).

31.3.5.6. Noise error

Over-sampling techniques are used (except in synchronous mode) for data recovery by discriminating between valid incoming data and noise.

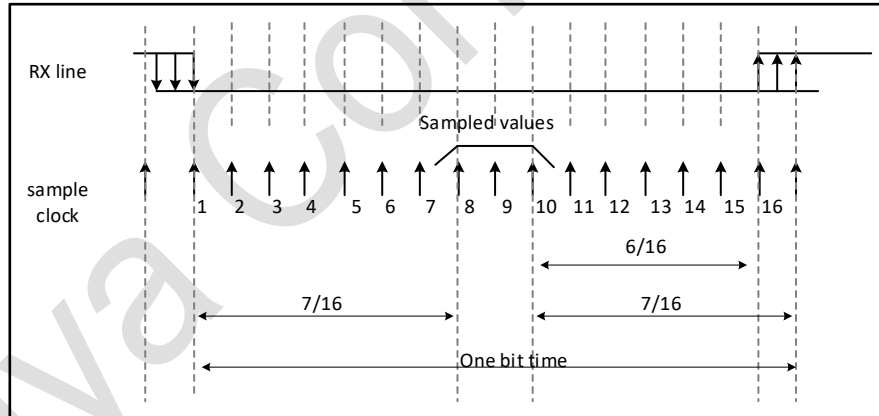


Figure 31-6 Data sampling during noise detection

Table 31-1 Data sampling for noise detection

Sampled value	NE status	Received bit	Data validity
000	0	0	Valid
001	1	0	Not valid
010	1	0	Not valid
011	1	1	Not valid
100	1	0	Not valid
101	1	1	Not valid
110	1	1	Not valid
111	0	1	Valid

When noise is detected in a frame:

- The NE is set at the rising edge of the RXNE bit.
- The invalid data is transferred from the Shift register to the USART_DR register.
- No interrupt is generated in case of single byte communication. However, since the NE flag and RXNE flag are set simultaneously, RXNE will generate an interrupt. In multi-buffer communication, if the EIE bit in the USART_CR3 register is set, an interrupt will be generated.

- The NE bit is reset by a USART_SR register read operation followed by a USART_DR register read operation.

31.3.5.7. Framing error

A framing error is detected when:

- A framing error is detected when the stop bit is not recognized on reception at the expected time, following either a de-synchronization or excessive noise.
- When the framing error is detected:
 - The FE bit is set by hardware
 - The invalid data is transferred from the Shift register to the USART_DR register.
 - No interrupt is generated in case of single byte communication. However, this bit rises at the same time as the RXNE bit which itself generates an interrupt. In case of multibuffer communication an interrupt is issued if the EIE bit is set in the USART_CR3 register.
 - The FE bit is reset by a read to the USART_SR register followed by a USART_DR register read operation.

31.3.5.8. Configure stop bits on receive

The number of stop bits to be received can be configured through the control bits of USART_CR: it can be either 1 or 2 in normal mode and 0.5 or 1.5 in Smartcard mode.

- 0.5 stop bits (when receiving in smartcard mode): The 0.5 stop bits are not sampled. As a result, when 0.5 stop bits are selected, frame errors and break frames cannot be detected.
- 1 stop bit: sampling for 1 stop bit is done on the 8th, 9th and 10th samples.
- 1.5 stop bits (smartcard mode): When transmitting in smartcard mode, the device must check whether the data has been transmitted correctly. Therefore, the receiver block must be activated (RE=1 in the USART_CR1 register), and the signal on the data line must be sampled during the transmission of the stop bits. If a parity error occurs, the smartcard will pull the data line low when the sender samples the NACK signal, i.e., during the time corresponding to the stop bit on the bus, to indicate a frame error. FE is set together with RXNE at the end of the 1.5 stop bits. Sampling for 1.5 stop bits is done on the 16th, 17th and 18th samples. The 1.5 stop bit can be decomposed into 2 parts: one 0.5 baud clock period during which nothing happens, followed by 1 normal stop bit period during which sampling occurs halfway through.
- 2 stop bits: sampling for 2 stop bits is done on the 8th, 9th and 10th samples of the first stop bit. The framing error flag is set if a framing error is detected during the first stop bit. The second stop bit is not checked for framing error. The RXNE flag is set at the end of the first stop bit.

31.3.6. USART baud rate generation

The baud rate for the receiver and transmitter (Rx and Tx) are both set to the value programmed in the USART_BRR register.

$$Tx / Rx \text{ baud} = f_{CK} / (16 * USARTDIV)$$

Here, f_{CK} is the clock provided to the peripheral, and USARTDIV is an unsigned fixed-point number stored in the USART_BRR register.

Note: The baud counters are updated with the new value of the Baud registers after a write to USART_BRR. Hence, the Baud rate register value should not be changed during communication.

How to derive USARTDIV from USART_BRR register values

Example 1:

If DIV_Mantissa = 27, DIV_Fraction = 12 (USART_BRR=0x1BC),

Then:

Mantissa (USARTDIV) = 27

Fraction (USARTDIV) = $12/16 = 0.75$

Thus, USARTDIV = 27.75

Example 2:

Required USARTDIV = 25.62,

There leads to:

DIV_Fraction = $16 * 0.62 = 9.92$

The nearest integer is: 10 = 0x0A

DIV_Mantissa = mantissa (25.620) = 25 = 0x19

Therefore, USART_BRR = 0x19A

Example 3:

Required USARTDIV = 50.99

There leads to:

$DIV\_Fraction = 16 \times 0.99 = 15.84$

The nearest integer is: $16 = 0x10 \Rightarrow DIV\_frac[3:0]$ overflow $\Rightarrow$ Carry must be added to the fractional part

$DIV\_Mantissa = \text{mantissa} (50.990 + \text{carry}) = 51 = 0x33$

Therefore: $USART\_BRR = 0x330$, $USARTDIV=51$

Table 31-2 Error calculation when setting baud rate

Baud rate		fpcIk = 36MHz			fpcIk = 72MHz		
N o.	kbp s	Actual	Value programmed in the baud rate register	Error%	Actual	Value programmed in the baud rate register	Error%
1	2.4	2.400	937.5	0.00 %	2.4	1875	0.00 %
2	9.6	9.600	234.375	0.00 %	9.6	468.75	0.00 %
3	19.2	19.200	117.1875	0.00 %	19.2	234.375	0.0 %
4	57.6	57.600	39.0625	0.00 %	57.6	78.125	0.0 %
5	115.2	115.385	19.5	0.15 %	115.2	39.0625	0.0 %
6	230.4	230.769	9.75	0.16 %	230.769	19.5	0.16 %
7	460.8	461.538	4.875	0.16 %	461.538	9.75	0.16 %
8	921.6	923.076	2.4375	0.16 %	923.076	4.875	0.16 %
9	2250	2250.000	1	0.00 %	2250	2	0 %
10	4500	Impossible	Impossible	Impossible	4500	1	0 %

Note: The lower the CPU clock the lower will be the accuracy for a particular Baud rate. The upper limit of the achievable baud rate can be fixed with this data.

31.3.7. USART reception tolerance

The USART asynchronous receiver can work properly only when the overall clock system deviation is within the tolerance range that the USART asynchronous receiver can handle. The causes which contribute to the total deviation are:

- DTRA: deviation due to the transmitter error (which also includes the deviation of the transmitter's local oscillator)
- DQUANT: error due to the baud rate quantization of the receiver
- DREC: deviation of the receiver local oscillator
- DTCL: Deviation caused by the transmission line (usually due to the inconsistency between the transition timing from low to high and from high to low in the transceiver).

$DTRA + DQUANT + DREC + DTCL < USART \text{ receiver tolerance}$

The USART receiver can receive data correctly at up to the maximum tolerated deviation specified in Tables below, depending on the following settings:

- 10- or 11-bit character length defined by the M bit in the USART_CR1 register
- 8x or 16x oversampling defined by the OVR8 bit in the USART_CR3 register
- Bits BRR[3:0] of USART_BRR register are equal to or different from 0000.

Table 31-3 USART reception tolerance when $DIV\_Fraction$ is 0

M-bit	OVR8=0		OVR8=1	
	NF is an error	NF doesn't care	NF is an error	NF doesn't care
0	1.2 %	5.05 %	2.43 %	2.52 %
1	1.2 %	4.98 %	2.43 %	2.45 %

Table 31-4 USART reception tolerance when $DIV\_Fraction$ is not 0

M-bit	OVR8=0		OVR8=1	
	NF is an error	NF doesn't care	NF is an error	NF doesn't care
0	1.2 %	5.05 %	2.45 %	2.5 %
1	1.2 %	4.92 %	2.45 %	2.45 %

31.3.8. USART Auto baud rate detection

The USART can detect and automatically set the value of the USART_BRR register based on the

reception of a character. Automatic baud rate detection is useful in the following scenarios:

- The communication speed of the system is not known in advance.
- The system is using a relatively low accuracy clock source and this mechanism allows the correct baud rate to be obtained without measuring the clock deviation.

The clock source frequency must be compatible with the expected communication rate. 16x oversampling must be selected, and chosen between $f_{CK}/65535$ and $f_{CK}/16$.

- When oversampling by 16, the baud rate range is $f_{CK}/65535$ to $f_{CK}/16$.
- When oversampling by 8, the baud rate range is $f_{CK}/65535$ to $f_{CK}/8$.

Before activating automatic baud rate detection, the auto baud rate detection mode must be selected via `ABRMOD[1:0]` in the `USART_CR3` register. There are various modes based on different character patterns.

In these auto baud rate modes, the baud rate is measured several times during the synchronization data reception and each measurement is compared to the previous one.

These modes are the following:

- Mode 0: Any character starting with 1. In this case the USART measures the duration of the start bit (falling edge to rising edge).
- Mode 1: Any character starting with 10xx bits. In this case, the USART measures the duration of the Start and of the 1st data bit. The measurement is performed between falling edge to falling edge to ensure better accuracy in slow signal rise scenarios.
- Mode 2: 0x7F character frame. In this case, the baud rate is first updated at the end of the start bit, then updated again at the end of bit 6.
- Mode 3: 0x55 character frame. In this case, the baud rate is first updated at the end of the start bit, then updated at the end of bit 0, and finally updated at the end of bit 6. Additionally, other checks are performed for each intermediate transition on the RX line.

Furthermore, checks for each RX transmission are also performed in parallel. If the transmission on RX is not sufficiently synchronized with the receiver (the receiver calculates the baud rate based on bit 0), an error will occur.

The automatic baud rate detection is activated by setting the `ABREN` bit in the `USART_CR3` register. The USART will then wait for the first character on the RX line. Setting the `ABRF` flag in the `USART_SR` register indicates the completion of automatic baud rate detection. If the line is noisy, the correct baud rate detection cannot be guaranteed. In this case the `BRR` value may be corrupted and the `ABRE` error flag will be set. If the communication speed is incompatible with the auto baud rate detection range (with 16x oversampling, the bit width is not between 16 and 65535 clock cycles), an `ABRE` error will also occur.

The `RXNE` interrupt indicates the completion of the operation. At any later time, automatic baud rate detection can be restarted by resetting the `ABRF` flag (by writing 0).

Note: If `UE` is cleared during auto baud rate detection, the `BRR` value may be corrupted.

31.3.9. USART multiprocessor communications

It is possible to perform USART multiprocessor communications (with several USARTs connected in a network). For example, a USART device can act as a master, with its TX output connected to the RX input of other USART slaves; the TX outputs of the USART slaves are logically ANDed together and connected to the RX input of the master.

In a multiprocessor configuration, ideally only the intended message recipient actively receives the full message content, thereby reducing redundant USART service overhead caused by all un-addressed receivers.

Unaddressed devices can be placed in mute mode through the mute function. When the Mute mode is enabled:

- Do not set the receive status bit to 1
- Any receive interruption is prohibited
- The `RWU` bit in the `USART_CR1` register is set to 1. `RWU` can be automatically controlled by hardware or written by software under certain conditions

Depending on the state of the `WAKE` bit in the `USART_CR1` register, the USART can enter or exit mute mode in two ways.

- Idle Line detection if the `WAKE` bit is reset,
- Address mark detection if the `WAKE` bit is set.

31.3.9.1. Idle frame detection (`WAKE=0`)

The USART enters mute mode when the `RWU` bit is written to 1. It wakes up when an Idle frame is detected. Then the `RWU` bit is cleared by hardware, but the `IDLE` bit is not set in the `USART_SR` register. `RWU` can also be written to 0 by software. The following figure shows an example of using idle bus detection to wake up and enter mute mode.

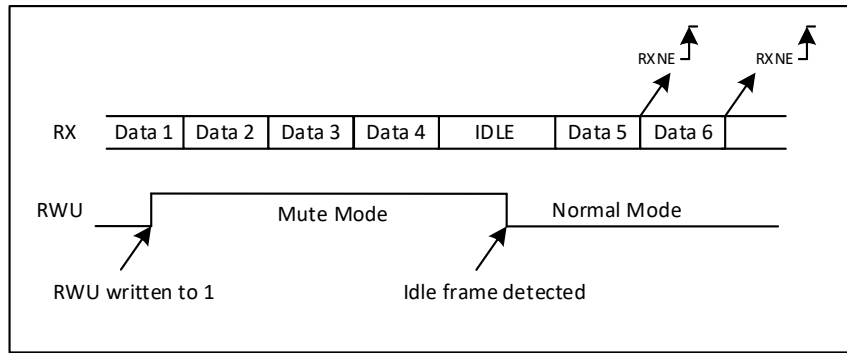


Figure 31-7 Mute Mode Using Idle Frame Detection

31.3.9.2. Address mark detection (WAKE=1)

In this mode, if the MSB of a byte is 1, the byte is considered an address; otherwise, it is considered data. In an address byte, the address of the targeted receiver is put in the 4 LSBs. This 4-bit address is compared by the receiver with its own address, which is programmed in the ADD[3:0] bits of the USART_CR2 register.

The USART enters mute mode when an address character is received which does not match its programmed address. In this case, the RWU bit is set by hardware.

The RXNE flag is not set for this address byte and no interrupt nor DMA request is issued as the USART would have entered mute mode.

When the received byte matches the receiver's configured address, the USART exits mute mode. The RWU bit is then cleared, and subsequent bytes are received normally. The RXNE bit is set for the address character since the RWU bit has been cleared.

The RWU bit can be written to as 0 or 1 when the receiver buffer contains no data (RXNE=0 in the USART_SR register). Otherwise, the write attempt is ignored. The following figure shows an example of using address mark detection to wake up and enter mute mode.

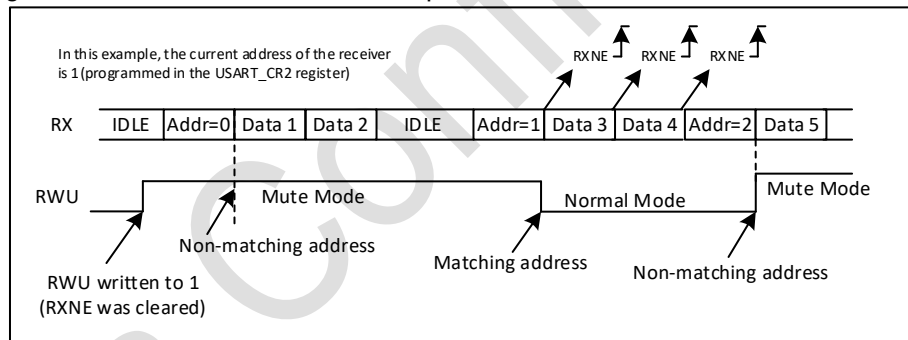


Figure 31-8 Mute Mode Using Address Mark Detection

31.3.10. USART Parity Control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PCE bit in the USART_CR1 register. Depending on the frame length defined by the M bit, the possible USART frame formats are as listed in the table below.

Table 31-5 Frame Modes

M bit	PCE bit	USART Frame
0	0	Start bit—8-bit data—Stop bit
0	1	Start bit—7-bit data—Parity bit—Stop bit
1	0	Start bit—9-bit data—Stop bit
1	1	Start bit—8-bit data—Parity bit—Stop bit

When waking up a device using an address mark, the address matching only considers the MSB of the data, regardless of the parity bit. (The MSB is the last transmitted data bit, followed immediately by the parity bit or stop bit)

31.3.10.1. Even parity

The parity bit is calculated to obtain an even number of “1s” inside the frame made of the 7 or 8 LSB bits and the parity bit.

As an example, if data = 00110101 and 4 bits are set, the parity bit is equal to 0 if even parity is selected (PS bit in USART_CR1 = 0).

31.3.10.2. Odd parity

The parity bit is calculated to obtain an odd number of “1s” inside the frame made of the 7 or 8 LSB

bits and the parity bit.

As an example, if data = 00110101 and 4 bits set, then the parity bit is equal to 1 if odd parity is selected (PS bit in USART_CR1 = 1).

31.3.10.3. Parity checking in reception

If the parity check fails, the PE flag in the USART_SR register is set to 1; if the PEIE bit in the USART_CR1 register is set to 1, an interrupt is generated.

31.3.10.4. Parity generation in transmission

If the PCE bit in the USART_CR1 register is set to 1, the MSB bit of the data written in the data register is transmitted but modified by the parity bit (if even parity is selected (PS=0), the number of '1's is even; if odd parity is selected (PS=1), the number of '1's is odd).

31.3.11. USART synchronous mode

The synchronous master mode is selected by programming the CLKEN bit in the USART_CR2 register to '1'. In synchronous mode, the following bits must be kept cleared:

- LINEN bit in the USART_CR2 register
- SCEN, HDSEL and IREN bits in the USART_CR3 register.

The USART allows users to control bidirectional synchronous serial communication in master mode. The CK pin is the output of the USART transmitter clock. No clock pulses are sent to the CK pin during start bit and stop bit. Depending on the state of the LBCL bit in the USART_CR2 register, clock pulses are, or are not, generated during the last valid data bit. The CPOL bit in the USART_CR2 register is used to select the clock polarity, and the CPHA bit in the USART_CR2 register is used to select the phase of the external clock.

During the Idle state, preamble and send break, the external CK clock is not activated.

In synchronous master mode, the USART transmitter operates exactly like in asynchronous mode. However, since CK is synchronized with TX (according to CPOL and CPHA), the data on TX is synchronous.

In synchronous master mode, the USART receiver operates in a different way compared to asynchronous mode. If RE=1, data is sampled on CK (on the rising or falling edge depending on CPOL and CPHA) without any oversampling. However, setup and hold times must be considered (depending on the baud rate: 1/16 of the bit time).

Note:

- The CK pin can operate in conjunction with the TX pin. Thus, the clock is provided only if the transmitter is enabled (TE = 1) and data are being transmitted (USART_TDR data register written). This means that it is not possible to receive synchronous data without transmitting data.
- The LBCL, CPOL, and CPHA bits should be correctly configured when both the transmitter and receiver are disabled; these bits cannot be changed once the transmitter or receiver is enabled.
- It is advised that TE and RE are set in the same instruction to minimize the setup and the hold time of the receiver.
- The USART **only supports master synchronous mode**: it cannot receive or transmit data using an input clock from another device (CK is always an output).

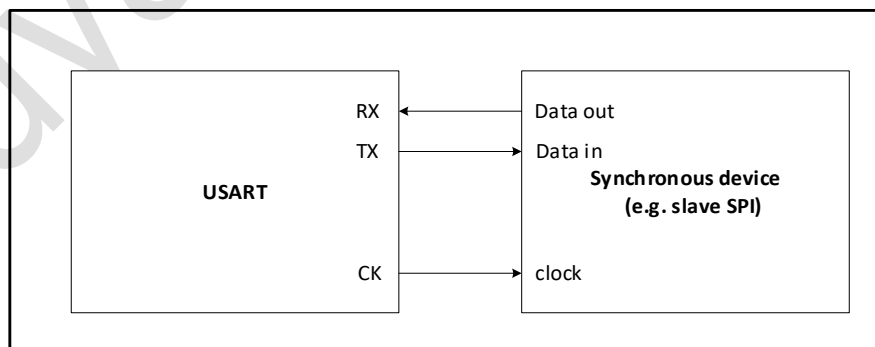


Figure 31-9 Example of USART Synchronous Transmission

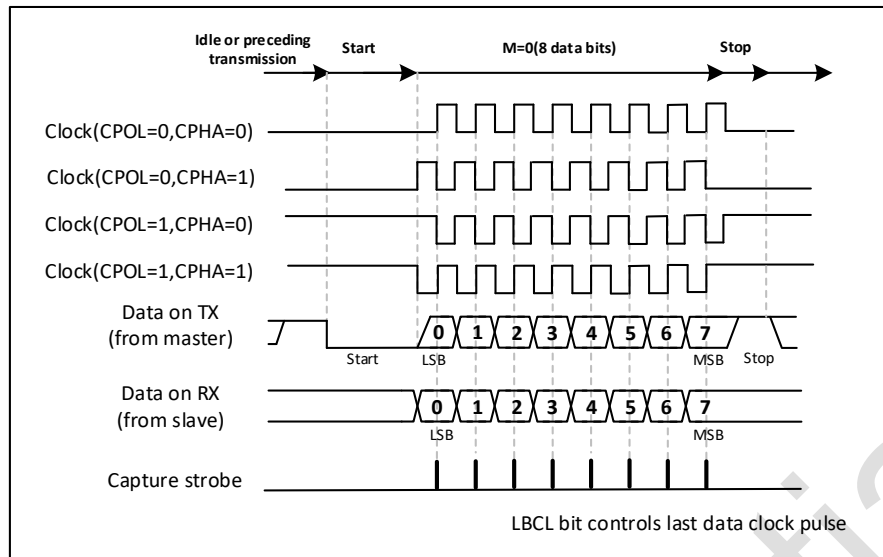


Figure 31-10 USART Data Clock Timing Diagram (M=0)

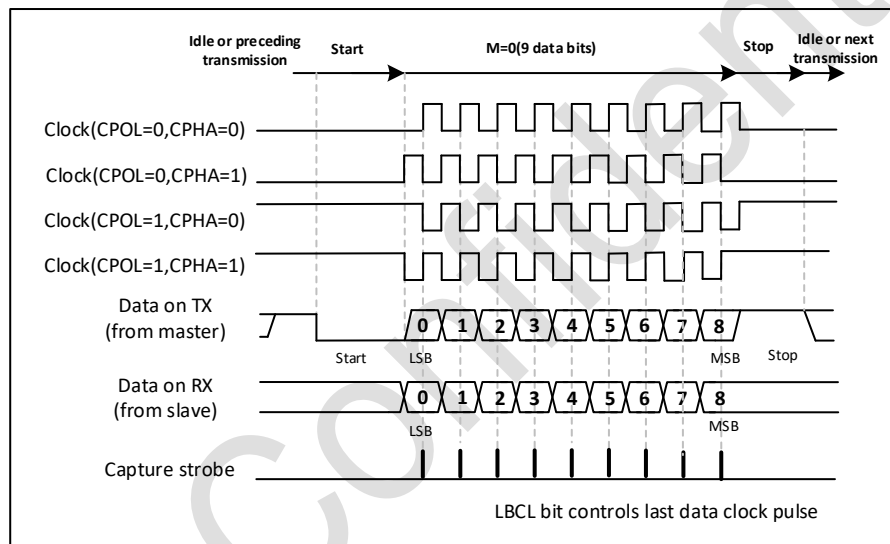


Figure 31-11 USART Data Clock Timing Diagram (M=1)

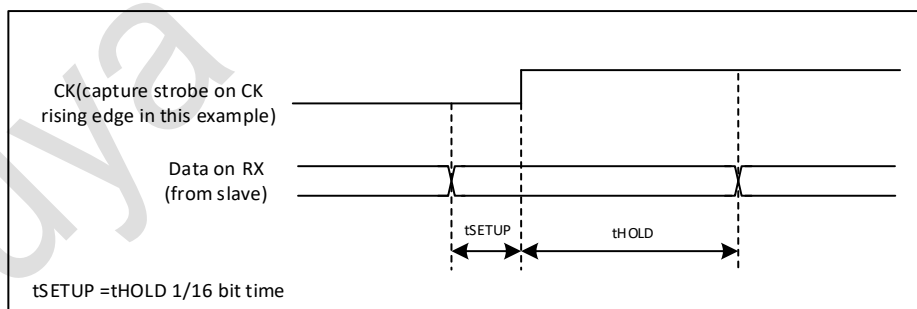


Figure 31-12 RX Data Sampling/Hold Time

Note: The functionality of CK differs in smart card mode; refer to the smart card mode section for details.

31.3.12. USART single-wire half-duplex communication

The single-wire half-duplex mode is selected by setting the HDSEL bit in the USART_CR3 register. In this mode, the following bits must be kept cleared:

- LINEN and CLKEN bits in the USART_CR2 register.
- SCEN and IREN bits in the USART_CR3 register

The USART can be configured to follow a Single-wire Half-duplex protocol. In single-wire half-duplex mode, the TX and RX pins are connected internally. Use the "Half-Duplex Selection" (HDSEL bit in USART_CR3) to choose between half-duplex and full-duplex communication.

When HDSEL is '1'

- The RX pin is no longer used.
- When no data is being transmitted, the TX pin remains in a released state. Therefore, it behaves as a standard I/O port when idle or received. This means that when the I/O is not driven by the USART, TX should be configured as an open-drain alternate function with an external pull-up resistor.

Apart from this, the communication protocol is similar to normal USART mode. Conflicts on the line are managed by software (e.g. through the use of a central arbiter). In particular, the transmission is never blocked by hardware. When the TE bit is set, transmission continues as soon as data is written to the data register.

31.3.13. USART DMA Continuous Communication

The USART can utilize DMA for continuous communication. The DMA requests for the receive buffer and transmit buffer are generated independently.

31.3.13.1. DMA transmission

DMA mode can be enabled for transmission by setting DMAT bit in the USART_CR3 register. When the TXE bit is set to '1', the DMA transfers data from the specified SRAM area to the USART_DR register. The steps to allocate a DMA channel for USART transmission (where x represents the channel number) are as follows:

- 1) Configure SYSCFG_CFGR3 or SYSCFG_CFGR4 to select the DMA channel used by the USART.
- 2) Write the USART_DR register address into the DMA control register, configuring it as the destination address for the transfer. After each TXE event occurs, data is moved from memory to this address.
- 3) Write the memory address into the DMA control register, configuring it as the source address for the transfer. After each TXE event occurs, data is loaded from this memory area into the USART_DR register.
- 4) Configure the total number of bytes to be transferred to the DMA control register.
- 5) Configure the channel priority in the DMA register
- 6) Depending on the requirements of the application, the configuration generates a DMA interrupt when the transfer is half or all complete.
- 7) Clear the TC bit in the USART_SR register by writing 0.
- 8) Activate the channel on the DMA register.

When the number of data transfers programmed in the DMA Controller is reached, the DMA controller generates an interrupt on the DMA channel interrupt vector.

In transmission mode, when the DMA completes the transfer of all data to be sent, the DMA controller sets the TCIF flag in the DMA_ISR register; checking the TC flag in the USART_SR register confirms whether the USART communication has ended, which helps avoid corrupting the last transmitted data before disabling the USART or entering low-power mode; the software must first wait for TXE=1, then wait for TC=1.

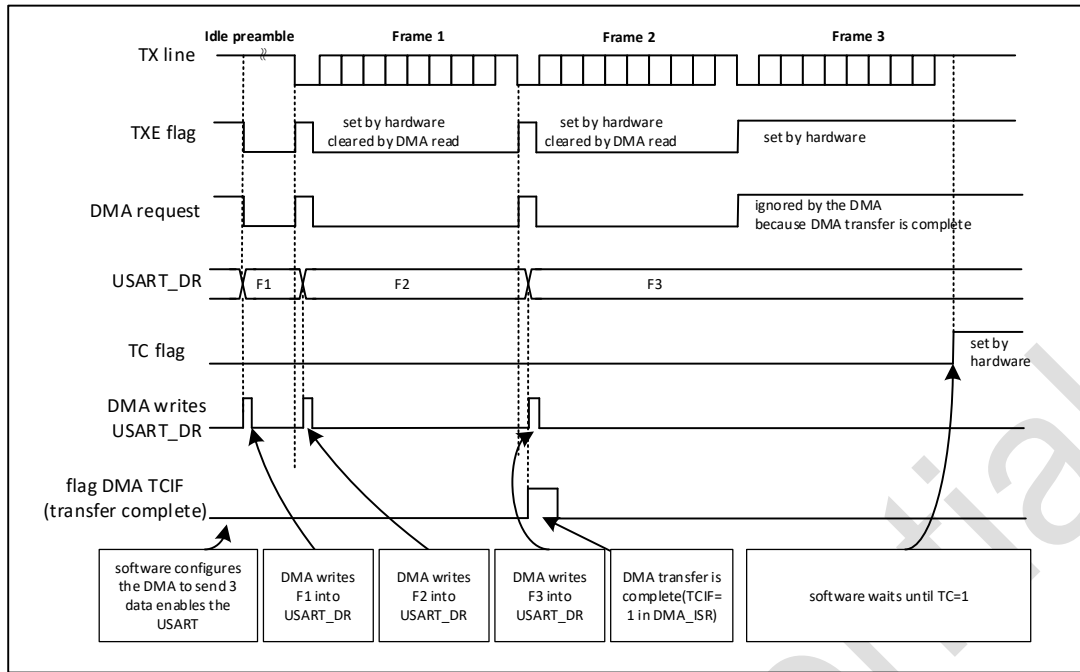


Figure 31-13 DMA Transmission

31.3.13.2. DMA reception

DMA-based reception can be activated by setting the DMAR bit in the USART_CR3 register. Each time a byte is received, the DMA controller transfers the data from the USART_DR register to the specified SRAM area (refer to the DMA-related documentation). The steps to assign a DMA channel for USART reception are as follows (x represents the channel number):

- 1) Configure SYSCFG_CFGR3 or SYSCFG_CFGR4 to select the DMA channel used by the USART.
- 2) Configure the USART_DR register address as the source address for the transfer via the DMA control register. The data will be moved from this address to the memory after each RXNE event.
- 3) Configure the memory address as the destination address for the transfer via the DMA control register. The data will be loaded from USART_DR to this memory area after each RXNE event.
- 4) Configure the total number of bytes to be transferred to the DMA control register.
- 5) Configure the channel priority in the DMA register
- 6) Configure whether a DMA interrupt is generated when the transfer is half-complete or fully complete, based on application requirements.
- 7) Activate the channel in the DMA control register.

When the reception completes the specified transfer amount by the DMA controller, the DMA controller generates an interrupt on the interrupt vector of this DMA channel.

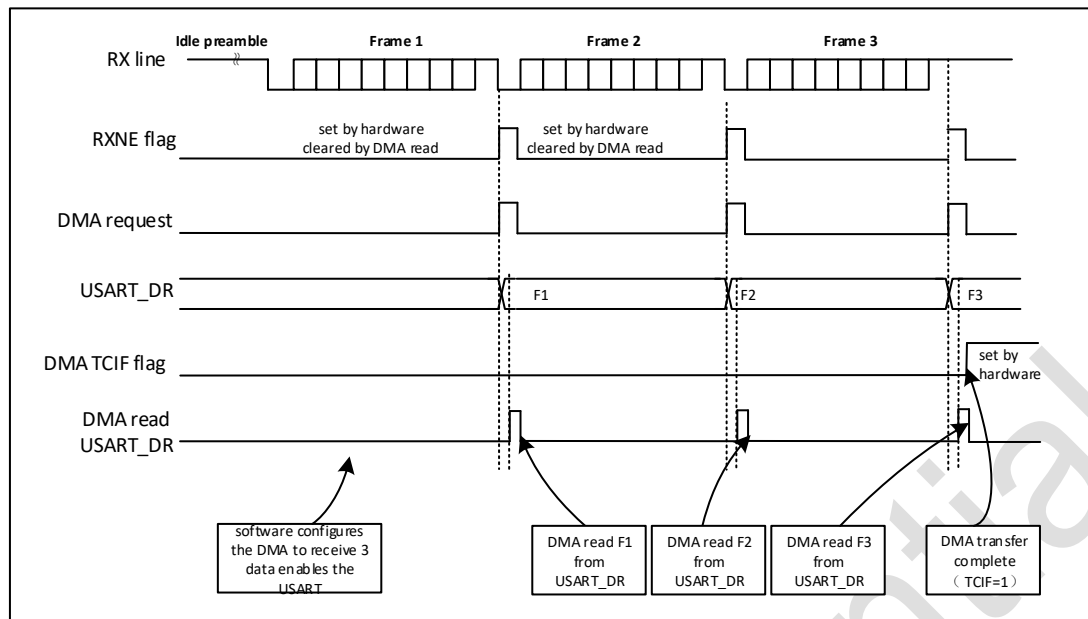


Figure 31-14 Using DMA for Reception

31.3.13.3. Multiprocessor Communication Error Flag and Interrupt Generation

In case of multibuffer communication if any error occurs during the transaction the error flag will be asserted after the current byte. An interrupt will be generated if the interrupt enable flag is set. For framing error, overrun error and noise flag which are asserted with RXNE incase of single byte reception, there will be separate error flag interrupt enable bit, which if set will issue an interrupt after the current byte with either of these errors.

31.3.14. RS232 hardware flow control

It is possible to control the serial data flow between 2 devices by using the nCTS input and the nRTS output. The figure below shows how to connect two devices in this mode.

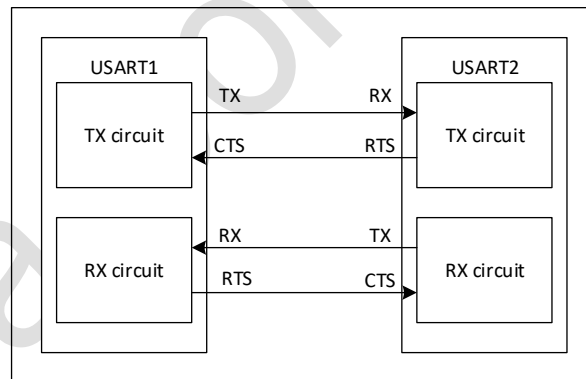


Figure 31-15 Hardware Flow Control Between USARTs

31.3.14.1. RS232 RTS Flow control

If RTS flow control is enabled (RTSE=1), nRTS becomes active (pulled low) whenever the USART receiver is ready to receive new data. When data arrives in the receive register, nRTS is released, indicating that the transmission will stop after the current frame.

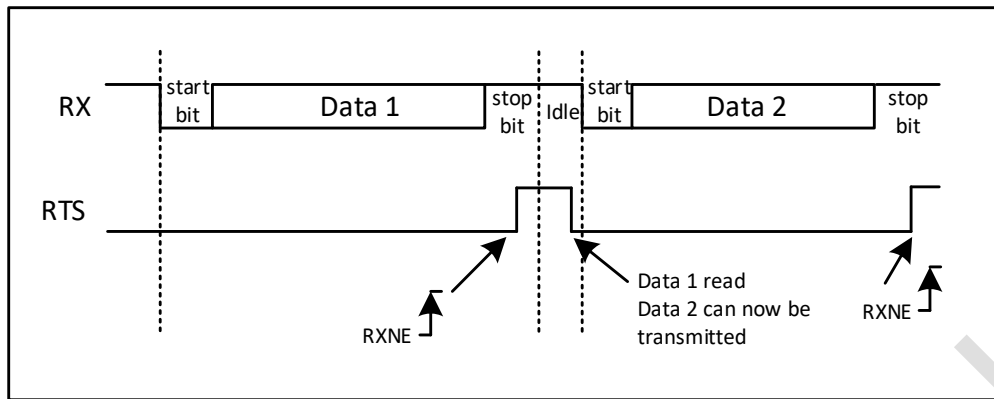


Figure 31-16 RTS flow control

31.3.14.2. RS232 CTS Flow Control

If CTS flow control is enabled (CTSE=1), the transmitter checks the nCTS input before sending the next frame. If nCTS is active (pulled low), the next data will be sent (assuming the data is ready to send, i.e., TXE=0); otherwise, the next frame will not be sent. If nCTS becomes inactive during transmission, the transmitter stops after completing the current transmission.

When CTSE = 1, the CTS status bit is automatically set by hardware as soon as the nCTS input toggles. It indicates when the receiver becomes ready or not ready for communication. An interrupt is generated if the CTSIE bit in the USART_CR3 register is set.

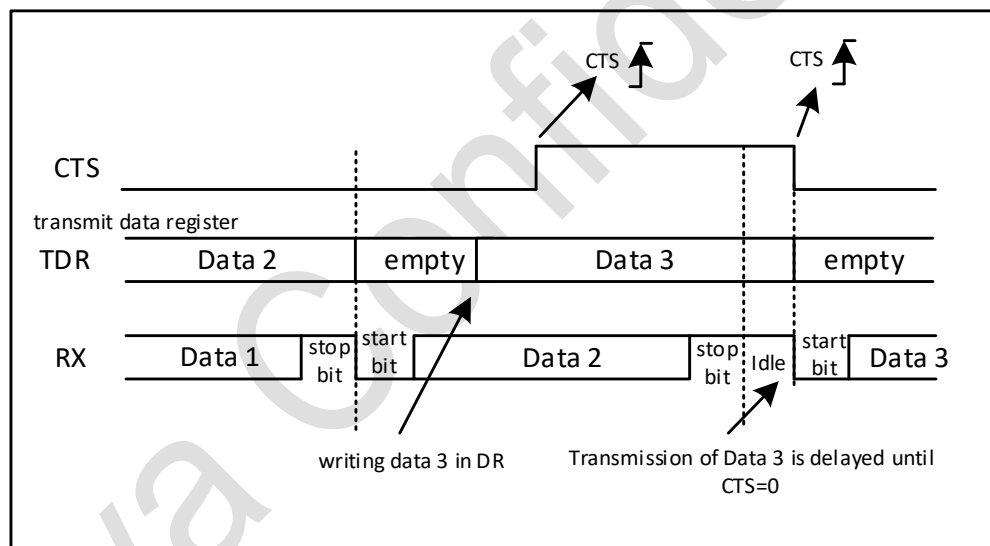


Figure 31-17 CTS flow control

31.3.15. USART LIN mode

Configure USART_CR2. LINEN = 1 to select LIN mode. In LIN mode, the following bits must be kept cleared:

- USART_CR2 register's STOP/CLKEN;
- USART_CR3 register's SCEN/HDSEL/IREN.

31.3.15.1. LIN transmission

Compared with general USART transmission, LIN transmission has the following differences:

- Clear the M bit to configure 8-bit word length;
- USART_CR2. LINEN = 1 needs to be configured. Setting SBK will send 13 bits of 0 as a break frame. Then, a '1' is sent to allow detection of the next start bit.

31.3.15.2. LIN reception

When LIN mode is enabled, the break symbol detection circuit is activated. The detection is totally independent from the normal USART receiver. A break can be detected whenever it occurs, during Idle state or during a frame.

After the receiver is enabled (USART_CR1's RE=1), the circuit begins monitoring the RX line for a start signal. The method for detecting start bits is the same when searching break characters or data. Once a start bit is detected, the circuit samples each subsequent bit using the same method as data

sampling (8th, 9th, and 10th samples). If 10 (when USART_CR2's LBDL=0) or 11 (when USART_CR2's LBDL=1) consecutive bits are 0, followed by a delimiter, the LBD flag in USART_SR is set. If LBDIE bit=1, an interrupt is generated. Before confirming the break symbol, the delimiter is checked because it indicates that the RX line has returned to a high level.

If a '1' is sampled before the 10 or 11 have occurred, the break detection circuit cancels the current detection and searches for a start bit again.

If LIN mode is disabled, the receiver continues to operate as a normal USART and no longer detects break symbols.

If LIN mode is activated (LINEN=1), whenever a frame error occurs (i.e., a '0' is detected in the stop bit, which happens in a break frame), the receiver stops until the break symbol detection circuit receives a '1' (in case of an incomplete break symbol) or a delimiter (in case a complete break symbol has been detected).

本情况：断开帧长度不够：需要舍弃该帧，不设置LBD

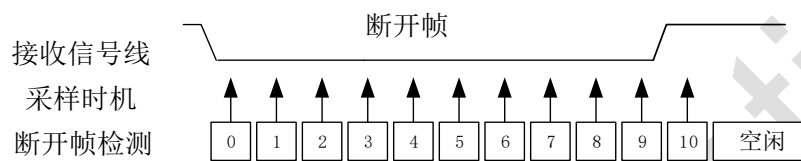
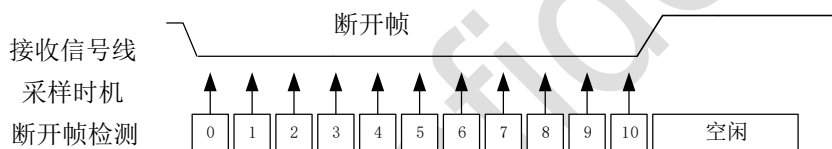


Figure 31- 18 Case of insufficient disconnected frame length in LIN mode

本情况：断开帧长度刚好，设置LBD



本情况：断开帧长度很长，设置LBD

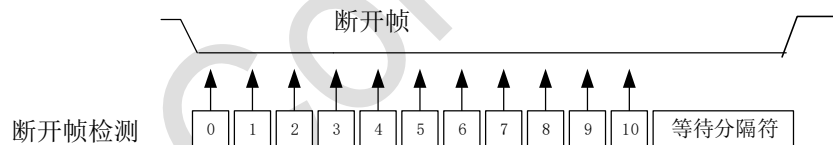
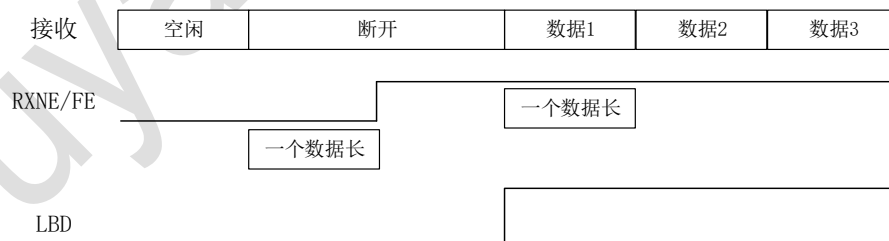


Figure 31-19 Case where the disconnected frame length in LIN mode is sufficient

断开发生在空闲后



断开发生在正在接收数据时

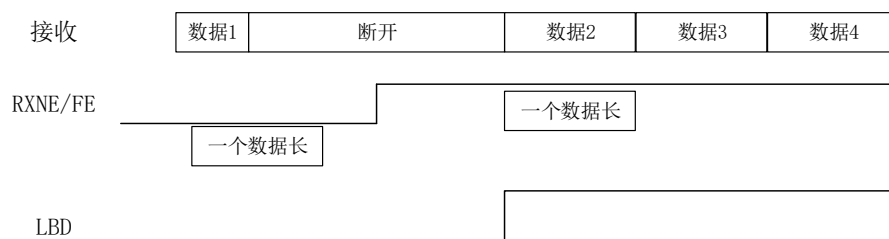


Figure 31-20 Disconnection detection vs. frame error detection in LIN mode

31.3.16. USART Smart Card Mode

Smartcard mode is selected by setting the SCEN bit in the USART_CR3 register. In Smartcard mode, the following bits must be kept cleared:

- LINEN in the USART_CR2 register,
- HDSEL and IREN bits in the USART_CR3 register.

Moreover, the CLKEN bit may be set in order to provide a clock to the smartcard.

Note: When smart card mode is supported, the TE and RE register bits must be configured after SCEN is enabled.

This interface complies with the ISO7816-3 standard for asynchronous smart card protocols. Both T=0 (character mode) and T=1 (block mode) are supported.

The USART should be configured as:

- 8 data bits plus parity bit: In this case, M=1 and PCE=1 in the USART_CR1 register.
- 1.5 stop bits for both transmission and reception: i.e., STOP=11 in the USART_CR2 register. 0.5 stop bits can also be selected for reception, but to avoid switching between configurations, it is recommended to use 1.5 stop bits for both transmission and reception.

Figure below shows examples of what can be seen on the data line with and without parity error.

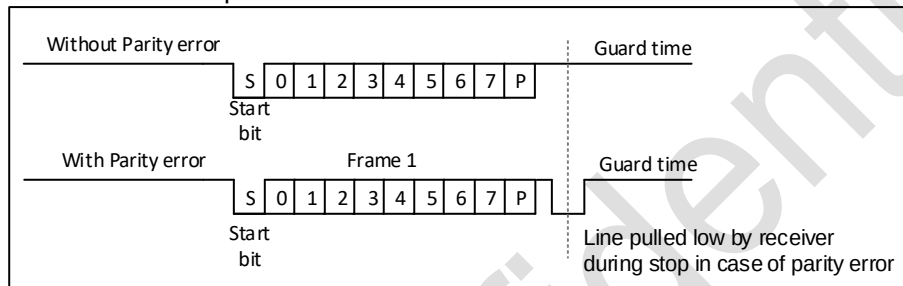


Figure 31-21 ISO7816-3 asynchronous protocol

When connected to a smart card, the USART's TX drives a bidirectional line (the smart card also drives it). To achieve this, RX must be connected to the same I/O pin as TX. During the transmission of the start bit and data byte, the transmitter output enable bit TX_EN is set, and it is released (weak pull-up) during the stop bit transmission, allowing the receiver to pull the data line low in case of a parity error. If TX_EN is not used, TX is pulled high during the stop bit: in this case, the receiver can drive this line as long as TX is configured as an open drain.

Smartcard is a single wire half duplex communication protocol.

- Data transmission from the transmit shift register is delayed by at least 1/2 of the baud clock. During normal operation, a full transmit shift register will begin shifting at the next baud clock edge. In smart card mode, this transmission is further delayed by 1/2 of the baud period.
- If a parity error is detected during the reception of a data frame configured for 0.5 or 1.5 stop bits, the transmit line is pulled low for one baud clock period after completing the reception of the frame (i.e., at the end of the stop bit). This is to indicate to the Smartcard that the data transmitted to USART has not been correctly received. This NACK signal (pulling down the transmit line by one baud clock cycle) will produce a frame error at the transmit end (the transmit end is configured with 1.5 stop bits). The application can handle re-sending of data according to the protocol. A parity error is 'NACK'ed by the receiver if the NACK control bit is set, otherwise a NACK is not transmitted.
- During transmission, the USART inserts a guard time (as programmed in the guard time register) between two consecutive characters. Since the guard time is measured after the stop bit of the previous character, the GT[7:0] register must be programmed to the desired CGT (Character Guard Time, as defined in the ISO7816-3 specification) minus 12 (the duration of one character).
- The assertion of the TC flag can be delayed by programming the Guard Time register. In normal operation, TC is asserted when the transmit shift register is empty and no further transmit requests are outstanding. In smart card mode, an empty transmit shift register triggers the guard time counter to start incrementing until it reaches the value in the guard time register. TC is forced low during this time. When the guard time counter reaches the programmed value TC is asserted high.
- The de-assertion of TC flag is unaffected by Smartcard mode.
- If the transmitter detects a frame error (receiving a NACK signal from the receiver), the receiving function module of the transmitter does not detect the NACK as the start bit. According to the ISO7816-3 protocol, the duration of a received NACK can be either 1 or 2 baud clock periods.
- On the receiver side, if a parity error is detected and a NACK is sent, the receiver will not detect the NACK as a start bit.

Note:

1. The break symbol has no meaning in smart card mode. A 0x00 data with a framing error will be treated as data and not as a break.
2. When toggling the TE bit, no idle frame will be transmitted. The ISO protocol does not define idle frames.

Figure below details how the NACK signal is sampled by the USART. In this example, the USART transmits a data and is configured with 1.5 stop bits. The receiver part of the USART is enabled in order to check the integrity of the data and the NACK signal.

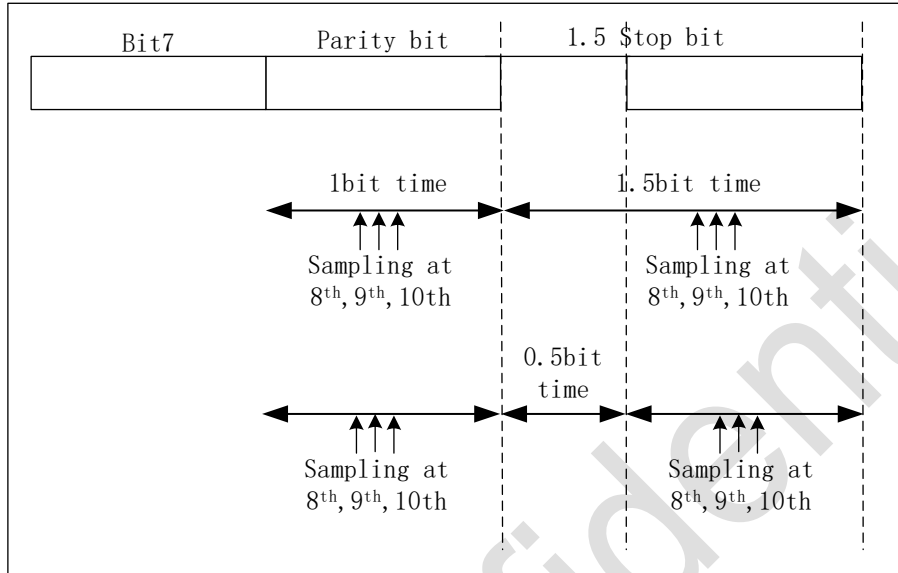


Figure 31-22 1.5 stop bit detection of parity error

The USART can provide a clock to the Smartcard through the CK output. In Smartcard mode, CK is not associated to the communication but is simply derived from the internal peripheral input clock through a 5-bit prescaler. The division ratio is configured in the prescaler register USART_GTPR. The CK frequency can be programmed between $f_{CK}/2$ and $f_{CK}/62$, where f_{CK} is the peripheral input clock.

31.3.16.1. Block mode (T=1)

In T=1 (block) mode, the parity error transmission can be deactivated by clearing the NACK bit in the USART_CR3 register. When requesting a read operation from a smart card in block mode, the software must program the RTOR register to BWT (Block Wait Time - value 11). If no answer is received from the card before the expiration of this period, a timeout interrupt will be generated. If the first character is received after this period, it is signaled by an RXNE interrupt.

Note: Even when using the USART in DMA mode to read from a smart card in block mode, the RXNE interrupt must be enabled. In parallel, the DMA must be enabled only after the first received byte.

After receiving the first character (RXNE interrupt), to allow automatic checking of the maximum wait time between two consecutive characters, the USART_RTOR.RTO register must be programmed to CWT (Character Wait Time - value 11). This time is expressed in baud time units. If the smart card does not send a new character within a period shorter than CWT after the end of the previous character, the USART will indicate this to the software via the RTOF flag and interrupt (when the RTOIE bit is set to 1).

Note: As defined by the smart card protocol, the BWT/CWT value should be defined from the start of the last character (start bit). The RTO register must be programmed to BWT - 11 or CWT - 11, respectively, taking into account the length of the last character itself.

A block length counter is used to count all the characters received by the USART. This counter is reset when the USART is transmitting. The block length is communicated by the smart card in the third byte of the block (start field). This value must be programmed into the BLEN field of the USART_RTOR register. When using DMA mode, before the start of the block, this register field must be programmed to the minimum value (0x0). With this value, an interrupt is generated after the 4th received character. The software must read the LEN field (third byte), its value must be read from the receive buffer.

In interrupt driven receive mode, the length of the block may be checked by software or by programming the BLEN value. However, before the block starts, the maximum value of BLEN (0xFF) can be programmed. The real value will be programmed after the reception of the third character.

If the block is using the LRC longitudinal redundancy check (1 epilogue byte), the BLEN=LEN. If the

block uses the CRC mechanism (2 trailing bytes), $BLN=LEN+1$ must be programmed. The total block length (including prologue, epilogue and information fields) equals $BLN+4$. The end-of-block signal is sent to the software via the EOB flag and an interrupt (when EOBI is set to 1).

In case of an error in the block length, the end of the block is signaled by the RTO interrupt (Character Wait Time overflow).

Note: The error-checking code (LRC/CRC) must be calculated/verified by software.

31.3.16.2. Direct and inverse convention

The Smartcard protocol defines two conventions: direct and inverse.

The direct convention is defined as: LSB first, logical bit value of 1 corresponds to a H state of the line and parity is even. In order to use this convention, the following control bits must be programmed: $MSBFIRST=0$, $DATAINV=0$ (default values).

The inverse convention is defined as: MSB first, logical bit value 1 corresponds to an L state on the signal line and parity is even. In order to use this convention, the following control bits must be programmed: $MSBFIRST=1$, $DATAINV=1$.

Note: When inverting the logical data values (0=H, 1=L), the parity bit will also be inverted.

In order to recognize the card convention, the card sends the initial character, TS, as the first character of the ATR (Answer To Reset) frame. TS supports two formats: LHH LLL LLH and LHH LHH LLH.

- (H) LHH LLL LLH establishes the inverse convention: state L is encoded as value 1, and time component 2 transmits the most significant bit (MSB first). When decoded by inverse convention, the conveyed byte is equal to '3F'.
- (H) LHH LHH LLH establishes the direct convention: state H is encoded as value 1, and time component 2 transmits the least significant bit (LSB first). When decoded by direct convention, the conveyed byte is equal to '3B'.

Character parity is correct when there is an even number of bits set to 1 in the nine moments 2 to 10.

As the USART does not know which convention is used by the card, it needs to be able to recognize either pattern and act accordingly. The pattern recognition is not done in hardware, but through a software sequence. Additionally, if the USART is configured for the direct convention (default) and the smart card responds with the inverse convention (TS = LHH LLL LLH), the byte received by the USART will be "03," and the parity will be odd.

Therefore, two methods are available for TS pattern recognition:

Method 1

Program the USART in standard smart card mode/direct convention. In this case, the TS pattern reception generates a parity error interrupt and error signal to the card.

- A parity error interrupt notifies the software that the smart card did not respond correctly with the direct convention. Software then reprograms the USART for inverse convention
- In response to the error signal, the smart card will retry the same TS character, which the reprogrammed USART will then receive correctly. Alternatively, to respond to the parity error interrupt, the software may decide to reprogram the USART, issue a new reset command to the smart card, and wait for TS again.

Method 2

The USART is programmed in 9-bit/no-parity mode, no bit inversion. In this mode it receives any of the two TS patterns as:

(H) LHH LLL LLH = 0x103 -> Selects inverse convention

(H) LHH LHH LLH = 0x13B -> Selects direct convention

The software checks the received character against these two patterns and, if any of them match, then programs the USART accordingly for the next character reception.

If none of the two is recognized, a card reset may be generated in order to restart the negotiation.

31.3.17. USART IrDA SIR ENDEC function block

IrDA mode is selected by setting the IREN bit in the USART_CR3 register. In IrDA mode, the following bits must remain cleared:

- LINEN, STOP and CLKEN bits in the USART_CR2 register,
- SCEN and HDSEL bits in the USART_CR3 register.

31.3.17.1. IrDA normal mode

The IrDA SIR physical layer specifies the use of an inverted return-to-zero (RZI) modulation scheme, where an infrared light pulse represents a logical '0'. The SIR transmit encoder modulates the NRZ (non-return-to-zero) bitstream output from the USART. The output pulse stream is transmitted to an external output driver and infrared LED. USART is SIR ENDEC only supports up to 115.2 Kbps

speeds. In normal mode the transmitted pulse width is specified as 3/16 of a bit period. The SIR receive decoder demodulates the return-to-zero bit stream from the infrared detector and outputs the received NRZ serial bit stream to USART. In the idle state, the decoder input is typically high (mark state). The transmit encoder output has the opposite polarity to the decoder input. A start bit is detected when the decoder input goes low.

- IrDA is a half duplex communication protocol. If the transmitter is busy (i.e. USART is sending data to the IrDA encoder), any data on the IrDA receive line will be ignored by the IrDA decoder. If the receiver is busy (i.e. the USART is receiving decoded data from the IrDA decoder), the data on the TX from the USART to the IrDA will not be encoded by the IrDA. While receiving data, transmission should be avoided as the data to be transmitted could be corrupted.
- The SIR transmission logic sends '0' as a high pulse and '1' as a low level. The width of the pulse is specified as 3/16th of the selected bit period in normal mode.
- The SIR reception logic interprets a high-level state as '1' and a low pulse as '0'.
- The transmit encoder output has the opposite polarity to the decoder input. The SIR output is in low state when idle.
- The SIR decoder converts the IrDA compliant received signal into a bitstream for the USART.
- The IrDA specification requires the acceptance of pulses greater than 1.41 us. The acceptable pulse width is programmable. The spike detection logic on the receiver side filters out pulses with a width less than 2 PSC cycles (PSC is the prescaler value programmed in the IrDA low-power baud rate register USART_GTPR). Pulses of width less than 1 PSC period are always rejected, but those of width greater than one and less than two periods may be accepted or rejected, those greater than 2 periods will be accepted as a pulse. The IrDA encoder/decoder doesn't work when PSC=0.
- The receiver can communicate with a low-power transmitter.
- In IrDA mode, the stop bits in the USART_CR2 register must be configured to '1 stop bit'.

Note:

1. Pulses having a width of less than 2 and greater than 1 PSC cycle may or may not be filtered out.
2. The setup time of the receiver should be managed by the software. The IrDA physical layer specification specifies a minimum delay of 10 ms between transmission and reception (IrDA is a half-duplex protocol).

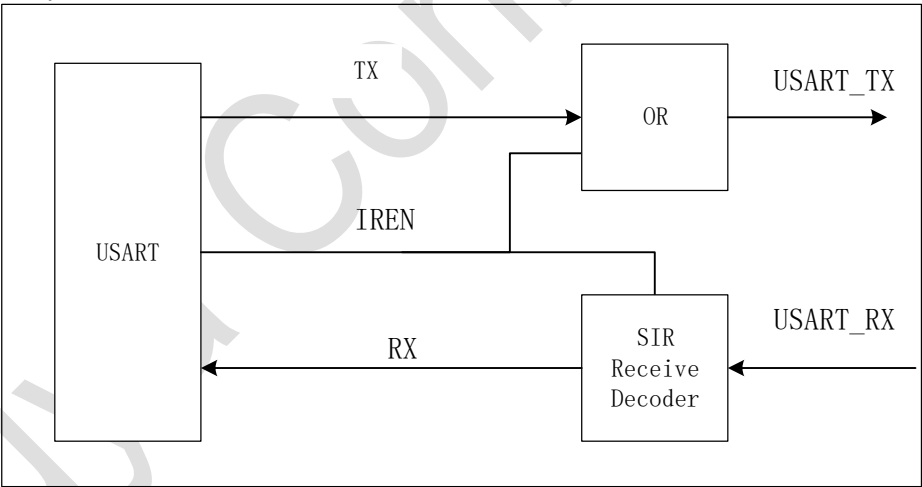


Figure 31-23 IrDA SIR ENDEC Block Diagram

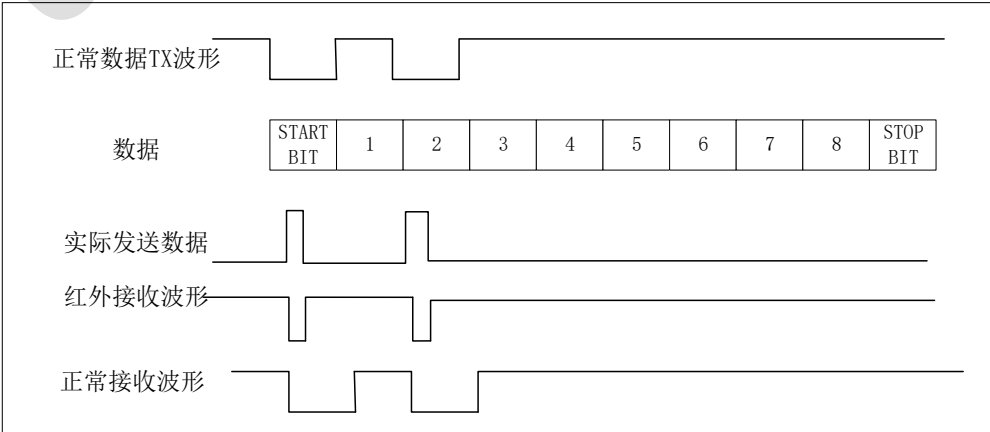


Figure 31-24 IrDA Data Modulation (3/16) — Normal Mode

31.4. USART interrupts

Table 31-6 USART interrupts

No.	Interrupt event	Event flag	Enable Control bit	Interrupt clear method
1	Transmit data register empty	TXE	TXEIE	TXE is cleared when data is written to USART_DR.
2	CTS interrupts	CTS	CTSIE	Writing USART_SR.CTS to 0 clears it.
3	Transmission is complete	TC	TCIE	1. Writing USART_SR.TC to 0 clears it. 2. Reading the USART_SR register and then writing to the USART_DR register clears it.
4	Receive data register not empty (data ready to be read)	RXNE	RXNEIE	1. Writing USART_SR.RXNE to 0 clears it. 2. Reading the USART_DR register clears it.
5	Overrun error	ORE		Reading the USART_SR register and then reading the USART_DR register clears it.
6	Idle line detected	IDLE	IDLEIE	Reading the USART_SR register and then reading the USART_DR register clears it.
7	Parity error	PE	PEIE	Reading the USART_SR register and then reading the USART_DR register clears it.
8	LIN break flag	LBD	LBDIE	Writing USART_SR.LBD to 0 clears it.
9	Noise, overflow, and framing errors in multiprocessor communication.	NE/ORE/FE	EIE	Reading the USART_SR register and then reading the USART_DR register clears it.
10	Transmission parity error (smart card).	TXPE	PEIE	Writing USART_SR.TXPE to 0 clears it.
11	Receive timeout	RTOF	RTOIE	Writing USART_SR.RTOF to 0 clears it.
12	End of block	EOBF	EOBIE	Writing USART_SR.EOBF to 0 clears it.

The USART interrupt events are connected to the same interrupt vector.

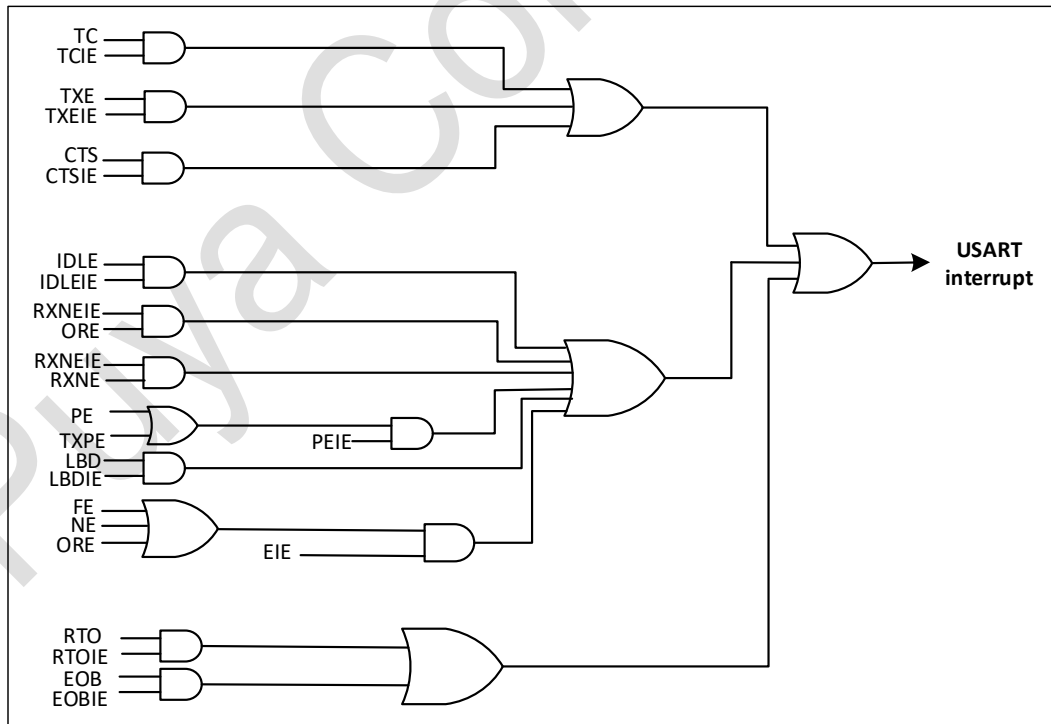


Figure 31-25 USART Interrupt Mapping Diagram

31.5. Software Flow

31.5.1. Smart Card Mode Operation Flow Using BLEN and EOBF Registers

Smart card mode block frame consists of three fields: the preamble field, the information field, and the postamble field, as shown in the table below:

Table 31-7 Smart Card Mode Block Frame

Preamble Field			Information Field	Postamble Field
Node Address NAD	Protocol Control Byte PCB	Length LEN	Optional INF	Error detection LRC or CRC EDC
1 byte	1 byte	1 byte	0-254 bytes	1/2 byte

31.5.1.1. DMA mode

DMA mode, configure according to the following process:

1. At the start or before the start of the frame, configure BLEN=4. When the first byte of INF is received, the hardware can generate an EOBIF interrupt.
2. In the EOBIF interrupt routine, the software parses the length field (the third data) and configures BLEN=LEN.

Note: The operation of clearing EOBIF in this step must be completed within one packet of data; otherwise, it may cause subsequent data to clear the count, resulting in the actual EOBIF being set later.

3. Afterward, the hardware normally receives the information field and will set EOBIF again after the block ends (after LRC/CRC). Indicates the end of block reception.

Note: After our BLEN configuration, EOBIF is set when BLEN bytes of data are received. That is, the value of BLEN includes the information field and the end field.

31.5.1.2. Interrupt mode

Method 1: Utilize the RXNE status bit

The software counts using the RXNE interrupt after receiving each byte of data to determine the end of the block. (EOBIF is not used, and EOBIEN is not enabled).

Method 2: Utilize the EOBIF status bit

1. At the start or before the start of the frame, configure BLEN=0xff,
2. Each time a byte of data is received, an RXNE interrupt is triggered, and the data needs to be parsed;
3. After receiving the third byte, parse the LEN field, and after receiving the fourth byte, configure the actual LEN to the BLEN register;
4. Afterward, the hardware normally receives the information field, and EBOF will be set after the block ends (after LRC/CRC). Indicates the end of block reception.

Note: After our BLEN configuration, the EOBIF flag is set upon receiving BLEN bytes of data. That is, the value of BLEN includes the information field and the end field.

31.6. USART register

31.6.1. USART Status Register (USART_SR)

Offset address: 0x00

Reset value : 0x0000_00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Re s.	Re s.	Res.	Res.	Res.	Res.	Res.	WAIT_R PT	TXBUS Y	RXBUS Y	TXPE	RTOF	EOBF	Re s.	Re s.
								R	R	R	RC_W 0	RC_W 0	RC_W 0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	Re s.	Re s.	ABRR Q	ABR E	ABR F	CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
			W	R	R	RC_W 0	RC_W 0	R	RC_W 0	RC_W 0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23	WAIT_RPT	R	0	This register applies to smart card mode. Set by hardware when a NACK signal is sent and waiting for the other party to retransmit data; Cleared by hardware upon receiving the start bit of data or entering the transmit state. Software can read this bit but cannot modify it.
22	TXBUSY	R	0	Transmit data busy flag. (Automatically cleared after transmission completes) 0: Data transmission idle 1: In data transmission state, the transmit shift register is sending data.

21	RXBUSY	R	0	Receive data busy flag. (Automatically cleared after reception completes) 0: Data reception idle; 1: In data reception state, the receive shift register is receiving data;
20	TXPE	RC_W0	0	Flag indicating a parity error in transmitted data. Set by hardware, cleared by writing 0 in software. Refer to the description of TREPEN for set conditions. This register applies to smart card mode.
19	RTOF	RC_W0	0	This bit is set by hardware when no communication occurs after reaching the timeout value programmed in the RTOR register. An interrupt is generated if RTOIE=1 in the USART_CR2 register. This bit is cleared by writing 0 In Smartcard mode, the timeout corresponds to the CWT or BWT timings. 0: Timeout value not reached 1: Timeout value has been reached, no data has been received Note: If a time equal to the value programmed in RTOR register separates 2 characters, RTOF is not set. If this time exceeds the value + 2 sampling times (2/16 or 2/8, depending on the oversampling mode), the RTOF flag is set to 1. The counter continues counting even if RE = 0, but RTOF is only set when RE = 1. If the timeout has already elapsed when RE is set, then RTOF will be set. If the USART does not support the Receiver timeout feature, this bit is reserved and kept at reset value.
18	EOBF	RC_W0	0	Block End Flag This bit is set by hardware when a complete block has been received (for example T=1 Smartcard mode). Detection is performed when the number of received bytes (from the start of the block, including the start field) is equal to or greater than BLEN + 4. An interrupt is generated if EOBFIE = 1. This bit is cleared by writing 0 0: Block end not reached 1: End of block reached (number of characters) Note: If Smartcard mode is not supported, this bit is reserved and kept at reset value.
17:13	Reserved	-	-	Reserved
12	ABRRQ	W	0	Auto baud rate request Writing 1 to this bit resets the ABRF flag and requests an automatic baud rate measurement on the next received data frame.
11	ABRE	R	0	Auto baud rate error This bit is set by hardware if the baud rate measurement failed (baud rate out of range or character comparison failed). It is cleared by software, by writing 1 to the ABRRQ bit.
10	ABRF	R	0	Auto baud rate detection flag. This bit is set by hardware when the automatic baud rate has been set (RXNE is also set, generating an interrupt if RXNEIE = 1) or when the auto baud rate operation was completed without success (ABRE = 1) (ABRE, RXNE and FE are also set in this case). The software clears the bit by writing 1 to the ABRRQ bit.
9	CTS	RC_W0	0	CTS flag If CTSE = 1 is configured, this register is set to 1 when the CTS input state changes. This bit is cleared by writing 0 An interrupt is generated on CTS flag if the CTSIE bit is set. 0: Value on the CTS line has not changed 1: Value on the CTS line has changed
8	LBD	RC_W0	0	LIN break frame detection flag. This register is configured by hardware when a LIN break frame is detected. This bit is cleared by writing 0 An interrupt is generated if the LBDIE bit =1.

				0: No LIN break frame detected; 1: LIN break frame detected; Note: If the LIN function is not supported, this register is fixed at 0.
7	TXE	R	1	Transmit data register empty During transmission, this register is set by hardware when data from the USART_DR register is transferred to the shift register. An interrupt is generated if the TXEIE bit =1. It is cleared by a write to the USART_DR register. 0: Data is not transferred to the shift register 1: Data is transferred to the shift register
6	TC	RC_W0	1	Transmission complete This bit is set by hardware if the transmission of a frame containing data is complete and if TXE is set. An interrupt is generated if TCIE=1. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). This clearing sequence is recommended only for multibuffer communication. The TC bit can also be cleared by writing a '0' to it. 0: Transmission not complete 1: Transmission is complete
5	RXNE	RC_W0	0	Receive data register not empty flag. During reception, this register is set by hardware when the shift register value is transferred to the USART_DR register. This bit is cleared by reading the USART_DR register or writing 0 to this bit. An interrupt is generated if the RXNEIE bit =1. 0: Data is not received 1: Data received and can be read
4	IDLE	R	0	DLE line detected Detects an idle frame, and this register is set by hardware. An interrupt is generated if the IDLEIE bit =1. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). 0: No idle frame detected 1: Idle frame detected
3	ORE	R	0	Overflow error flag. This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RXNE=1. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). An interrupt is generated if the RXNEIE bit =1. 0: No overflow error occurred 1: Overrun occurred Note: When this register is set, the content of the receive DR register is not lost, but the shift register content is overwritten. An ORE interrupt is generated when EIE or RXNEIE=1.
2	NE	R	0	Noise error flag This bit is set by hardware when noise is detected on a received frame. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). 0: No noise is detected 1: Noise is detected Note: This bit does not generate interrupt as it appears at the same time as the RXNE bit which itself generates an interrupting. Interrupt is generated on NE flag in case of MultiBuffer communication if the EIE bit is set.
1	FE	R	0	Framing error

				This bit is set by hardware when a synchronization mismatch, excessive noise, or break character is detected. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). 0: No frame error detected 1: Frame error or break character detected Note: When RXNE and FE occur simultaneously, no interrupt is generated when FE=1, but an interrupt is generated when RXNE=1. If the word currently being transferred causes both frame error and overrun error, it will be transferred, and only the ORE bit will be set. In multi-buffer communication mode, an interrupt is generated when FE=1 and EIE=1.
0	PE	R	0	Parity error This bit is set by hardware when a parity error occurs in receiver mode. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). However, software must wait for RXNE=1 before clearing this bit. An interrupt is generated if the PEIE bit =1. 0: No parity error 1: Parity error

31.6.2. USART data register (USART_DR)

Offset address: 0x04

Reset value: undefined

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	DR[8:0]	RW	undefined	Data value Contains the Received or Transmitted data character, depending on whether it is read from or written to. The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR). The TDR register provides the parallel interface between the internal bus and the output shift register. The RDR register provides the parallel interface between the input shift register and the internal bus. When transmitting with the parity enabled, the value written in the MSB (bit 7 or bit 8 depending on the data length) has no effect because it is replaced by the parity. When parity is enabled during reception, the MSB bit read is the received parity bit.

31.6.3. USART baud rate register (USART_BRR)

Offset address: 0x08

Reset value : 0x0000_0000

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

Note: This register can only be written when the USART is disabled (USART_CR1.UE=0). In automatic baud rate detection mode, the hardware updates this register.

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:4	DIV_Mantissa[15:4]	RW	0	12-bit integer
3:0	DIV_Faction[3:0]	RW	0	4bit decimal

31.6.4. USART control register 1 (USART_CR1)

Offset address: 0x0C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	DATAINV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWAP	Res.	Res.
				RW									RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSBFIRST	Res.	UE	M	WAKE	PC	PS	PE	TXE	TC	RXNE	IDLE	TE	RE	RW	SB
RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27	DATAINV	RW	1'b0	Binary data reverse. 0: Send/receive logic data in data register in forward/forward logic (1 = H, 0 = L) 1: Send/receive logical data in the data register according to negative/reverse logic (1 = L, 0 = H) Note: The parity bit is also flipped by this definition. This bit field can only be written when the USART is disabled (UE = 0). Note 1: In synchronous mode, this register must be configured as 0. Note 2: During auto baud rate detection, this register is configured as 1, but detection is still based on 0x7F or 0x55, meaning the received data must be inverted to 0x7F or 0x55. In this case, the data received by the DR register will be adjusted by DATAINV.
26:19	Reserved	-	-	Reserved
18	SWAP	RW	1'b0	Swap TX/RX pins. 0: Use the TX/RX pins defined in the standard pinout 1: Swap the functions of TX and RX pins. This allows operation when cross-connected with another USART. This bit field can only be written when the USART is disabled (UE = 0).
17:16	Reserved	-	-	Reserved
15	MSBFIRST	RW	1'b0	The most significant bit comes first. 0: After the start bit, data is sent/received first using data bit 0. 1: After the start bit, data is sent/received first using the MSB (bit 7/8). This bit field can only be written when the USART is disabled (UE = 0). Note 1: In synchronous mode, this register must be configured as 0. Note 2: During auto-baud rate detection, this register is configured as 1, but detection still follows 0x7F or 0x55, meaning the received data must be 0x7F or 0x55. In this case, the data received by the DR register will be adjusted with MSBFIRST.
14	Reserved	-	-	Reserved
13	UE	RW	0	USART enable When this bit is cleared the USART prescalers are stopped. This bit is set and cleared by software. 0: Disable USART prescaler and output, low-power mode 1: Enable USART Note: Before clearing the UE bit, the DMA channel must be disabled, and TE and RE must also be turned off.
12	M	RW	0	0: 1 start bit, 8 data bits, n stop bits 1: 1 start bit, 9 data bits, n stop bits This bit field can only be written when the USART is disabled (UE = 0).
11	WAKE	RW	0	Receive wakeup mode.

				Wake up mode from silent mode. Set and cleared by software. 0: Idle frame wake-up 1: Address Mark This bit field can only be written when the USART is disabled (UE = 0).
10	PCE	RW	0	Parity control enable 0: Parity control disabled 1: Parity control enabled Parity bit: The 9th bit in a 9-bit data frame; the 8th bit in an 8-bit data frame. This bit field can only be written when the USART is disabled (UE = 0).
9	PS	RW	0	Parity selection Set and cleared by software. 0: Even parity 1: odd parity This bit field can only be written when the USART is disabled (UE = 0).
8	PEIE	RW	0	PE interrupt enable Set and cleared by software. 0: Disabled 1: PE interrupt enabled
7	TXEIE	RW	0	TXE interrupt enable Set and cleared by software. 0: Disabled 1: TXE interrupt enabled
6	TCIE	RW	0	Transmission complete interrupt enable Set and cleared by software. 0: Disabled 1: TC interrupt enabled
5	RXNEIE	RW	0	RXNE interrupt enable; Set and cleared by software. 0: Disabled 1: ORE or RXNE interrupt enabled
4	IDLEIE	RW	0	IDLE interrupt enable Set and cleared by software. 0: Disabled 1: IDLE interrupt enabled
3	TE	RW	0	Transmitter enable 0: Transmitter is disabled 1: Transmitter is enabled Note: When disabling UE, TE must also be disabled; when enabling TE, UE must be enabled first or simultaneously.
2	RE	RW	0	Receiver enable 0: Receiver is disabled 1: Enable reception, start detecting the start bit
1	RWU	RW	0	Receiver wakeup This bit indicates if the USART is in mute mode. The register is set when a silent mode sequence is received; If a wake-up sequence is received, the register is cleared. The specific wake-up sequence (address or idle frame) is controlled by the USART_CR1.WAKE bit. 0: Receiver in active mode 1: Receiver in mute mode Note: 1: Before selecting Mute mode, the USART must first receive a data byte, otherwise it cannot function in Mute mode with wakeup by Idle line detection. Note: 2: In Address Mark Detection wakeup configuration (WAKE bit=1) the RWU bit cannot be modified by software while the RXNE bit is set.
0	SBK	RW	0	Send a break frame. Set this register via software to send a break byte. After the stop bit of the disconnected frame is transmitted, the hardware clears this register. 0: Do not send a break byte 1: Send a break byte

31.6.5. USART control register 2 (USART_CR2)

Offset address: 0x10

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	RREPE N	TREPE N	REP_ T	ERSW[1:0]		RTOE N	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	RW	RW	RW	RW	RW	RW									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	LINEN	STOP[1:0]		CLKE N	CPO L	CPHA	LBC L	Res.	LBDI E	LBD L	Res.	ADD[3:0]			
	RW	RW	RW	RW	RW	RW	RW		RW			RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function															
31	Reserved	-	-	Reserved															
30	RREPEN	RW	1'b0	This register applies to smart card mode. Selection of handling method for received data parity error. 0: For parity error, do not automatically send NACK, set PE flag, and generate interrupt when PEIE=1. 1: For parity error, automatically send NACK according to T=0 protocol. When the number of consecutive single-byte receptions exceeds REP_T, set PE flag and generate interrupt when PEIE=1. This bit field can only be written when SCEN=1. When RREPEN and NACK register configurations differ, the conditions for sending NACK and setting PE are as follows: <table><tr><th>RREPEN</th><th>NACK</th><th>Description</th></tr><tr><td>0</td><td>0</td><td>Parity error, do not send NACK, set PE.</td></tr><tr><td>0</td><td>1</td><td>Parity error, do not send NACK, set PE.</td></tr><tr><td>1</td><td>0</td><td>Parity error, do not send NACK, set PE after consecutive receptions exceed REP_T.</td></tr><tr><td>1</td><td>1</td><td>Parity error, send NACK, set PE after consecutive receptions exceed REP_T.</td></tr></table>	RREPEN	NACK	Description	0	0	Parity error, do not send NACK, set PE.	0	1	Parity error, do not send NACK, set PE.	1	0	Parity error, do not send NACK, set PE after consecutive receptions exceed REP_T.	1	1	Parity error, send NACK, set PE after consecutive receptions exceed REP_T.
RREPEN	NACK	Description																	
0	0	Parity error, do not send NACK, set PE.																	
0	1	Parity error, do not send NACK, set PE.																	
1	0	Parity error, do not send NACK, set PE after consecutive receptions exceed REP_T.																	
1	1	Parity error, send NACK, set PE after consecutive receptions exceed REP_T.																	
29	TREPEN	RW	1'b0	This register applies to smart card mode. Selection of handling method for transmitted data parity error. 0: When NACK is received, do not automatically retransmit, hardware sets TXPE flag, and generates interrupt if PEIE is enabled; 1: When parity error flag (NACK) is received, automatically retransmit according to T=0 protocol. After the number of consecutive single-byte transmissions exceeds REP_T, hardware sets TXPE flag, and generates interrupt if PEIE is enabled; This bit field can only be written when SCEN=1.															
28	REP_T	RW	1'b0	This register applies to smart card mode. Controls the number of automatic retransmissions upon receiving data parity errors 0: 1 time 1: 3 times This bit field can only be written when SCEN=1.															
27:26	ERSW	RW	2'b0	NACK width selection. 11: NACK width is 1 bit; 10: NACK width is 1.5 bits; 01: NACK width is 2 bits; 00: NACK width is 2 bits; This bit field can only be written when SCEN=1. This register applies to smart card mode.															
25	RTOEN	RW	1'b0	Receiver timeout enable. This bit is set and cleared by software. 0: Disable receiver timeout function. 1: Enable the receiver timeout function.															

				When enabled, if the RX line remains idle for the duration programmed in the RTOR (Receiver Timeout Register) (no reception), the RTOF flag in the USART_SR register is set to 1.
00:15	Reserved	-	-	Reserved
14	LINEN	RW	0	LIN mode enable. Set and cleared by software. 0: LIN mode disabled; 1: LIN mode enabled; In LIN mode, enabling the SBK bit sends a LIN sync break frame (13 low bits). Note: If LIN functionality is not supported, this register is fixed at 0.
13:12	STOP[1:0]	RW	2'b0	Stop bits 00: 1 stop bit; 01: 0.5 stop bit; 10: 2 Stop bits 11: 1.5 stop bits;
11	CLKEN	RW	0	CK pin enable. 0: CK pin disabled 1: CK pin enable; This bit is reserved when synchronous mode is not supported.
10	CPOL	RW	0	Clock polarity Synchronous mode, CK pin output clock polarity selection. 0: When idle, the CK pin maintains a stable low level; 1: When idle, the CK pin maintains a stable high level; This bit can only be written when UE=0.
9	CPHA	RW	0	This bit is used in synchronous mode to select the phase of the CK pin output clock. It works in conjunction with the CPOL bit to produce the desired clock/data relationship. 0: The first clock transition is the first data capture edge; 1: The second clock transition is the first data capture edge; This bit can only be written when UE=0.
8	LBCL	RW	0	Whether the clock pulse of the last data bit is output on the CK pin. 0: The clock pulse of the last data bit is not output on the CK pin; 1: The clock pulse of the last data bit is output on the CK pin; This bit can only be written when UE=0.
7	Reserved	-	-	Reserved
6	LBDIE	RW	0	LIN break interrupt enable. 0: Interrupt is disabled 1: Interrupt generated Note: If LIN functionality is not supported, this register is fixed at 0.
5	LBDL	RW	0	LIN disconnect frame detection length. 0: 10-bit break frame detection; 1: 11-bit break frame detection; Note: If LIN functionality is not supported, this register is fixed at 0.
4	Reserved	-	-	Reserved
3:0	ADD[3:0]	RW	4'b0	Address of the USART node This register is used in multiprocessor mute mode as the address for 4-bit address wake-up.

31.6.6. USART control register 3 (USART_CR3)

Offset address: 0x14

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	EOBI E	RTOI E	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

	RW	RW													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	ABRMOD[1:0]		ABR REN	OVER 8	CTSI E	CTS E	RTS E	DMA T	DMA R	SCE N	NAC K	HDSE L	IRLP	IRE N	EIE
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30	EOBIE	RW	1'b0	EOB interrupt enable 0: Interrupt is disabled 1: interrupt enable;
29	RTOIE	RW	1'b0	RTO interrupt enable 0: Interrupt is disabled 1: interrupt enable;
28:15	Reserved	-	-	Reserved
14:13	ABRMOD[1:0]	RW	2'b0	Auto baud rate mode 00: Measure baud rate from the start bit. 01: Falling edge to falling edge measurement 10: 0x7F frame detection method. 11: 0x55 frame detection method. This register can only be configured when ABREN=0 or UE=0.
12	ABREN	RW	0	Auto baud rate enable 0: Auto baud rate detection is disabled 1: Auto baud rate detection is enabled
11	OVER8	RW	0	Oversampling mode 0: Oversampling by 16 1: Oversampling by 8 This bit is only writeable when UE = 0. Note: When enabling synchronous mode or LIN mode, this register must be configured to 0.
10	CTSIE	RW	0	CTS interrupt enable 0: CTS interrupt disabled; 1: CTS interrupt enable;
9	CTSE	RW	0	CTS enable 0: CTS hardware flow control mode disabled; 1: CTS hardware flow control mode enabled. Data is only transmitted when the CTS input is asserted (tied to 0). If a data is written into the data register while CTS is deasserted, the transmission is postponed until CTS is asserted.
8	RTSE	RW	0	RTS enable 0: RTS hardware flow control disabled 1: RTS hardware flow control enabled, the next data will only be requested when the receive buffer is not full. The transmission of data is expected to cease after the current character has been transmitted. The RTS output is asserted (tied to 0) when a data can be received.
7	DMAT	RW	0	DMA enable transmitter. 0: Interrupt is disabled 1: Enable DMA during transmission;
6	DMAR	RW	0	DMA enable receiver 0: Interrupt is disabled 1: DMA mode is enabled for reception
5	SCEN	RW	0	Smartcard mode enable. 0: Smart card mode disabled; 1: Smart card mode enabled; This register can only be configured when UE=0.
4	NACK	RW	0	Smartcard NACK enable 0: NACK transmission in case of parity error is disabled; 1: NACK transmission during parity error is enabled;
3	HDSEL	RW	0	Half-duplex selection 0: Half duplex mode is not selected 1: Select half-duplex mode; This register can only be configured when UE=0.
2	IRLP	RW	0	IrDA low power 0: Normal mode;

				1: IrDA low power mode; This register can only be configured when UE=0.
1	IREN	RW	0	IrDA mode enable This bit is set and cleared by software. 0: IrDA disabled; 1: IrDA enabled; This register can only be enabled when UE=0, TE=0, and RE=0.
0	EIE	RW	0	Error interrupt enable 0: Interrupt is disabled 1: Frame error FE, overflow error ORE, and noise NF interrupts are enabled.

31.6.7. USART Guard Time and Prescaler Register (USART_GTPR)

Offset address: 0x18

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:8	GT[7:0]	RW	0	Guard time value. This bit-field gives the Guard time value in terms of number of baud clocks. This is used in Smartcard mode. The Transmission Complete flag is set after this guard time value. Note: If the smart card function is not supported, this register is fixed at 0.
7:0	PSC[7:0]	RW	0	Prescaler value. - In infrared (IrDA) low power mode: PSC[7:0]=Infrared low-power baud rate. Used for programming the prescaler for dividing the system clock to achieve the low-power frequency: The source clock is divided by the value in the register (only 8 bits are valid). 00000000: Reserved – do not write this value; 00000001: divides the source clock by 1; 00000010: divides source clock by 2; - In normal mode for infrared (IrDA): PS can only be set to 00000001. - In smartcard mode: PSC [4: 0]: prescaled frequency value Used for programming the prescaler for dividing the system clock to provide the smartcard clock. The value given in the register (the lower 5 bits are valid) is multiplied by 2 as the frequency division factor for the source clock. 00000: Reserved - do not program this value; 00001: divides the source clock by 2; 00010: divides the source clock by 4; 00011: divides the source clock by 6; 11111: Divide the source clock by 62; Note: Bits [7:5] are meaningless in smart card mode.

31.6.8. USART receive timeout register (USART_RTOR)

Offset address: 0x1C

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:24	BLLEN	RW	0	Block length. This register defines the block length for T=1 reception in smart card mode. Its value equals the number of information characters plus the length of the end field minus 1. For example: BLLEN =0: 0 information characters + LEC BLLEN =1: 0 information characters + CRC BLLEN = 255: 254 information characters + CRC In smart card mode, the block length calculation is reset when TXE=0. This bit field can also be used in other modes. In this case, the block length counter is reset when RE = 0 (receiver disabled) and/or the EOBFF bit is written to 0.
23:0	RTO[23:0]	RW	0	Receive timeout duration when there is no activity on the RX line. In standard mode, if no new start bit is detected within the RTO period after receiving the last character, the RTOF flag is set at the configured time. In smart card mode, this value is used to execute CWT and BWT. In this case, CWT and BWT are calculated from the start bit of the last received character.

32. Universal asynchronous receivers/transmitter (UART)

32.1. Introduction to UART

This project implements two UART modules, both with identical functionality.

UART is a programmable universal asynchronous transceiver.

32.2. UART main features

- AMBA APB interface
- Support 5/6/7/8/9-bit serial data
- Support 1/2 STOP bits (1/1.5 STOP bits for 5-bit data)
- Support sending address/data
- Support fixed parity check
- Supports disconnected frames
- Detect start bit errors
- Support programmable fractional baud rate: The serial data baud rate is programmable and derived from the formula: Baud Rate = Serial Clock Frequency / (16 × Divisor).
- Support 4-bit fractional baud
- Supports Tx/Rx pin interchange function
- Support MSB FIRST endianness switching
- Support DMA transmission

32.3. UART functional description

32.3.1. UART block diagram

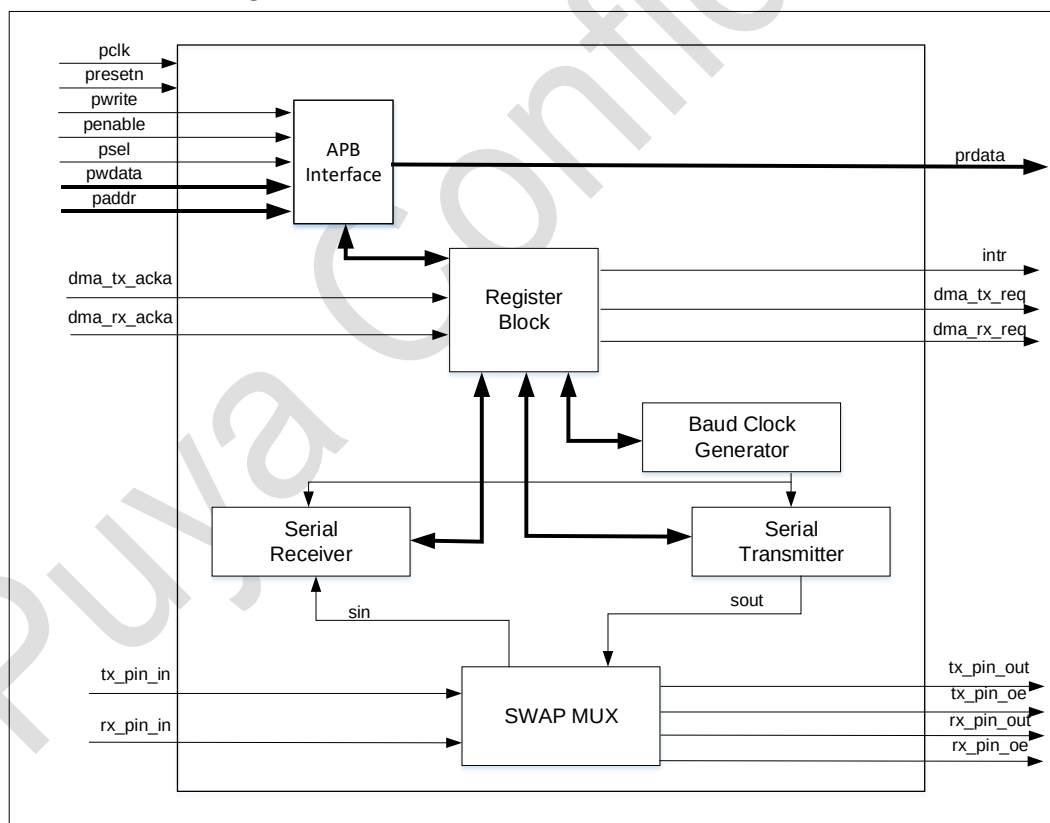


Figure 32-1 UART block diagram

Note: 1. The bold line indicates that it exists and is a bus.

Main modules:

- APB slave interface (APB interface)--APB bus interface.
- Register Block (Register Block)---Responsible for the main functions of UART, including control, status and interrupt generation.
- Baud Clock Generator (Baud Clock Generator)--Generates transmitter and receiver baud clocks and outputs reference clock signals.

- Serial Transmitter---Converts parallel data written to UART into serial form and adds all additional bits specified by the control register for transmission.
- Serial Receiver (Serial Receiver)---Converts the serial data characters specified by the control register received in UART into parallel form. This block controls:
 - Parity error detection
 - Frame error detection
 - Disconnected frame detection
- Pin SWAP Processing (SWAP_MUX)--SWAP-enabled pin handling

32.3.2. UART (RS232) serial protocol

Since the serial communication between the UART and the selected device is asynchronous, additional bits (start and stop) are added to the serial data to indicate the beginning and end. These bits can be used to synchronize the two devices. This serial data structure consisting of start bits and stop bits is called a character, as shown in the following figure.

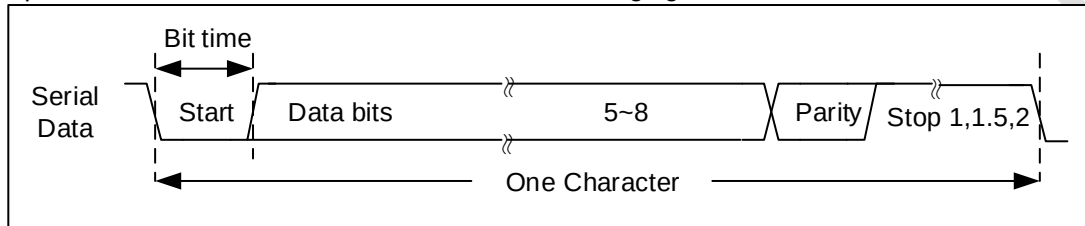


Figure 32-2 serial data format

An additional parity bit can be added to the serial character. This bit occurs after the last data bit in the character structure and before the stop bit in order to provide the UART with the ability to perform simple error checking on the received data.

The UART line control register is used to control serial character characteristics. The individual bits of the data word are transmitted after the start bit, starting with the least significant bit (LSB). They are followed by an optional parity bit followed by a stop bit, which can be 1, 1.5, or 2.

Note: The stop bit duration for UART implementations may be longer for the following reasons:

- Idle time inserted between some configured characters

All bits in the transmission are transmitted for exactly the same duration; The exception to this is the half stop bit when 1.5 stop bits are used. This duration is referred to as a bit period or bit time; One bit of time is equal to sixteen baud clocks.

To ensure line stability, once the start bit is detected, the receiver samples the serial input data at approximately the midpoint of the bit time. Because the exact number of baud clocks transmitted per bit is known, it is not difficult to calculate the midpoint of the sample; That is, every sixteen baud clocks after the midpoint sampling of the start bit.

Together with serial input dejittering, this sampling helps avoid the detection of erroneous start bits. Short faults are filtered out by de-dithering and no transitions are detected on the line. A falling edge is detected if the fault is wide enough that filtering can be avoided by de-dithering. However, the start bit is detected only when the line is again sampled low after half a sampling period.

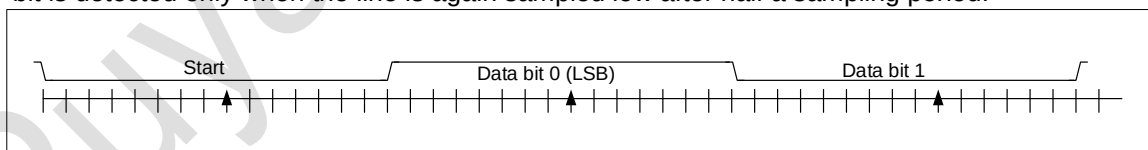


Figure 32-3 receiver serial data sampling points

As part of the 16550 standard, an optional baud clock reference output signal (baudout_n) provides timing information for receiving devices that need it. The baud rate of the UART is controlled by the serial clock pclk in a single clock implementation as well as the frequency division latch register (BRR).

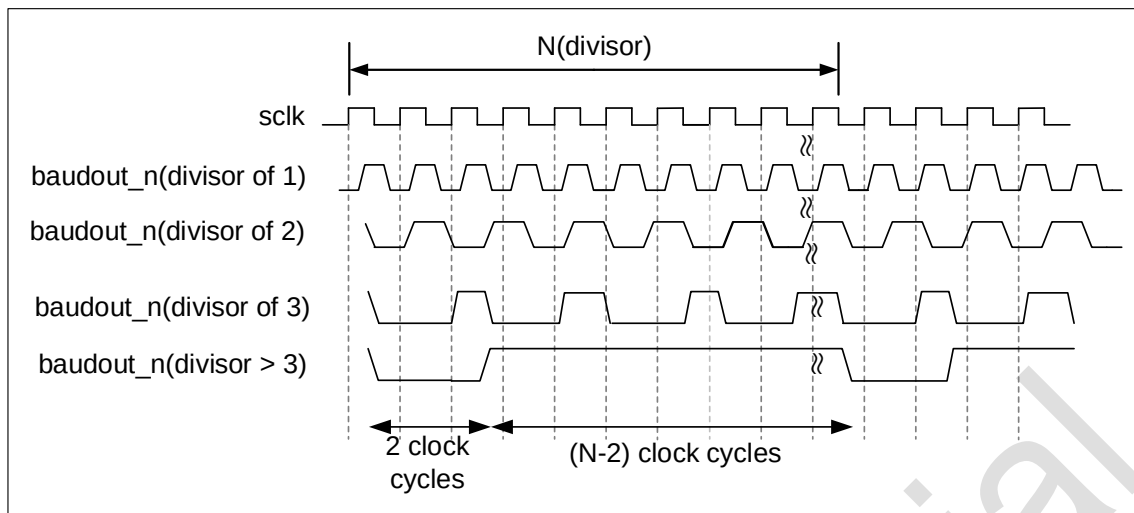


Figure 32-4 Timing diagram of baudout_n output with different divisors

32.3.3. UART 9-bit data transmission

The UART may be configured to have 9-bit data transmission in both transmit and receive modes. The 9th bit in the character occurs after the 8th bit of the character and before the parity bit. The figure below shows the serial transfer of characters, where D8 denotes the 9th bit, and also shows the general serial transfer in 9-bit mode. (This is the normal little-endian mode. If it is the big-endian mode, the order of 9 data bits will be switched)

9-bit data

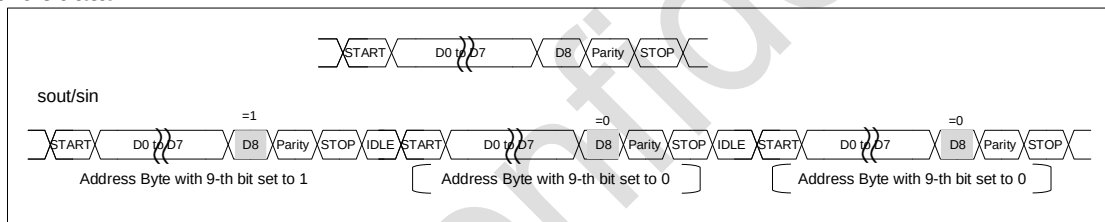


Figure 32-5 9 Bit characters

By enabling 9-bit data transfer mode, UART can be used in multipoint systems where one master is connected to multiple slaves in the system. The master communicates with one of the slaves. When a master wants to transfer a block of data to a slave, it first sends an address byte to identify the destination slave.

The distinction between address/data bytes is done based on the 9th bit in the input character. If the 9th bit is set to 0, the character represents a data byte. If the 9th bit is set to 1, the character represents the address byte. All slave systems compare address bytes to their own addresses, and only the destination slave (where the addresses match) is able to receive data from the master. The master starts sending data bytes to the target slave. An unaddressed slave ignores incoming data until a new address byte is received.

In the image above, notice an address followed by 2 data bytes. The address byte is output with the 9th bit (D8) set to 1, while the data byte is emitted with the 9th bit (D8) set to 0. The parity bit is an optional field.

The configuration of the UART for 9-bit data transmission does the following:

- The UART_CR3. M_E bit is used to enable or disable 9-bit data transmission.
- In the case of reception, UART_CR3. ADDR_MATCH is used to select between hardware and software based address matching.
- The UART_CR3. SEND_ADDR bit is used to enable the transmission address in the case of transmission.
- The UART_CR3. TX_MODE bit is used to select between hardware and software based address transfer.
- The UART_TAR and UART_RAR registers are used to transmit the address and match the received address, respectively.
- The UART_DR (TDR/RDR) register is a 9-bit register used for data transfer in 9-bit mode.
- The UART_SR.ADDR_RCVD bit is used to indicate an address reception interrupt.

32.3.3.1. Send Mode

UART supports two types of transmission modes:

- Transfer Mode 0 (when UART_CR3. TX_MODE is set to 0)

■ Transfer Mode 1 (when UART_CR3.TX_MODE is set to 1)

Mode 0

In transfer mode 0, addresses are programmed in the transfer address register (TAR) and data is written to the transfer hold register (TDR). The 9th bit of the TDR register is not applicable in this mode.

The following figure illustrates address and data transmission based on SEND_ADDR (CR3 [2]), TDR NULL conditions.

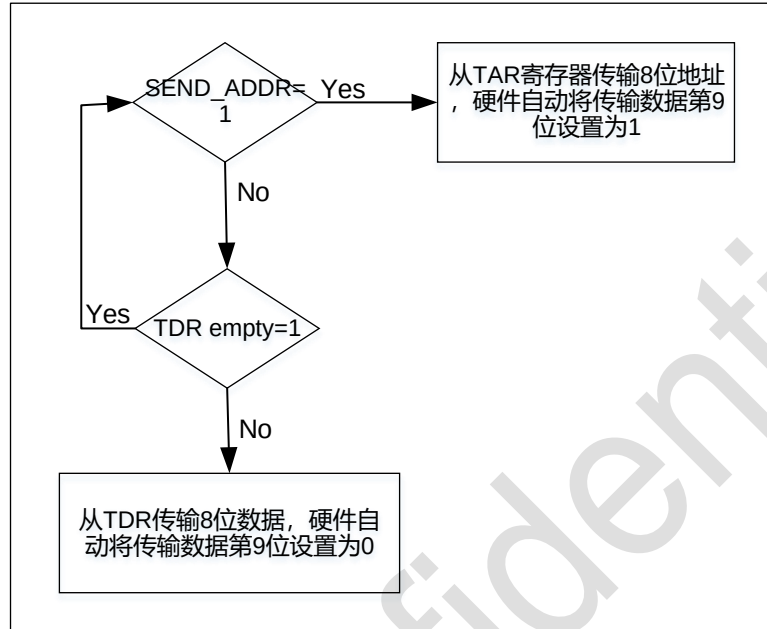


Figure 32-6 Automatic address transfer flow chart

The address of the destination slave to which data is to be transferred is programmed in the TAR register. The SEND_ADDR(CR3 [2]) bit must be enabled to transfer the destination slave address in the TAR register on the serial UART line, where the 9th data bit is set to 1, indicating that the address is being sent to the slave. UART clears the SEND_ADDR bit after the address character begins to be transferred on the line.

The data needed to be transferred to the target slave is programmed through a transfer hold register (TDR). Data is transmitted on the UART line, and the 9th data bit is set to 0, indicating that data is being sent to the slave.

Mode 1

In Transfer Mode 1, the TDR register is 9 bits wide and both addresses and data are programmed through the TDR register. UART does not distinguish between address and data. The SEND_ADDR (UART_CR3 [2]) bit and the Transfer Address Register (UART_TAR) are not available for this mode. Depending on whether address/data must be sent, the software must write the 9th bit with 1/0.

M_E	TX_MODE	SEND_ADDR	Usage scenarios
0	X	X	Send 8-bit TDR data
1	0	0	Send "0 +8-bit TDR data"
1	0	1	Send "1 +8-bit TAR address"
1	1	X	Send "9-bit TDR data"

Note: This table describes several sending scenarios and corresponding configurations that can be used as a sender.

32.3.3.2. Receiving Mode

UART supports two reception modes:

- Hardware address matching receive mode (when ADDR_MATCH (UART_CR3 [1]) is set to 1)
- Software address matching receive mode (when ADDR_MATCH (UART_CR3 [1]) is set to 0)

Hardware address matching receiving mode

In hardware address matching receive mode, if the 9th bit of the receive character is set to 1, UART matches the receive character to the address programmed in the receive address register (UART_RAR):

If the received address successfully matches the programmed address in the UART_RAR register, the data byte is then received.

If the address match fails, the UART controller discards the data characters until a matching address

is received.

The following figure shows the flow chart of data byte receiving based on the address matching function.

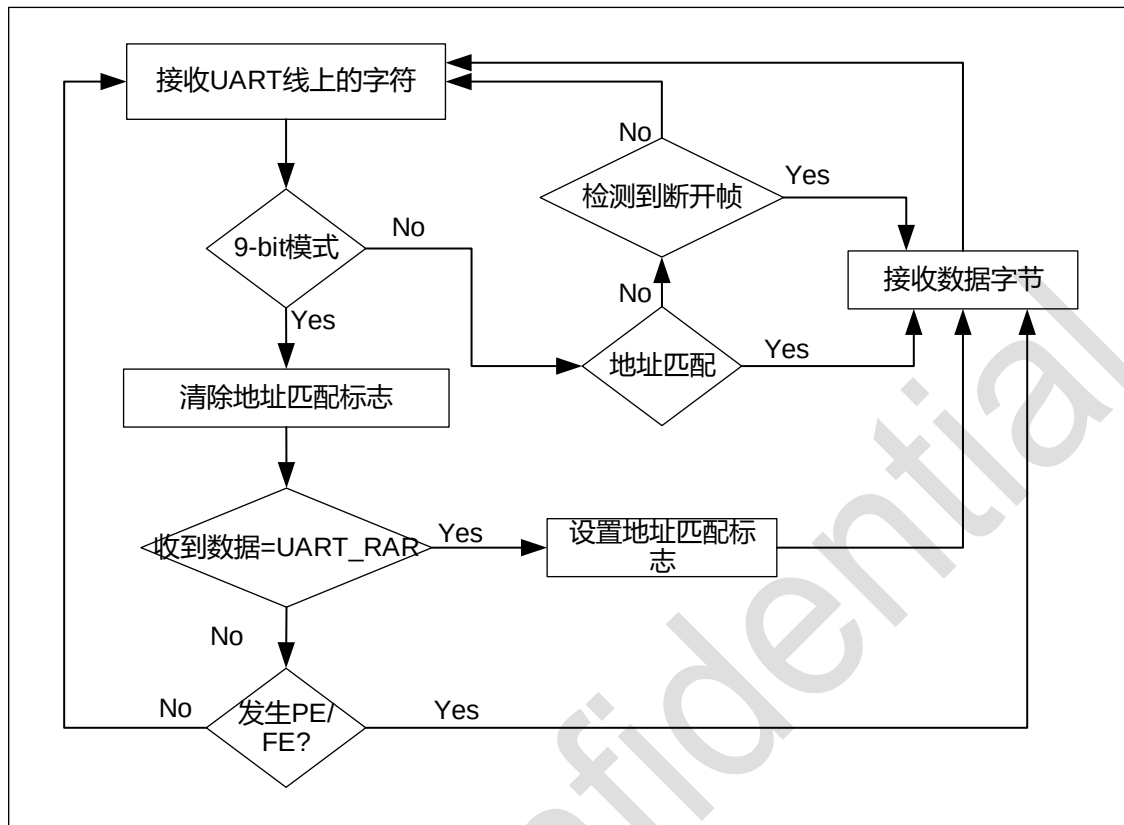


Figure 32-7 Hardware address matching receive mode

The UART receives characters regardless of whether the 9th bit of data is set to 1. If the 9th bit of the received character is set to 1, it clears the internal address match flag and then compares the received 8-bit character information with the address programmed in the UART_RAR register.

If the received address character successfully matches the address programmed in the UART_RAR register, the address match flag is set to 1, and the received character is pushed to the UART_RDR register, and the ADDR_RCVD bit in the UART_SR register is set to indicate that the address has been received, and the subsequent data byte (the 9th bit of the received character is set to 0) is pushed to UART_RDR.

If the address fails to match the UART_RAR register and (parity or a frame error is found in the received address character), the received address string is still pushed to the UART_RDR register, and the ADDR_RCVD and PE/FE error bits are set to 1.

If any interrupt character is received, UART treats it as a special character and pushes it to the UART_RDR register regardless of the address match flag.

Software address matching receiving mode

In this mode of operation, the UART does not perform address matching with the received address character of the UART_RAR register (the 9th bit of data is set to 1). The UART always receives 9 bits of data, shifted into the UART_RDR register. Whenever an address byte is received and indicated by the ADDR_RCVD bit in the status register, the address must be compared by the user himself.

32.3.4. UART baud rate

The baud rate of the UART is controlled by the PCLK and frequency division latch register (UART_BRR) and the fractional baud rate register UART_BRRF.

The baud rate is determined by:

- Serial Clock Operating Frequency (PCLK)
- Desired baud rate
- Baud rate generator divisor (consists of UART_BRR register)
- Acceptable baud rate error%

The equation for calculating the baud rate is as follows:

$$\text{Baud rate} = \text{serial clock operating frequency} / (16 * \text{DIVISOR}) \quad (1)$$

Where, DIVISOR-the number (hex) used to program the BRR.

Serial clock frequency – PCLK frequency of UART.

According to equation (1), DIVISOR can be calculated as: $\text{DIVISOR} = \text{serial clock frequency} / (16 * \text{baud rate})$

(The fractional part of the equation is placed in UART_BRRF)

Also from Equation (1), it can also be shown that serial clock frequency = baud rate * 16 * divisor

The error between the baud rate and the selected baud rate calibration is as follows:

Percentage error = $(| \text{baud rate} - \text{selected baud rate} |) / \text{baud rate} * 100\%$

Table 32-1 Error calculation when setting baud rate

Baud rate		fPCLK = 4MHz			fPCLK = 24MHz			fPCLK = 48MHz		
No.	kbps	Actual	Value programmed in the baud rate register	Error%	Actual	Value programmed in the baud rate register	Error%	Actual	Value programmed in the baud rate register	Error%
1	2.4	2.404	104	0.16 %	2.4	625	0.00 %	2.4	1250	0.00 %
2	9.6	9.615	26	0.16 %	9.615	156	0.16 %	9.615	312	0.16 %
3	19.2	19.231	13	0.16 %	19.231	78	0.16 %	19.231	156	0.16 %
4	57.6	62.5	4	8.51 %	57.692	26	0.16 %	57.692	52	0.16 %
5	115.2	125	2	8.51 %	115.385	13	0.16 %	115.385	26	0.16 %
6	230.4	250	1	8.51 %	250	6	8.51 %	230.769	13	0.16 %
7	460.8	Impossible	Impossible	Impossible	500	3	8.51 %	500	6	8.51 %
8	921.6	Impossible	Impossible	Impossible	1500	1	62.76 %	1000	3	8.51 %
9	2250	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible	3000	1	33.33 %
10	4500	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible

Notes:

1. The lower the clock frequency of the CPU, the lower the error at a particular baud rate. The upper limit of the achievable baud rate can be fixed with this data.
2. When configuring the clock, because the baud rate clock is divided on the system clock, there will be an error between the obtained clock frequency and the actual clock frequency. The configured clock error needs to be within 2% to work properly.

32.3.5. UART reception tolerance

The UART asynchronous receiver operates correctly only if the total clock system deviation is less than the tolerance of the UART receiver. The causes which contribute to the total deviation are:

- DTRA: deviation due to the transmitter error (which also includes the deviation of the transmitter's local oscillator)
- DQUANT: error due to the baud rate quantization of the receiver
- DREC: deviation of the receiver local oscillator
- DTCL: deviation due to the transmission line (generally due to the transceivers which can introduce an asymmetry between the low-to-high transition timing and the high-to-low transition timing).

$\text{DTRA} + \text{DQUANT} + \text{DREC} + \text{DTCL} < \text{UART receiver tolerance}$

For normal data reception, the tolerance of the UART receiver is equal to the maximum allowable variation of **96% to 105%**.

32.3.6. UART DMA communications

The UART may utilize DMA continuous communication. DMA requests for the Rx buffer and the Tx buffer are generated separately.

32.3.6.1. DMA transmission

Transmission with DMA is enabled by configuring UART_CR3. DMAT. When the transmit data register is empty, DMA transfers data from the specified memory area to the UART_DR register.

The configuration steps of DMA transmission are as follows:

- Configure SYSCFG_CFGR3 or SYSCFG_CFGR4 to select the DMA channel used by the USART.
- Configure the UART_TDR register address as the destination address for the DMA transfer. After each send data register empty event, data will be transferred to this address;

- Configure a memory (e.g. SRAM) address as the source address for DMA transfers. Loading read data from this memory into the UART_TDR register after each transmit data register empty event;
- Write the DMA control register to configure the number of transferred bytes;
- Write the DMA register to configure channel priority;
- Enable a DMA channel;
- When the transmission reaches the configured number of transmission bytes, the DMA controller generates an interrupt request.

32.3.6.2. DMA reception

Utilizing DMA reception is enabled by configuring UART_CR3.DMAR. For each byte received, the DMA transfers data from the UART_DR register to the designated SRAM area.

The configuration steps of DMA reception are as follows:

- Configure SYSCFG_CFGR3 or SYSCFG_CFGR4 to select the DMA channel used by the USART.
- Configure the UART_RDR register address as the source address for the DMA transfer. After each received data register non-empty event, data is transferred to memory by reading data from this UART_RDR register address;
- Configure the SRAM memory address as the destination address for the DMA transfer. After each reception of a data register non-empty event, data will be transferred from the UART_RDR register to this address;
- Write the DMA control register to configure the number of transferred bytes;
- Write the DMA register to configure channel priority;
- Enable a DMA channel;
- When the transmission reaches the configured number of transmission bytes, the DMA controller generates an interrupt request.

32.4. UART interrupts

Assertion of UART Interrupt Output Signal (INTR)-An interrupt occurs as soon as one of several preferential interrupt types is enabled and activated.

The following interrupt types can be enabled via the UART_CR2 register:

- Receiver data available (RXNEIE)
- Send register empty (TXEIE)
- Send hold register empty (in programmable TDR interrupt mode) (TDREIE)
- Busy error detection indication (BUSYERRIE)
- Receive Line State (LSIE)

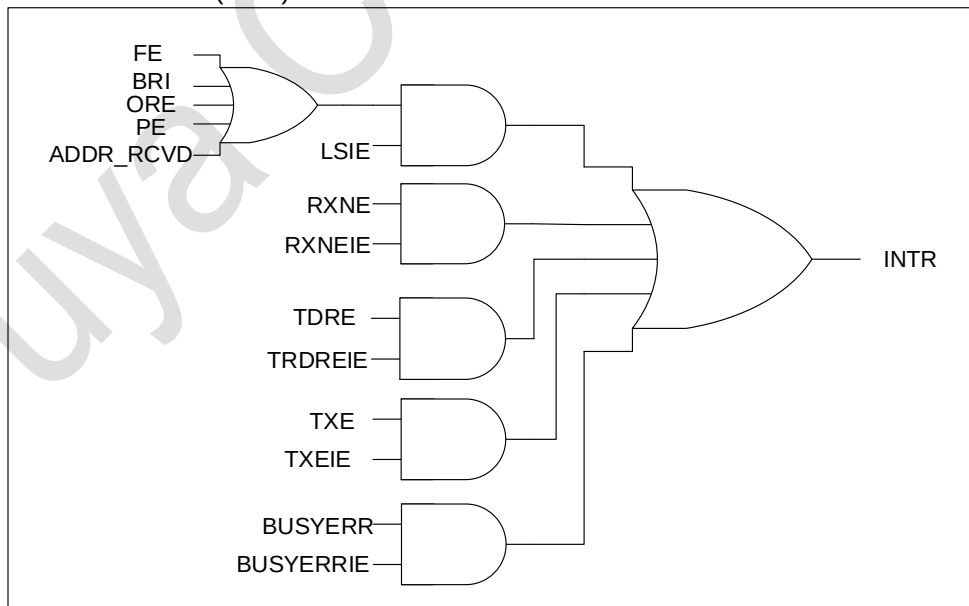


Figure 32-8 Interrupt Maps

These interrupt types are detailed in the table below.

Table 32-2 interrupt control function

Interrupt setting and clearing function		
Interrupt Type	Interrupt Source	Interrupt clearing control
Receive line status	Overflow/Parity/Frame Error Interrupt or Address Received Interrupt	Write 1 and clear 0 for the corresponding bit
Receive data available RXNE	Receiver data available	Read receiver buffer register
Send Hold Register Empty TDRE	Send hold register empty	Write TDR
Send register is empty TXE	Send register is empty	Write TDR
Busy error detection BUSYERR	When UART is BUSY (BUSY [0] is set to 1), the host attempts to write to the CR1 register.	Write 1 and clear 0 for the corresponding bit

32.5. UART registers

Terms: set = set to 1; Clear = Clear is 0;

32.5.1. UART Data Register (UART_DR)

Address offset: 0x00

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	DR[8:0]	RW	9'b0	Data value The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR). The RDR register is the data byte received on the serial input port in UART mode. The data in the status register (SR) is only valid if the Receive Non-Null (RXNE) bit in this register is set. The data in the RDR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overflow error. The TDR register is the data transferred on the serial output port in UART mode. The software guarantees that data can be written to the TDR again only if the TDR NULL (UART_SR.TDRE) bit is set. If the TDRE is set, writing a single character to the TDR will clear the TDRE. Any additional writes to the TDR before the TDRE is set again will cause the TDR data to be overwritten. The 9th bit applies only if UART_CR3.TX_MODE = 1.

32.5.2. UART baud rate register (UART_BRR)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	BRR	RW	16'b0	This register constitutes a 16-bit divisor, which contains the baud rate divisor of UART. The output baud rate is equal to sixteen times the serial clock (PCLK) frequency divided by the baud rate divisor, as follows: baud rate = (serial clock frequency)/(16 * divisor). Note that when the Divisor Latch Register (BRR) is set to zero, the Baud clock is disabled and no serial communication occurs. Furthermore, once the BRR is set, you should wait at least 8 clock cycles before sending or receiving data.

32.5.3. UART status register (UART_SR)

Address offset: 0x10

Reset value: 0x0000_0060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res.	Res.	Res.	Res	Res	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	BUSY_ER R	BUS Y	ADDR_RC VD	Res	TX E	TDR E	BRI	FE	PE	ORE	RXN E
					RC_W1	R	RC_W1		R	O	RC_W 1	RC_W 1	RC_W 1	RC_W 1	R

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10	BUSY_ERR	RC_W1	1'b0	Busy detection error, misoperation when busy is detected. 0: No busy misoperation error. 1: When UART is busy (UART_SR.BUSY is set to 1), the host attempts to write to the UART_CR1 register. This bit is cleared by writing 1.
9	BUSY	R	1'b0	UART Busy This indicates that a serial transfer is in progress, and when cleared it indicates that the UART is idle or inactive. This bit will be set to 1 (busy) in either of the following cases: ---Sending on serial interface ---Receiving on serial interface ---Transmitting data existing in TDR with non-zero baud divisor (BRR is not equal to 0) ---Receiving data existing in RDR Note: The UART busy bit may be cleared even if a new character may have been sent from another device. That is, if the UART has no data in the TDR and RDR, and there is no ongoing transmission, and the start bit of the new character has just arrived at the UART. This is because a valid start is not seen until the middle of the bit period, and this duration depends on the programmed baud divisor. 0 (IDLE): UART is IDLE or inactive 1 (busy): UART busy (active data transmission)
8	ADDR_RCVD	RC_W1	1'b0	Address receive. If 9-bit data mode is enabled (UART_CR3. M_E = 1), this bit is used to indicate that the 9th bit of the received data is set to 1. This bit can also be used to indicate whether the input character is an address or data. 0: indicates that the character is data. 1: Indicates that the character is an address. Note: The user needs to ensure that the interrupt is cleared before the next address byte arrives (this bit writes 1 clear 0). If there is a delay in clearing an interrupt, the software will not be able to distinguish between multiple address-dependent interrupts.
7	Reserved	-	-	Reserved

6	TXE	R	1'b1	Send is empty. This bit is set whenever no data is currently being sent and the TDR is empty. If TXEIE is enabled, a TXE interrupt is generated 0: Send is not empty 1: Send is empty
5	TDRE	R	1'b1	Send keeps register empty. This bit indicates that the TDR is null. This bit is set whenever data is transferred from the TDR to the transmitter shift register and no new data is written to the TDR. If TDRIE is enabled, an interrupt is generated. 0: TDR non-empty 1: TDR is empty
4	BRI	RC_W1	1'b0	Disconnect interrupt bit This is used to indicate that an interrupt sequence has been detected on the serial input data. If in UART mode, it is set whenever the serial input remains in a logic "0" state for longer than the sum of the start time + data bits + parity bits + stop bits. Writing 1 to this bit will clear the BRI bit. 0: No broken sequence detected 1: Broken sequence detected
3	FE	RC_W1	1'b0	Frame error bit. This is used to indicate that a frame error has occurred in the receiver. A frame error occurs when the receiver does not detect a valid stop bit in the received data. It should be noted that if a break occurs, the frame error (UART_SR.FE) bit will also be set, as indicated by the break interrupt (UART_SR.BRI) bit. This happens because the broken character implicitly generates a frame error by holding the input to a logical 0 longer than the duration of the character. Writing 1 to this bit will clear the FE bit. 0: no frame error 1: frame error
2	PE	RC_W1	1'b0	Parity error bit. If the parity enable (UART_CR1. PEN) bit is set, this register is used to indicate that a parity error has occurred at the receiver. It should be noted that if a disconnection occurs, in which case PE is also set if parity generation and detection is enabled (UART_CR1.PCE = 1) and parity is set to odd (UART_CR1.PS = 0). Writing 1 to this bit will clear the PE bit. 0: No parity error 1: parity error
1	ORE	RC_W1	1'b0	Overflow error bit. This is used to indicate the occurrence of an overflow error. This happens if a new data character is received before the previous data is read. The ORE bit is set when a new character reaches the receiver before the previous character is read from the RDR. When this happens, the data in the RDR is overwritten. Writing 1 to this bit will clear the ORE bit. 0: No overflow error 1: overflow error
0	RXNE	R	1'b0	Data ready bit. This is used to indicate that the receiver includes at least one character in the RDR. This bit is cleared when the RDR is read. 0: Data not ready 1: Data ready

32.5.4. UART control register 1 (UART_CR1)

Address offset: 0x14

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	MSBFIRST	SWAP	Res.	SBK	SP	PS	PCE	STOP	M[1:0]	

						RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	MSBFIRST	RW	1'b0	The most significant bit comes first. 0: After the start bit, the 0th bit data is transmitted and received; 1: After the start bit, send and receive the 5/6/7/8/9th bit data; This bit cannot be modified during data transfer.
8	SWAP	RW	1'b0	Interchangeable Tx/Rx pins 0: TX/RX pins are defined by standard pin mapping; 1: TX/RX pin interchange. At this time, it is used when cross-connecting other UARTs.
7	Reserved	-	-	Reserved
6	SBK	RW	1'b0	Disconnect the control bit. This is used to send the disconnect to the receiving device. If set to 1, the serial output will be forced into an interval (logic 0) state. The output line is forced low until the SBK bit is cleared. 0: Serial output is released for data transmission 1: Serial output is forced into interval state
5	SP	RW	1'b0	Fixed parity. Writable only when UART is not busy (UART_SR.BUSY is 0). This bit is used to force a fixed parity value. When PCE, PS, and SP are set to 1, a parity bit is sent and checked to be a logical 0. If PCE and SP are set to 1 and PS is a logical 0, a parity bit is transmitted and checked to be a logical 1. If this bit is set to 0, SP (Fixed Parity) is disabled. 0: Fixed parity disabled 1: Fixed parity enabled
4	PS	RW	1'b0	Even parity selection. Writable only when UART is not busy (UART_SR.BUSY is 0). This is used to select between even and odd parity when parity is enabled (PCE set to 1). If set to 1, an even number of logical "1s" is transmitted or checked. If set to zero, an odd number of logical "ones" is transmitted or checked. 0: odd check 1: even check
3	PCE	RW	1'b0	Parity enabled. Writable only when UART is not busy (UART_SR.BUSY is 0). This bit is used to control parity generation and detection in transmitted and received serial characters. 0: Disable parity 1: Enable parity
2	STOP	RW	1'b0	Number of Stop bits. Writable only when UART is not busy (UART_SR.BUSY is 0). This is used to select the number of stop bits for each character that the peripheral device will send and receive. If set to zero, a stop bit is transferred in the serial data. If set to 1 and the data bit is set to 5 (UART_CR1. M is set to 0), a half stop bit is sent. Otherwise, two stop bits are transmitted. Note that regardless of the number of stop bits selected, the receiver will only check the first Stop bit. Note: The stop bit duration of UART implementations may be longer due to the idle time inserted between characters and the baud clock divisor value in the transmission direction for some configurations; 0: 1 stop bit 1: 1.5/2 stop bits
1:0	M [1: 0]	RW	2'b0	Data length selection. Writable only when UART is not busy (UART_SR.BUSY is 0). When M_E in UART_CR3 is set to 0, this register is used to select the number of data bits for each character that the peripheral device

				will send and receive. 0x0: 5 data bits per character 0x1: 6 data bits per character 0x2: 7 data bits per character 0x3: 8 data bits per character

32.5.5. UART control register 2 (UART_CR2)

Address offset: 0x18

Reset value: 0x0000 0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	BUSYERRIE	LSIE	TDREIE	RXNEIE
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	Reserved
4	TXEIE	RW	1'b0	Enable send null interrupts. This is used to enable/disable the generation of a transport send null interrupt. 0: Disable sending null interrupt 1: Enable sending null interrupt
3	BUSYERRIE	RW	1'b1	Enable the BUSYERR state interrupt. This is used to enable/disable BUSYERR state interrupt generation. When UART is busy (UART_SR.BUSY set to 1), the host attempts to write to the UART_CR1 register. 0: Disable BUSYERR State Interrupt 1: Enable BUSYERR State Interrupt
2	LSIE	RW	1'b0	Enable receiver line status interrupt. This is used to enable/disable the generation of receiver line state interrupts. The interrupt flag is a combination of PE, FE, ORE, BRI, ADDR_RCVD interrupt flags. 0: Disable receiver line state interrupt 1: Enable receiver line state interrupt
1	TDREIE	RW	1'b0	Enable Transfer Keep Register Null interrupt. This is used to enable/disable the generation of a transfer hold register null interrupt. 0: Disable Transfer Hold Register Empty Interrupt 1: Enable Transfer Hold Register Empty Interrupt
0	RXNEIE	RW	1'b0	Enable the Receive Data Availability interrupt. This is used to enable/disable the received data available interrupt. 0: Disable Receive Data Interrupt 1: Enable Receive Data Interrupt

32.5.6. UART control register 3 (UART_CR3)

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Res.	Res.	Res.	Res.	Res.	Res.	Re s.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	DMA_SOFT_ACK	DMA_R	DMA_T	TX_MO DE	SEND_AD DR	ADDR_MAT CH	M_E
									RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6	DMA_SOFT_ACK	RW	1'b0	The software clears the DMA send request and the DMA receive request. 0: does not affect 1: Clear DMA send request and DMA receive request

5	DMAT	RW	1'b0	DMA is enabled when transmitting. 0: Interrupt is disabled 1: enable transmission DMA;
4	DMAR	RW	1'b0	DMA enable receiver 0: Interrupt is disabled 1: Enable receive DMA;
3	TX_MODE	RW	1'b0	Transfer mode control bit. This bit is used to control the type of transmission mode during the 9-bit data transmission. Value: 1: In this mode of operation, the transfer hold register (TDR) is 9 bits wide. The user needs to ensure that the address/data of the TDR register is written correctly. Address: Bit 9 set to 1 Data: Bit 9 set to 0 Note: The Transfer Address Register (UART_TAR) is not available for this mode of operation. 0: In this mode of operation, the Transfer Hold Register (TDR) is 8 bits wide. The user needs to program the address into the transfer address register (UART_TAR) and the data into the TDR register.
2	SEND_ADDR	RW	1'b0	Send the address control bit. This bit serves as a control knob for the user to determine when to send an address in transmission mode. 0: 9-bit character content from TDR register 1: 9-bit character content: The 9th bit is set to 1 and the remaining 8 bits will match what is being programmed in the "Transfer Address Register". Notes: 1. After the address character is issued, this bit is automatically cleared by the hardware. Users should not program this bit to 0. 2. This field is only applicable if the M_E bit is set to 1 and TX_MODE is set to 0.
1	ADDR_MATCH	RW	1'b0	Address matching pattern. This bit is used to enable address matching during reception. ■ In address match mode, UART will wait for an incoming character with the 9th bit set to 1. And further check whether the address matches the address programmed in the "Receive Address Match Register". If there is a match, the subsequent characters are considered valid data and UART starts receiving data. ■ In normal mode, the UART will start receiving data and 9-bit characters will form. The user is responsible for reading data and distinguishing between b/n addresses and data. 0: normal mode 1: Address matching mode Note: This field applies only if M_E is set to 1.
0	M_E	RW	1'b0	9 Data Enable This bit is used to enable the 9-bit data used to send and receive transmissions 0: Prohibit 9-bit data 1: Enable 9-bit data

32.5.7. UART receive address register (UART_RAR)

Address offset: 0x20

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	RAR[7:0]	RW	8'b0	Address match register in receive mode.

				If the 9th bit is set to 1 in the input character, the remaining 8 bits will be checked against the register value. If the match is successful, the subsequent characters with bit 9 set to 0 are treated as data bytes until the next address byte is received. Notes: 1. This register applies only if the "ADDR_MATCH" (UAR_CR3 [1]) and "M_E" (UART_CR3 [0]) bits are set to 1. 2. The RAR can be programmed at any point in time. However, the user must not change this register value while any reception is in progress.

32.5.8. UART transmit address register (UART_TAR)

Address offset: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	TAR[7:0]	RW	8'b0	This is the address match register in transfer mode. If the M_E bit is enabled and the "SEND_ADDR" (UART_CR3 [2]) bit is set to 1, then UART will send a 9-bit character with the 9th bit set to 1, and the remaining 8-bit address will be sent from this register. Notes: 1. This register is only used to send addresses. Normal data should be sent through the programmed TDR register. 2. After starting sending addresses on the UART serial channel, the hardware will automatically clear the "SEND_ADDR" bit

32.5.9. UART baud rate decimal register (UART_BRRF)

Address offset: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3:0	BRRF[3:0]	RW	4'b0	Baud rate fractional part. The decimal value is determined by $(BRRF)/(2^4)$.

33. Low-power universal asynchronous receivers/transmitters (LPUART)

The LPUART is an UART which allows bidirectional UART communications with a limited power consumption. UART communication up to 9600 baud/s is achievable with only a 32.768 kHz LSE clock. Higher baud rates can be reached when the LPUART is clocked by clock sources different from the LSE clock.

Even when the microcontroller is in low-power mode, the LPUART can wait for an incoming UART frame while having an extremely low energy consumption. The LPUART includes all necessary hardware support to make asynchronous serial communications possible with minimum power consumption.

It supports half-duplex single-wire communication and modem operation (CTS/RTS), and also supports multi-processor communication.

DMA (direct memory access) can be used for data transmission/reception.

33.1. LPUART main features

- AMBA APB interface
- Full-duplex asynchronous communication
- NRZ Standard Mode (Mark/Space)
- Programmable baud rate
- 32.768 kHz clock with baud rate range of 300~9600 baud/s. Higher baud rates require higher clock frequency support
- Dual clock domain: PCLK and dedicated core clock
- Data word length configurable (7 bit/8 bit/9 bit)
- Programmable data sequence, shifting MSB or LSB first
- Number of stop bits is configurable (1/2 bit)
- Single-wire Half-duplex communications
- Support continuous DMA transfer
- Independent enable for transmit and receive
- Independent polarity control for Tx/Rx signals
- Tx/Rx pins are interchangeable
- Hardware flow control for modems and RS-485 transceivers
- Transfer detection flag
 - Busy flag
 - End of transmission flags
- Parity control:
 - Generate parity bits when transmitted
 - Check parity of received data bytes
- 4 error flags
 - Overrun error
 - Noise detection
 - Frame error
 - Parity error
- Interrupt sources with flags:
 - CTS change
 - Transmit data register empty
 - Transmission complete
 - Receive data register full
 - Bus idle detected
 - Overrun error
 - Framing error
 - Noise operation
 - Error detection
 - Match address byte
- Multiprocessor communication: LPUART enters silent mode when addresses do not match
- Supports silent mode wake-up (idle frame/address tag detection)

33.2. LPUART functional description

33.2.1. LPUART block diagram

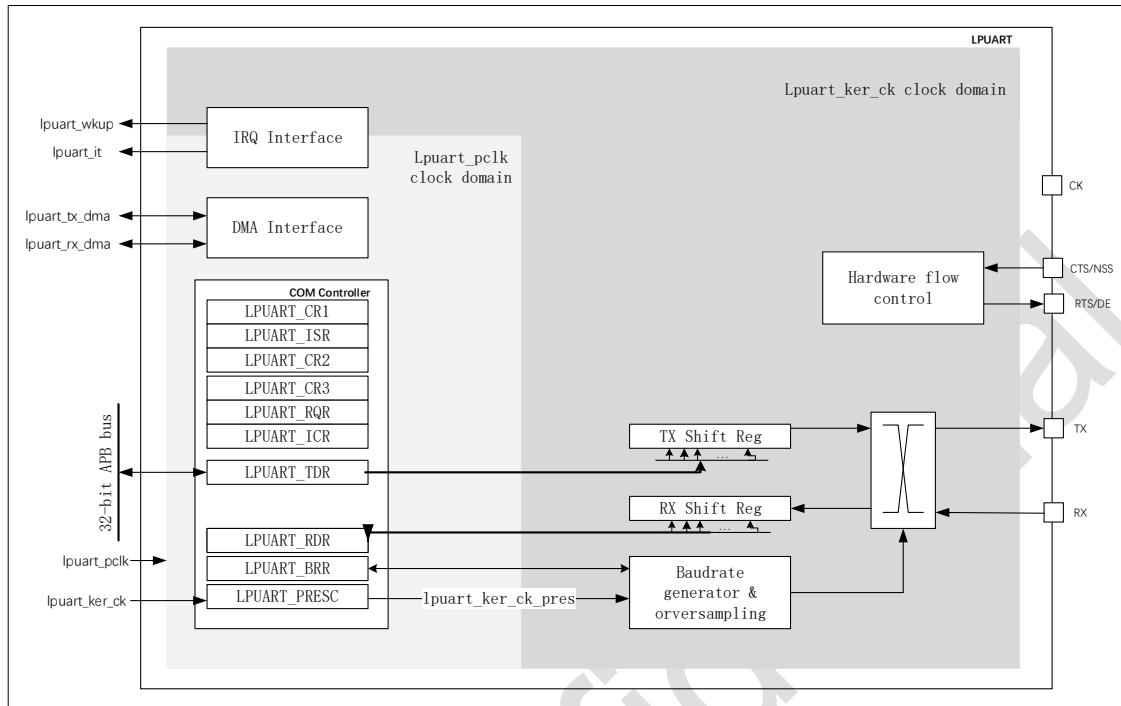


Figure 33-1 LPUART block diagram

33.2.2. LPUART signals

Any LPUART bidirectional communications require a minimum of two pins: Receive Data In (RX) and Transmit Data Out (TX).

- RX: Receive data serial input.
- TX: Transmit data output. If the transmitter is turned off, the output pin mode is determined by its I/O port configuration. If the transmitter is enabled but there is no data to send, the TX pin remains at a high level. In single-wire mode, this I/O is used for both sending and receiving data.

33.2.2.1. Hardware flow control mode

- CTS: transmission starts at low level, and transmission ends to high level
- RTS: Low level indicates that LPUART is ready to receive data

33.2.2.2. RS485 hardware flow control mode

- DE: Driver enable, activates transmission from external transceiver

Note: DE and RTS multiplex the same pin

33.2.3. LPUART Character Description

According to the {M1, M0} configuration, there are three data lengths of 7 bit/8 bit/9 bits.

M [1: 0] = 2 'b10, position 7;

M [1: 0] = 2 'b00: 8 bits;

M [1: 0] = 2 'b01: 9 position;

By default, the signal (TX or RX) is in a low state during start bit operation. It is in high state during the stop bit.

These values can be inverted, separately for each signal, through polarity configuration control.

Idle character definition: All 1s within 7-bit/8-bit/9-bit data length + stop bit time;

Break character definition: all zeros for 7 bit/8 bit/9 bit data length time; Disconnect the end of the frame and the transmitter inserts 2 stop bits. That is, 8-bit mode, the length of the disconnected frame is 10 bits 0; 9-bit mode, the disconnected frame length is 11 bits 0.

Transmission and reception are driven by a common baud rate generator. The transmission and reception clocks are generated when the enable bit is set for the transmitter and receiver, respectively. Below are the waveforms for each frame of different lengths.

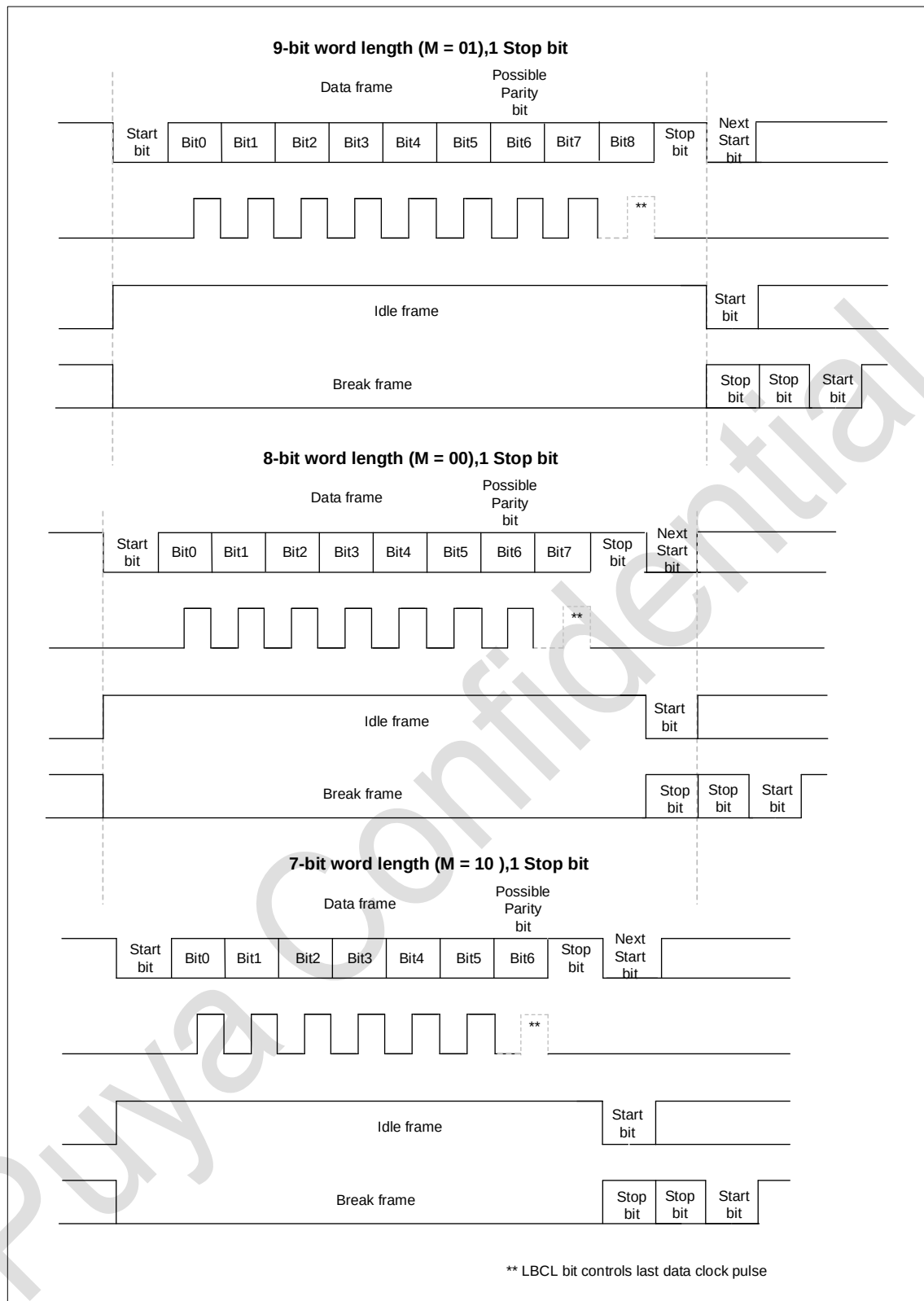


Figure 33-2 LPUART word length programming

33.2.4. LPUART transmission

The LPUART transmission steps are as follows.

33.2.4.1. Transmission steps

- Configure LPUART_CR1. {M1, M0} to define the transmission data width (7/8/9bits);
- Configure the LPUART_BRR register and select the baud rate;
- Configure the LPUART_CR2. STOP register to define the number of stop bits (1/2bits);
- Configure LPUART_CR1. UE = 1 to enable LPUART;
- When multi-processor transmission, configure LPUART_CR3. DMAT = 1 to enable DMA;
- Configuring TE = 1, hardware inserting idle frame timing;

- Write transfers data to the LPUART_TDR register (this operation clears the TXE bit). This operation may be repeated multiple times.
- After writing the last data to LPUART_TDR, the software waits for TC = 1, indicating that the last frame has been sent.
- The hardware generates the transmission timing:
 - Generate a start bit and start transmission
 - Transfer shift register output data (default LSB) to the TX pin
 - Stop bits (number sent according to configuration)
- After the last frame of data is transmitted, TC = 1 (when TXE = 1) is generated. When TCIE = 1, a TC interrupt is generated.

33.2.4.2. TC/TXE Timing

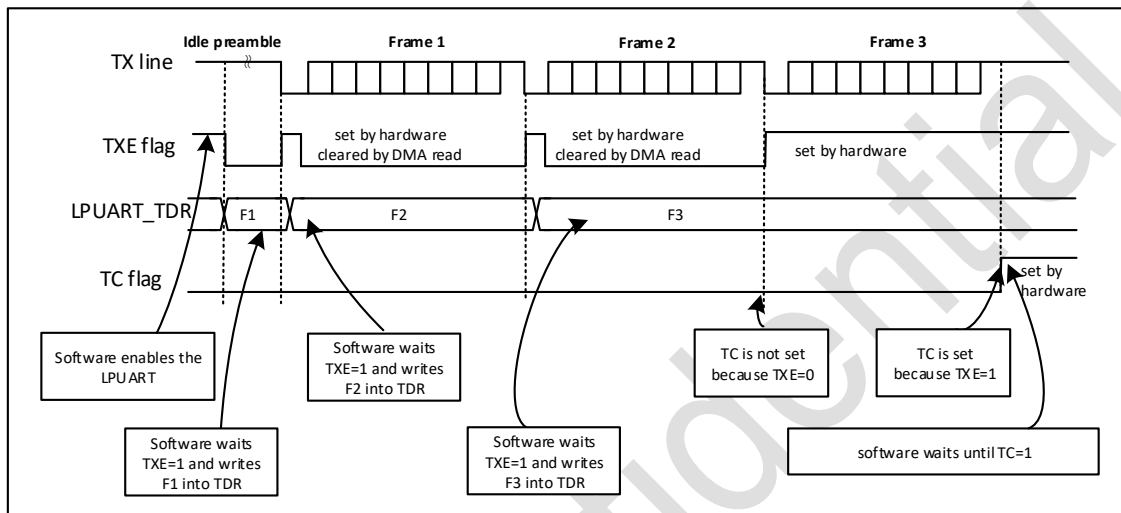


Figure 33-3 TC/TXE behavior during transmission

33.2.4.3. Special frame

Disconnect frame

Setting SBKRQ = 1 will transmit a disconnect frame after the current transmission is complete. The length of the break frame depends on the configuration of the {M1,M0} registers. After the disconnected frame is sent, the hardware will clear the SBKRQ.

Idle line detected

Set TE = 1 to send an idle frame before the first data frame.

33.2.5. LPUART reception

LPUART can receive 7-bit, 8-bit, or 9-bit data words, depending on the {M1,M0} bits in the LPUART_CR1 register.

33.2.5.1. Start bit detection

After a falling edge is detected on the RX, the hardware starts detecting the start bit. When a 0 is detected in the middle of the bit, it is determined that the bit is the start bit. If 1 is detected, the noise error flag (NE) is set, the current start bit is invalid, and the start bit is re-detected.

33.2.5.2. Data character reception

Reception Steps

Character reception configuration steps:

- Configure LPUART_CR1. {M1,M0}, select a reception length;
- Configure the LPUART_BRR register and select the baud rate;
- Configure USART_CR2. STOP and select 1-bit or 2-bit stop bits;
- Configure LPUART_CR1. UE = 1 to enable LPUART;
- When the multi-processor transmits, configure LPUART_CR3. DMAR = 1 to enable DMA reception;
- Configure LPUART_CR1. RE, enable reception, and start detecting the start bit;

When the character is received:

- Set RXNE. Indicates that the content of the shift register has been transferred to RDR, meaning the data can be read out.
- When RXNEIE = 1, an interrupt is generated;
- During reception, if a framing error, noise error, or overflow error is detected, the corresponding error flag is set.

- Multiprocessor communication:
 - RXNE is set after each byte is received and cleared by a DMA read operation on the receive data register.
- In single buffer mode:
 - Reading the LPUART_RDR register in software clears the RXNE bit. The RXNE bit must be cleared before the end of the next character reception to avoid an overflow error.

33.2.5.3. Special frames and error handling

-Break character

When a broken frame is received, LPUART treats it as an error frame.

-Idle character

When an idle frame is detected, it is processed as a normal frame, and if IDLEIE = 1, an interrupt is generated.

- Overrun error

If the RXNE is not reset and another character is received, an overflow error occurs. Data can not be transferred from the shift register to the RDR register until the RXNE bit is cleared. The RXNE flag is set after each byte received. If the next data has been received or the previous DMA request has not been executed, the RXNE flag is still set and an overflow error occurs.

When an overflow error occurs, the following error handling is performed:

- SET ORE;
- RDR content is not lost. Reading the LPUART_RDR register can still normally read out the previous data.
- The shift register content is overwritten, and subsequently received data is lost.
- If RXNEIE=1 or EIE=1, an interrupt is generated.
- Writing ORECF=1 clears the ORE bit.

ORE is set to indicate that at least 1 piece of data has been lost. There are the following two situations:

- If RXNE = 1, the last valid data is stored in the receive register (RDR) and can be read.
- If RXNE=0, it means the last valid data has been read, so there is no data to read in the RDR. This situation occurs when the last valid data in the RDR register is read while new (and lost) data is received.

33.2.5.4. Clock source selection

The clock source of LPUART is configured by the RCC_CCIPR.LPUARTxSEL register. Before configuring LPUART_CR1.UE = 1, you must configure the clock source.

The selection of the clock source determines the communication speed range.

LPUART has dual clock domains, one for the kernel clock (LPUART_ker_ck) and the other for the system clock (LPUART_pclk). When used to wake up from low power, the kernel clock source is configured by the RCC_CCIPR.LPUARTxSEL register.

When the dual clock domain is not supported, the kernel clock is the same as the system clock.

When the MCU enters low power mode and the core clock source selects low frequency, LPUART can receive data and wake up the MCU, and then the software or DMA can read the data in LPUART_RDR.

33.2.5.5. Framing error

A framing error is detected when the stop bit is not recognized on reception at the expected time, following either a de-synchronization or excessive noise.

When a frame error is detected, handle it as follows:

- The hardware sets the FE bit;
- Invalid data is transferred from the shift register to the LPUART_RDR register;
- In the case of single-byte communication, no interrupt is generated. However, because the NE flag bit and the RXNE flag bit are set at the same time, the FE bit has a rising edge. In the case of multiprocessor communication, if LPUART_CR3. EIE = 1, an interrupt will be generated.

Write LPUART_ICR.FECF = 1 to clear the FE flag.

33.2.5.6. Configure the number of stop bits

Configurable bits 1/2 stop bits:

- 1 stop bit: sampling stop bit at 8/9/10 sampling point;
- 2 stop bits: Sample the second stop bit after sampling the first stop bit. The RXNE and FE flags are set after detecting the end of the 2nd stop bit. When a frame error occurs, the first stop bit is not detected.

33.2.6. LPUART baud rate generation

Transmit and receive are configured with the same baud rate.

$$\text{Tx/Rx baud rate} = \frac{256 \times \text{Lpuart\_ker\_ck\_pres}}{\text{LPUARTDIV}}$$

LPUARTDIV is configured in the LPUART_BRR register. Do not change the value of the baud rate while the communication is in progress. The LPUART_BRR register value cannot be lower than 0x300.

The frequency range of fCK is 3 ~ 4096 x * baud rate.

When the LPUART clock source is LSE, the maximum baud rate is 9600 baud rate. Higher baud rates can be achieved when LPUART is clocked by a different clock source than the LSE clock. For example, with an LPUART clock source frequency of 100 MHz, the maximum achievable baud rate is approximately 33 Mbaud.

Table 33-1 Error calculation of programmed baud rate at lpuart_ker_ck_pres = 32,768 kHz

Baud rate	Lpuart_ker_ck_pres = 32,768 kHz		
Expected value	Actual value	Value programmed in the baud rate register	Error%
0.3 Kbps	0.3 Kbps	0x6D3A	0
0.6 Kbps	0.6 Kbps	0x369D	0
1200 Bps	1200.087 bps	0x1B4E	0.007
2400 Bps	2400.17 bps	0xDA7	0.007
4800 Bps	4801.72 bps	0x6D3	0.035
9600 Bps	9608.94 bps	0x369	0.035

Table 33-2 Error calculation of programmed baud rate when lpuart_ker_ck_pres = 100MHz

Baud rate	Lpuart_ker_ck_pres = 100 MHz		
Expected value	Actual value	Value programmed in the baud rate register	Error%
38400 baud	38400.04 baud	A2C2A	0.0001
57600 baud	38400.04 baud	6C81C	0.0001
115200 baud	115200.12 baud	3640E	0.0001
230400 baud	230400.23 baud	1B207	0.0001
460800 baud	460804.61 baud	D903	0.001
921600 baud	921625.81 Porter	6C81	0.0028
4000000 baud	4000000.00 baud	1900	0
10000000 baud	10000000.00 Baud	A00	0
20000000 baud	20000000.00 baud	500	0
33000000 baud	33032258.06 Porter	307	0.1

33.2.7. LPUART reception tolerance

The LPUART asynchronous receiver operates correctly only if the total clock system deviation is less than the tolerance of the LPUART receiver. The causes which contribute to the total deviation are:

- DTRA: deviation due to the transmitter error (which also includes the deviation of the transmitter's local oscillator)
- DQUANT: error due to the baud rate quantization of the receiver
- DREC: deviation of the receiver local oscillator
- DTCL: deviation due to the transmission line (generally due to the transceivers which can introduce an asymmetry between the low-to-high transition timing and the high-to-low transition timing).

Need to meet: DTRA + DQUANT + DREC + DTCL + DWU < LPUART receiver tolerance

Where DWU is the error caused by sampling point bias when waking up using low power mode.

The LPUART receiver can receive data correctly at the specified maximum tolerance (theoretical value)

Table 33-3 LPUART Reception Tolerance

{M1, M0} bits	768 < BRR < 1024	1024 < BRR < 2048	2048 < BRR < 4096	4096 ≤ BRR
8 bits ({M1, M0} = 00), 1 stop bit	2 %	2.8 %	3.90 %	4.5 %
9 bits ({M1, M0} = 01), 1 stop bit	2 %	2.3 %	3.53 %	4.4 %
7 bits ({M1, M0} = 10), 1 stop bit	2.4 %	2.9 %	4.7 %	4.4 %
8 bits ({M1, M0} = 00), 2 stop bits	1.8 %	2.9 %	3.8 %	4.3 %
9 bits ({M1, M0} = 01), 2 stop bits	2 %	2.4 %	3.4 %	3.8 %

7 bits ({M1, M0} = 10), 2 stop bits	2.4 %	2.9 %	4 %	4.4 %

Note: When the received frame contains an idle frame that is exactly 10/11/9 bits (corresponding to M = 'b00/11/10'), the data specified in the table may vary slightly under special circumstances.

33.2.8. LPUART multiprocessor communication

Multi-processor communication can be realized through LPUART (connecting several LPUARTs in a network). For example, a certain LPUART device may be the master, and its TX output is connected to the RX input of other LPUART slaves; The respective TX output logic of the LPUART slaves is coupled together and connected to the RX input of the master.

An unaddressed device may configure the MME register to be put into silent mode. In silent mode:

- No receive status bits will be set;
- Disable all reception interruptions;
- LPUART_ISR.RWU = 1. By configuring LPUART_RQR.MMRQ, hardware and software can control LPUART to enter silent mode;

Depending on the value of LPUART_CR1.WAKE, LPUART can enter or exit silent mode in two ways:

- WAKE = 0: Idle bus detection;
- WAKE = 1: address flag detection;

33.2.8.1. Idle bus detection

When MMRQ writes 1 and RWU automatically sets 1, LPUART enters silent mode.

When an idle frame is detected, LPUART wakes up. The RWU is then cleared by hardware, but LPUART_ISR.IDLE is not set.

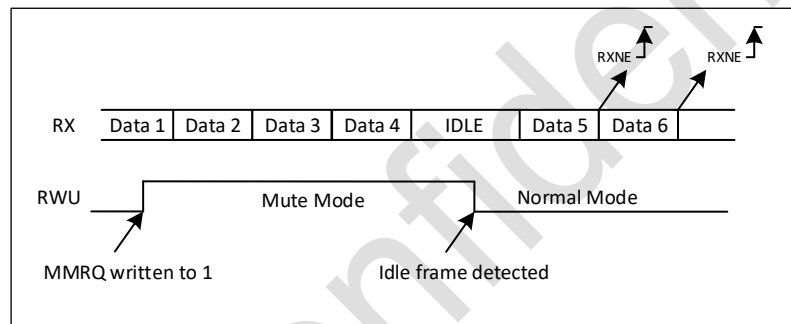


Figure 33-4 Silent Mode Detected by Bus Frame

Notes:

If the MMRQ position 1 is placed when the IDLE character has passed, the silent mode will not be entered (RWU is not set to 1).

If the LPUART is activated while the line is IDLE, the idle state is detected after the duration of one IDLE frame (not only after the reception of one character frame).

33.2.8.2. 4-bit/7-bit address flag detection

In this mode, if MSB = 1, the byte is considered an address, otherwise it is considered data. In one address byte, the address of the target receiver is placed in four or seven LSBs (several selected by the ADDM7 register). This 4-bit/7-bit address is compared by the receiver to the address configured in the ADDR in the LPUART_CR2 register.

If the received byte does not match the configured address, LPUART enters silent mode. At this time, the hardware sets the RWU. The RXNE flag is not set for this address byte and no interrupt nor DMA request is issued as the LPUART would have entered mute mode.

When MMRQ is configured to 1, LPUART also enters silent mode and sets RWU = 1.

If the received byte matches the configured address, LPUART exits silent mode. At this time, the RWU is cleared, and then data is normally received. Since RWU has been cleared, this condition will set the RXNE bit.

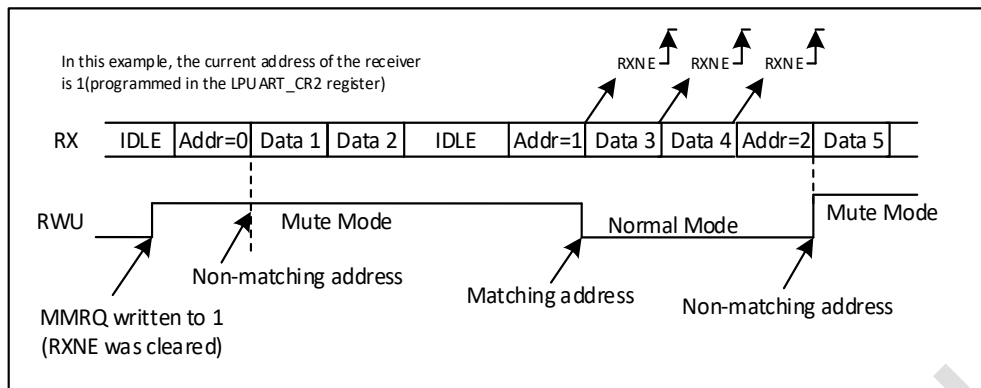


Figure 33-5 Silent mode detected by address flags

33.2.9. LPUART parity

Set LPUART_CR1.PCE = 1 to enable parity control (generate a parity bit when sending and perform parity check when receiving). According to the frame length defined by {M1, M0} bits, the supported LPUART frame formats are shown in the following table:

Table 33-4 supported LPUART frame formats

{M1, M0} bits	PCE bit	LPUART frame
00	0	Start bit 8 bit data Stop bit
00	1	Start bit 7 bit data Parity bit Stop bit
01	0	Start bit 9-bit data Stop bit
01	1	Start bit 8 bit data parity bit stop bit
10	0	Start bit 7-bit data Stop bit
10	1	Start bit 6 bit data parity bit stop bit

33.2.9.1. Even parity

The even check bit causes 6, 7, or 8 pieces of LSB data in a frame and the number of "1" in the check bit to be even.

33.2.9.2. Odd parity

The odd check bits make 6, 7, or 8 LSB data in a frame and the number of "1" in the check bits odd.

33.2.9.3. Receipt check

Set PE = 1 in case of parity error and generate interrupt when PEIE = 1. The software writes PECF = 1 to clear PE.

33.2.9.4. Transmit parity bit

The MSB is replaced by a parity bit when transmitted.

33.2.10. LPUART single-line half-duplex communication

When LPUART_CR3.HDSEL = 1 is configured, LPUART enters single-wire half-duplex communication mode.

In this mode:

- TX and RX are connected inside the module;
- The RX pin is not used;
- Release the TX pin as standard I/O when there is no data transfer

33.2.11. DMA continuous communication

LPUART can utilize DMA continuous communication. The DMA requests for the receive buffer and the transmit buffer are generated separately.

33.2.11.1. Transmission using DMA

DMA transmission is enabled by configuring LPUART_CR3.DMAT. When TXE = 1, the DMA transfers data from the designated SRAM area to the LPUART_DR register.

The configuration steps of DMA transmission are as follows:

- Configure SYSCFG_CFGR3 or SYSCFG_CFGR4 to select the DMA channel used by LPUART
- Configure the LPUART_TDR register address as the destination address for the DMA transfer. After each TXE event, data will be transferred to this address;
- Configure a memory (e.g. SRAM) address as the source address for DMA transfers. Loading read data from this memory into the LPUART_TDR register after each TXE event;
- Write the DMA control register to configure the number of transferred bytes;
- Write the DMA register to configure channel priority;
- Depending on the application, the configuration generates a DMA interrupt when the transmission is half or all complete;

- Write `TCIF = 1` to clear the `LPUART_ISR.TC` flag;
- Enable a DMA channel;

When the transmission reaches the configured number of transmission bytes, the DMA controller generates an interrupt request.

When the DMA has written all the data to be transmitted, set `TCIF = 1`. Then the software needs to wait for `TC = 1`, indicating that LPUART has sent the last frame of data.

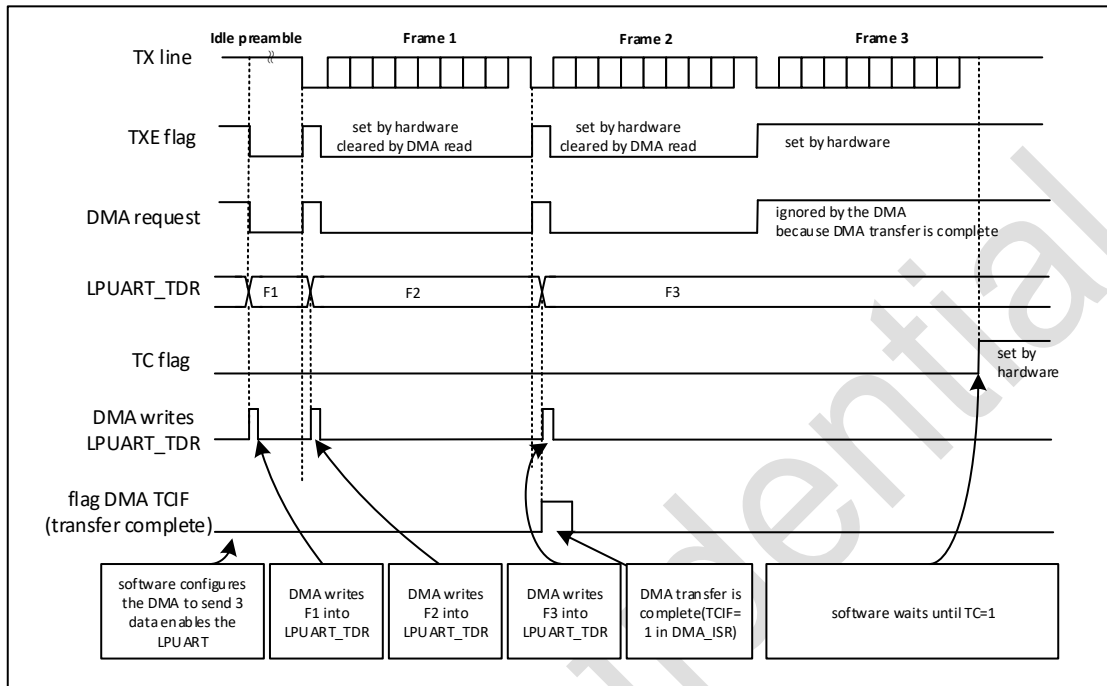


Figure 33-6 LPUART transmitted with DMA

33.2.11.2. Reception using DMA

DMA reception is enabled by configuring `LPUART_CR3.DMAR`. For each byte received, the DMA transfers data from the `LPUART_RDR` register to the designated SRAM area.

The configuration steps of DMA reception are as follows:

- Configure `SYSCFG_CFGR3` or `SYSCFG_CFGR4` to select the DMA channel used by LPUART.
- Configure the `LPUART_RDR` register address as the source address for the DMA transfer. After each `RXNE` event, data will be transferred to memory by reading data from this `LPUART_RDR` register address;
- Configure the SRAM memory address as the destination address for the DMA transfer. After each `RXNE` event, data will be transferred from the `LPUART_RDR` register to this address;
- Write the DMA control register to configure the number of transferred bytes;
- Write the DMA register to configure channel priority;
- Depending on the application, the configuration generates a DMA interrupt when the transmission is half or all complete;
- Enable a DMA channel;

When the transmission reaches the configured number of transmission bytes, the DMA controller generates an interrupt request.

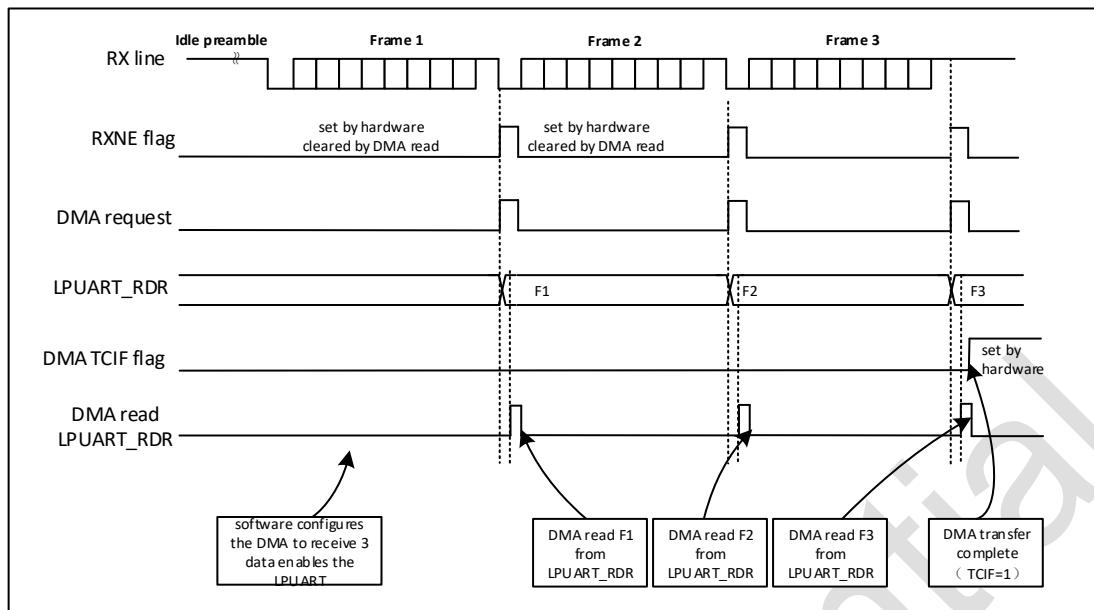


Figure 33-7 LPUART receives with DMA

33.2.11.3. Error flags and interrupts

In multiprocessor communication mode, any error generated during transmission will generate an error flag after the current byte transmission is completed, and an interrupt will be generated if the corresponding interrupt is enabled.

33.2.12. RS232 hardware flow control and RS485 driver enablement

The serial data flow of the two LPUART modules can be controlled through CTS and RTS.

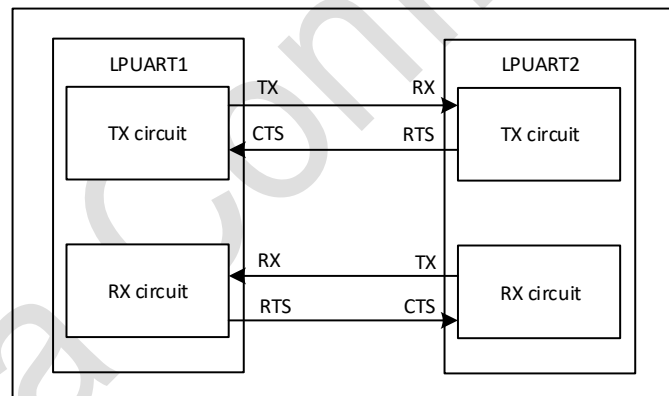


Figure 33-8 Hardware Flow Control

33.2.12.1. RS232 RTS Flow control

RTS flow control is enabled by configuring LPUART_CR3. RTSE = 1. When LPUART is ready to receive data, pull RTS low. When the receive register is full, the RTS pulls high (invalid), indicating that the current frame is the last frame.

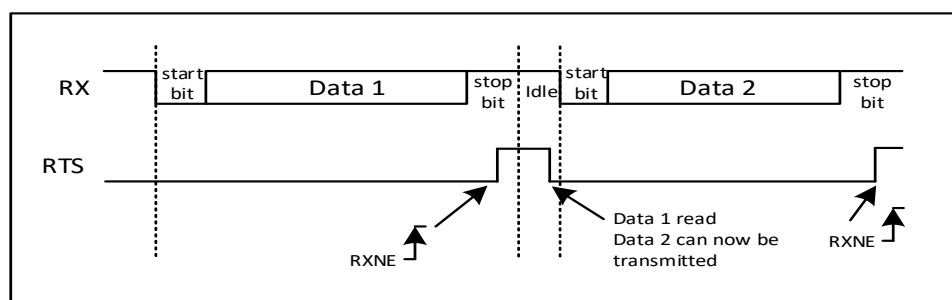


Figure 33-9 RTS flow control

33.2.12.2. RS232 CTS Flow control

CTS flow control is enabled by configuring LPUART_CR3. CTSE = 1. When enabled, the CTS signal is

detected before transmitting the next frame. If CTS = 0 (valid), transmit the next data (assuming there is data to be transmitted). If CTS = 1 (invalid), the current transmission is complete (before the stop bit).

When CTSE=1, the hardware sets CTS as soon as the CTS signal toggles. Indicates that the receiver is ready to communicate or not. An interrupt is generated if the CTSIE bit =1.

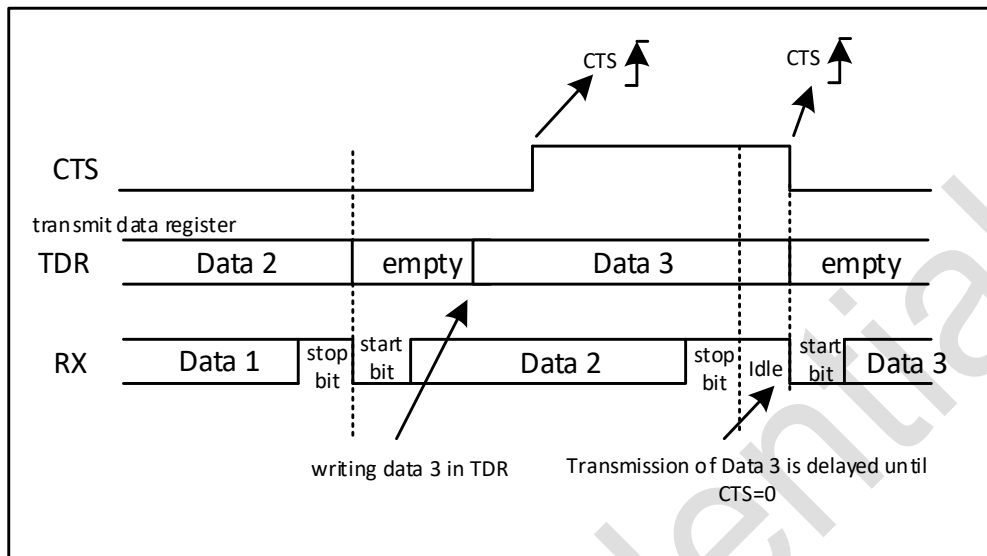


Figure 33-10 CTS flow control

Note: CTS maintains at least 3 LPUART source clocks high before the end of the current frame. CTSCF is also required to maintain at least $2 \times \text{PCLK}$ cycles.

33.2.12.3. RS485 Driver Enable

Configure LPUART_CR3. DEM = 1 to enable the DE function. With the DE signal, the user can activate the external transmission control. The effective polarity of the DE signal is configured by the LPUART_CR3.DEP register.

The valid time is configured by LPUART_CR1. DEAT and is the time between the valid edge of the DE signal and the start of the start bit.

The release time is configured by LPUART_CR1. DEDT and is the time between the last stop bit and the DE is invalidated.

The specific time is calculated as follows, where $P = \text{BRR} [20:11]$:

- DE insertion time:
 - $(1 + (\text{DEAT} \times P)) \times \text{fCK}$, $P \neq 0$
 - $(1 + \text{DEAT}) \times \text{fCK}$, $P = 0$
- DE release time:
 - $(1 + (\text{DEDT} \times P)) \times \text{fCK}$, $P \neq 0$
 - $(1 + \text{DEDT}) \times \text{fCK}$, $P = 0$

33.2.13. LPUART low-power modes

LPUART supports low-power modes, enabling data transmission even when PCLK is absent.

When UESM = 1, LPUART can wake up the MCU in low power mode.

33.2.13.1. Generate an interrupt wake-up request according to WUS

When a wake-up event is detected, the hardware generates a WUF flag. If WUFIE = 1, a WUF interrupt is generated. In this case, setting the WUF flag to 1 is sufficient to wake up the MCU from low power mode.

Notes:

- Before entering low power mode, ensure that there is no LPUART transmission;
- Regardless of whether the MCU is in low power mode or working mode, whenever a wake-up event is detected, the WUF flag will be set;
- After initializing and enabling LPUART reception, it is necessary to confirm whether the LPUART is truly enabled through the REACK bit, and then configure to enter the low power mode;
- When DMA reception is enabled, DMA reception must be turned off before entering low power mode; Then wake up from low power and re-enable DMA;

33.2.13.2. Entering low power mode from silent mode

If the LPUART is put into Mute mode before entering low-power mode:

- Silent mode cannot be awakened with idle frames because idle frame detection is not supported in low power mode;
- If the silent mode is waked up by address matching, the address matching method is also used to wake up from the low power mode. When entering low power, if RXNE = 1, LPUART will not enter the low power mode, but will remain in the silent mode;

33.2.13.3. Kernel clock off in low power mode

If the core clock is also turned off in low-power mode, the LPUART will generate a signal to enable the core clock when a falling edge is detected on the RX line.

If the wake-up event LPUART_CR3. WUS is configured to address match:

- After the core clock is enabled, if the addresses match after receiving one frame, the MCU wakes up and starts normal data reception;

The following figure shows the address verification match after the wake-up event:

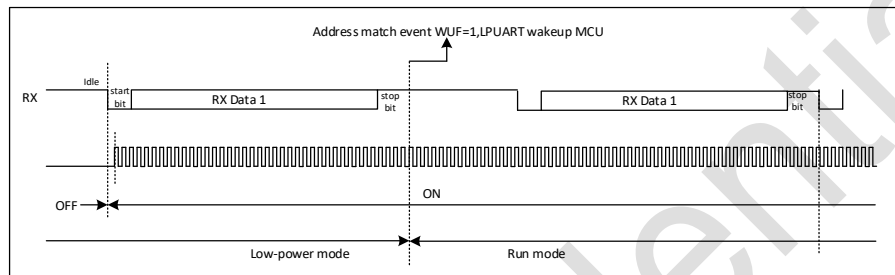


Figure 33-11 LPUART address matching wakeup

- If the address validation does not match after the wake-up event, the core clock shuts down again, the MCU does not wake up, and the system remains in low power mode.

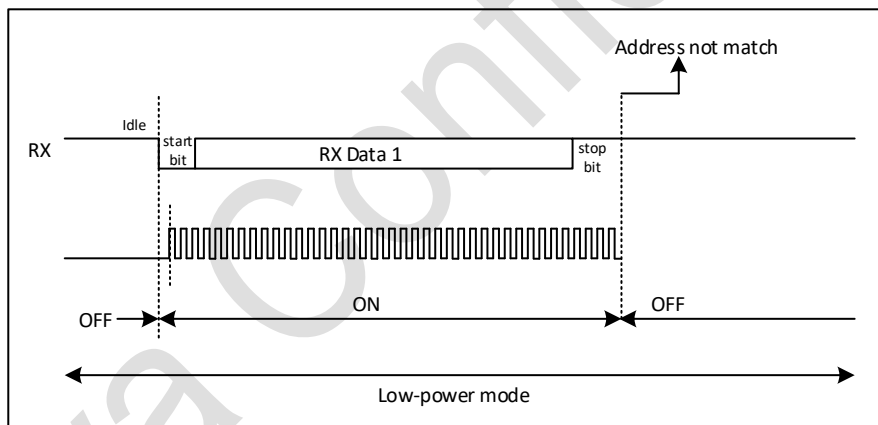


Figure 33-12 LPUART address does not match and cannot wake up

Note: When using address matching or any received frame as a wake-up event, the above two diagrams apply. If the wake-up event is start bit detection, the LPUART will send a wake-up event to the MCU at the end of the start bit.

33.3. LPUART interrupts

Table 33-5 LPUART interrupts

Interrupt event	Event flag	Enable Control bit	Interrupt zero-ing	Is the interrupt signal valid	
				Interrupts and events	Wake-up
Transmit data register empty	TXE	TXEIE	Write data to TDR	YES	NO
CTS interrupts	CTSIF	CTSIE	Software configuration CTSCF = 1	YES	NO

Transmission is complete	TC	TCIE	Write data to TDR or TCCF = 1	YES	NO
Receive register is not empty (read data is ready)	RXNE	RXNEIE	Reading RDR registers	YES	YES
Overrun error	ORE	RXNEIE	ORECF = 1	YES	NO
Idle line detected	IDLE	IDLEIE	IDLECF = 1	YES	NO
Parity error	PE	PEIE	PECF = 1	YES	NO
Noise, overrun, and framing errors during multiprocessor communication	NE/ORE/FE	EIE	NECF = 1 clears NE; ORECF = 1 clear ORE; FECF = 1 clear FE;	YES	NO
Match address byte	CMF	CMIE	CMCF = 1	YES	NO
Wake up from low power mode	WUF	WUFIE	WUCF = 1	YES	YES

33.4. LPUART registers

33.4.1. LPUART control register 1 (LPUART_CR1)

Address offset: 0x00

Reset value: 0x0000_0000

When configuring the DEAT and DEDT bits of this register, to improve configuration efficiency, all other registers in LPUART_CR1 except UE need to be configured at once. If this register needs to be configured multiple times, a 5 LPUART core clock cycle interval is required between each two configurations.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	Res.	Res.	DEAT[4:0]				DEDT[4:0]					
			RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	Reserved
28	M1	RW	0	The value of {M1, M0} configures the length and is set or cleared by the software. 2'b00: 1 start bit, 8 data bits, n stop bits; 2'b01: 1 start bit, 9 data bits, n stop bits; 2'b10: 1 start bit, 7 data bits, n stop bits; This bit can only be written when LPUART is not enabled (UE = 0).
27:26	Reserved	-	-	Reserved
25:21	DEAT[4:0]	RW	5'h0	This register configures the time between the active and start bit of the drive enable (DE) signal. Use the sampling time as the unit. If the DE functionality is not supported, the bit is reserved. This bit can only be written when LPUART is not enabled (UE = 0)
20:16	DEDT[4:0]	RW	5'h0	This register configures the time between sending the frame stop bit and the DE signal being invalidated. Use sampling time as the unit (1/8 or 1/16 bit period, depending on the oversampling rate). If there is a write LPUART_TDR register request to send within the DEDT time, the data will only be sent when both DEDT and DEAT have ended. If the DE functionality is not supported, the bit is reserved. This bit can only be written when LPUART is not enabled (UE = 0)
15	Reserved	-	-	Reserved
14	CMIE	RW	0	Character match interrupt enabled. This bit is set or cleared by software.

				0: interrupt disabled; 1: CMF interrupt enable;
13	MME	RW	0	Silent mode enabled. 0: The receiver is operating in the working mode; 1: The receiver switches between working mode and silent mode;
12	M0	RW	0	The word length is configured in conjunction with M1. This bit can only be written when LPUART is not enabled (UE = 0)
11	WAKE	RW	0	Receive wakeup mode. Wake up mode from silent mode. Set and cleared by software. 0: Idle frame wake-up; 1: Address flag wake up; This bit can only be written when LPUART is not enabled (UE = 0)
10	PCE	RW	0	Parity control enable 0: parity disabled; 1: parity enabled; Parity bit: the 9th bit of 9-bit data; 8th bit of 8-bit data; The 7th bit of the 7bit data. This bit can only be written when LPUART is not enabled (UE = 0)
9	PS	RW	0	Parity selection Set and cleared by software. 0: even check; 1: odd check This bit can only be written when LPUART is not enabled (UE = 0)
8	PEIE	RW	0	PE interrupt enable Set and cleared by software. 0: Interrupt is disabled 1: PE interrupt enable;
7	TXEIE	RW	0	Transfer register null interrupt enabled. Set and cleared by software. 0: Interrupt is disabled 1: TXE interrupt enable;
6	TCIE	RW	0	Transmission complete interrupt enable Set and cleared by software. 0: Interrupt is disabled 1: TC interrupt enable;
5	RXNEIE	RW	0	Receive register non-empty interrupt enable; Set and zeroed by software. 0: Interrupt is disabled 1: ORE or RXNE interrupt enable;
4	IDLEIE	RW	0	IDLE interrupt enable Set and cleared by software. 0: Interrupt is disabled 1: IDLE interrupt enable;
3	TE	RW	0	Transmitter enable 0: Transmission prohibited 1: Transmission enabled
2	RE	RW	0	Receiver enable 0: Reception prohibited; 1: Receive enable, start detecting the start bit;
1	UESM	RW	0	Stops LPUART wake-up enable in low power mode. This register is 1, and LPUART can wake up the MCU if the LPUART clock selects LSI or LSE. 0: LPUART cannot wake up MCU; 1: LPUART mode wakes up the MCU. When enabled, the LPUART clock source selects LSI or LSE. After waking up from low power mode, this bit is cleared.
0	UE	RW	0	LPUART enable When this bit is cleared the LPUART prescalers are stopped. The configuration of LPUART remains, but all status flags, the LPUART_ISR register value changes to the default value. This bit is set and cleared by software. 0: LPUART prescaler and output inhibition, low power mode; 1: LPUART enabled

				The software needs to wait for LPUART_ISR.TC to be set before it can clear the UE bit and enter the low power mode; At the same time, the DMA channel needs to be disabled before the UE bit is cleared. When this bit is cleared, the LPUART configuration register value is maintained, but the LPUART_ISR status registers are all reset.

33.4.2. LPUART control register 2 (LPUART_CR2)

Address offset: 0x04

Reset value: 0x0000_0000

To configure the ADD bit of this register, for improved configuration efficiency, all registers in LPUART_CR2 need to be configured at once. If this register needs to be configured multiple times, a 5 LPUART core clock cycle interval is required between each two configurations.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD[7:0]								Res	Res	Res	TXOE_ALWAYS_ON	MSBFIRST	DATAINV	TXINV	RXINV
RW	RW	RW	RW	RW	RW	RW	RW				RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWAP	Res	STOP[1:0]		Res	Res	Res	Res	Res	Res	Res	ADDM7	Res	Res	Res	Res
RW		RW	RW								RW				

Bit	Name	R/W	Reset Value	Function
31:24	ADD[7:0]	RW	8'b0	Address of the LPUART node This register is used in multiprocessor silent or stop mode and is used as an address when the address wakes up. The MSB bit sent by the transmitter is 1. In normal receive mode, this register is used for character detection. At this time, the received content is compared with ADD [7: 0], and if it matches, the CMF flag bit is set. This register can only be written if RE = 0 or UE = 0.
23:21	Reserved	-	-	Reserved
20	TXOE_ALWAYS_ON	RW	1	This register is used to control the LPUART output enable. 0: Hardware autonomous control output enables ON/OFF 1: Software control output enabled normally on This bit can only be written when LPUART is not enabled (UE = 0) Note: In half duplex mode, this bit needs to be turned off.
19	MSBFIRST	RW	0	The most significant bit comes first. 0: After the start bit, the 0th bit data is transmitted and received; 1: After the start bit, send and receive the 7/8/9th bit data; This bit can only be written when LPUART is not enabled (UE = 0)
18	DATAINV	RW	0	Binary data inverse processing. 0: Send and receive data register data in forward/forward logic. (1=H, 0=L) 1: Send and receive data register data with reverse/reverse logic. The parity bits are also processed inversely. (1 = L, 0 = H) This bit can only be written when LPUART is not enabled (UE = 0)
17	TXINV	RW	0	The TX pin active level is inverted. 0: The TX pin is at the standard logic level (V_{CC} = 1/idle, Gnd = 0/mark); 1: The TX pin signal value is inverted (V_{CC} = 0/mark, Gnd = 1/idle). This bit can only be written when LPUART is not enabled (UE = 0)
16	RXINV	RW	0	The RX pin active level is inverted.

				0: The RX pin is at the standard logic level ($V_{CC} = 1/\text{idle}$, $Gnd = 0/\text{mark}$); 1: The RX pin signal value is inverted ($V_{CC} = 0/\text{flag}$, $Gnd = 1/\text{idle}$). This bit can only be written when LPUART is not enabled ($UE = 0$)
15	SWAP	RW	0	TX/RX pin interchange. 0: TX/RX pins are defined according to standard pinout; 1: TX/RX pin interchange. This is used when cross-connecting other LPUARTs. This bit can only be written when LPUART is not enabled ($UE = 0$)
14	Reserved	-	-	Reserved
13:12	STOP[1:0]	RW	2'b0	Stop bit configuration. 00: 1 stop bit; 01, 11: reserved; 10: 2 stop bit; This bit can only be written when LPUART is not enabled ($UE = 0$)
11:5	Reserved	-	-	Reserved
4	ADDM7	RW	0	7-bit/4-bit address detection configuration. 0: 4-bit address detection; 1: 7 bit address detection (8 bit data mode); This bit can only be written when LPUART is not enabled ($UE = 0$)
3:0	Reserved	-	-	Reserved

33.4.3. LPUART control register 3 (LPUART_CR3)

Address offset: 0x08

Reset value: 0x0000_0000

To configure the WUS bit of this register, for improved configuration efficiency, all registers in LPUART_CR3 need to be configured at once. If this register needs to be configured multiple times, a 5 LPUART core clock cycle interval is required between each two configurations.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUFIE	WUS[1:0]		Res.	Res.	Res.	Res.
									RW	RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEP	DEM	DDRE	OVRDIS	Res.	CTSIE	CTSE	RTSE	DMAT	DMAR	Res.	Res.	HDSEL	Res.	Res.	EIE
RW	RW	RW	RW		RW	RW	RW	RW	RW			RW			RW

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	Reserved
22	WUFIE	RW	0	Low power wake-up interrupt enabled. 0: Interrupt prohibition 1: WUF interrupt enabled
21:20	WUS[1:0]	RW	0	Low power wake-up selection. 00: Address matching generates WUF 01: Reserved 10: Start bit detection generates WUF 11: RXNE generates WUF This register can only be written when the LPUART is disabled ($UE=0$).
19:16	Reserved	-	-	Reserved
15	DEP	RW	0	DE polarity selection. 0: DE high active 1: DE low active This bit can only be written when LPUART is not enabled ($UE = 0$)
14	DEM	RW	0	DE mode enable. 0: Disabled 1: DE mode is enabled, and DE signal is output at RTS pin. The user controls external transmissions by enabling this bit. This bit can only be written when LPUART is not enabled ($UE = 0$)
13	DDRE	RW	0	Whether to disable DMA when receiving an error. 0: DMA is not disabled when receiving an error. The corresponding error flag is set, but RXNE = 0. No

				DMA request is generated, so the error data will not be output. 1: Disable DMA when receiving an error. Both the RXNE and error flag bits are set. Mask the DMA request before the error flag bit is cleared. This bit can only be written when LPUART is not enabled (UE = 0)
12	OVRDIS	RW	0	Overflow prohibited. 0: Set overflow error flag (PRE) when received data is not read before new data is received 1: Overflow function is prohibited. After receiving new data, RXNE = 1 and ORE = 0, the new data will overwrite the data in the LPUART_RDR register. This bit can only be written when LPUART is not enabled (UE = 0)
11	Reserved	-	-	Reserved
10	CTSIE	RW	0	CTS interrupt enable 0: Disabled 1: CTSIF interrupt enabled
9	CTSE	RW	0	CTS enable 0: CTS hardware flow control disabled 1: CTS mode enabled Data is only transmitted when the CTS input is asserted (tied to 0). If a data is written into the data register while CTS is deasserted, the transmission is postponed until CTS is asserted. This bit can only be written when LPUART is not enabled (UE = 0)
8	RTSE	RW	0	RTS enable 0: RTS hardware flow control disabled 1: RTS interrupt enabled, data is only requested when there is space in the receive buffer. The transmission of data is expected to cease after the current character has been transmitted. The RTS output is asserted (tied to 0) when a data can be received. This bit can only be written when LPUART is not enabled (UE = 0)
7	DMAT	RW	0	DMA is enabled when transmitting. 0: Disabled 1: Enable DMA during transmission
6	DMAR	RW	0	DMA enable receiver 0: Disabled 1: DMA enabled on reception
5:4	Reserved	-	-	Reserved
3	HDSEL	RW	0	Half-duplex selection 0: Non-half-duplex mode 1: Half-duplex mode selection This bit can only be written when LPUART is not enabled (UE = 0)
2:1	Reserved	-	-	Reserved
0	EIE	RW	0	Error interrupt enable 0: Disabled 1: Frame Error FE, Overflow Error ORE, and Noise NF interrupt generation

33.4.4. LPUARTbaud rate register (LPUART_BRR)

Address offset: 0x0C

Reset value: 0x0000_0300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BBR[19:16]			
												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BBR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19:0	BBR[19:0]	RW	0x300	LPUART baud rate.

				It is forbidden to write values lower than 0x300 in the LPUART_BRR register. This register can only be written when the LPUART is disabled (UE=0).

33.4.5. LPUARTrequest register (LPUART_RQR)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MMRQ	SBKRQ	Res.
													W	W	

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2	MMRQ	W	0	Quiet mode request. This bit write 1 puts LPUART into silent mode and sets the RWU flag.
1	SBKRQ	W	0	Send an abort request. This bit write 1 sets the SBKF and sends a BREAK request.
0	Reserved	-	-	Reserved

33.4.6. LPUARTinterrupt and status register (LPUART_ISR)

Address offset: 0x1C

Reset value: 0x0080 00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	Res.	Res.	Res.	Res.	Res.	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY
									R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	CTS	CTSIF	Res.	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
					R	R		R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	Reserved
22	REACK	R	0	Receive an enable acknowledgement flag. Hardware sets/resets this register. Indicates that the hardware is ready to receive before entering low power mode. This bit is reserved if LPUART does not support wake-up from stop low power mode.
21	TEACK	R	0	Send an enable acknowledgement flag. Hardware sets/resets this register. This register may be used when writing TE = 0 in the LPUART_CR1 register to generate an idle frame request, and then writing TE = 1 to satisfy the minimum period of TE = 0.
20	WUF	R	0	Wake up flag from low power mode. The hardware writes this bit as 1 when a wake-up event is detected. The event is defined by the WUS bitfield. This bit is cleared by software by writing 1 to WUCF in the LPUART_ICR register. If WUFIE = 1 in the LPUART_CR3 register, an interrupt is generated. Note: When UESM is cleared, WUF flag is also cleared.
19	RWU	R	0	Received silent mode wake-up. The register is set when a silent mode sequence is received; If a wake-up sequence is received, the register is cleared. The specific wake-up sequence (address tag or idle frame) is controlled by the register LPUART_CR1. WAKE bit. 0: The receiver is in the working mode; 1: The receiver is in silent mode;
18	SBKF	R	0	Send abort flag. This register indicates that the disconnect character is requested to be sent. This register is set by writing 1 to the LPUART_RQR.SBKQR register by software. The

				hardware clears the register after sending the stop bit of the abort character. 0: No disconnect character is transmitted; 1: Send disconnect character;
17	CMF	R	0	Address match flag. This register is set when a character matching the ADD [7: 0] value is received. Software writing 1 to the CMCF register clears the bit.
16	BUSY	R	0	Busy flag When the RX line receives data (the start bit is correctly received), the hardware sets this register. When the reception is completed, the hardware clears the register. 0: LPUART idle; 1: Reception is in progress;
15:11	Reserved	-	-	Reserved
10	CTS	R	0	CTS flag 0: CTS line is 1; 1: CTS line is 0;
9	CTSIF	R	0	CTS interrupt flag. 0: CTS status line has not changed; 1: CTS state line value changes;
8	Reserved	-	-	Reserved
7	TXE	R	1	Transfer register null flag, can write data to LPUART_TDR. 0: Data register full; 1: Data register empty;
6	TC	R	1	Transmission complete This bit is set by hardware if the transmission of a frame containing data is complete and if TXE is set. An interrupt is generated if TCIE=1. The software writes 1 to the TCCF register to clear the bit. 0: Transmission not complete; 1: Transmission completed
5	RXNE	R	0	Read data register not empty When the shift register value is transferred to the LPUART_RDR register, the hardware sets the register. Software reads the LPUART_RDR register and clears the bit. An interrupt is generated if the RXNEIE bit =1. 0: No data received; 1: Received data ready for reading;
4	IDLE	R	0	DLE line detected An idle frame is detected and the hardware sets the register. An interrupt is generated if the IDLEIE bit =1. The software writes 1 to IDLECF to clear the bit. 0: IDLE frame not detected; 1: IDLE frame detected
3	ORE	R	0	Overflow error flag. This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RXNE=1. The software writes 1 to the ORECF register to clear the bit. When RXNEIE = 1 or EIE = 1, an interrupt is generated. Note: When this bit is set, the RDR register content will not be lost, but the shift register will be overwritten. An interrupt is generated on ORE flag if the EIE bit is set.
2	NE	R	0	START bit noise flag. This bit is set by hardware when noise is detected on a received frame. Software writing NCF = 1 will clear the bit.
1	FE	R	0	Framing error This bit is set by hardware when out-of-sync, excessive noise, or interrupt characters are detected. The software writes FECF = 1 to clear this bit.

				An interrupt is generated if the EIE bit =1.
0	PE	R	0	Parity error This bit is set by hardware when a parity error occurs in receiver mode. The software writes PECF = 1 to clear this bit. An interrupt is generated if the PEIE bit =1.

33.4.7. LPUARTInterrupt Flag Zero Register (LPUART_ICR)

Address offset: 0x20

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUCF	Res.	Res.	CMCF	Res.
											W			W	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CTSCF	Res.	Res.	TCCF	Res.	IDLECF	ORECF	NECF	FECF	PECF
						W			W		W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	Reserved
20	WUCF	W	0	Low power wakeup wakeup flag cleared. Software writing 1 will clear the WUF register.
19:18	Reserved	-	-	Reserved
17	CMCF	W	0	Address match flag cleared. Software writing 1 will clear the CMF register.
16:10	Reserved	-	-	Reserved
9	CTSCF	W	0	CTS flag cleared. The software writes 1 to clear the CTSIF register.
8:7	Reserved	-	-	Reserved
6	TCCF	W	0	The transfer completion flag is cleared. The software writes 1 to clear the TC register.
5	Reserved	-	-	Reserved
4	IDLECF	W	0	Idle flag cleared. The software writes 1 to clear the IDLE register.
3	ORECF	W	0	Overflow error flag cleared. The software writes 1 to clear the ORE register.
2	NECF	W	0	Noise flag cleared. Software writing 1 will clear the NE register.
1	FECF	W	0	Frame error flag cleared. The software writes 1 to clear the FE register.
0	PECF	W	0	The check value error flag is cleared. The software writes 1 to clear the PE register.

33.4.8. LPUARTreceive data register (LPUART_RDR)

Address offset: 0x24

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDR[8:0]							
								R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	RDR[8:0]	R	0	Receive data register.

33.4.9. LPUARTSend Data Register (LPUART_TDR)

Address offset: 0x28

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	TDR[8:0]	RW	0	Send data register.

				When parity check is enabled, bit7 or bit8 of this register (depending on data length) will be replaced by the parity bit during transmission. Note: It is recommended to turn on TE and UE first, and then write TDR data.

33.4.10. LPUARTPrescaler Register (LPUART_PRESC)

Address offset: 0x2C

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3:0	PRESCALER	RW	0	Input clock prescalation register. 0000: No division; 0001: Divide by 2; 0010: Divide by 4; 0011: Divide by 6; 0100: Divide by 8; 0101: Divide by 10; 0110: Divide by 12; 0111: Divide by 16; 1000: Divide by 32; 1001: Divide by 64; 1010: 128 division; 1011: 256 division; Others: 256 division. This register can only be written when the LPUART is disabled (UE=0).

34. Serial peripheral interface / inter-IC sound (SPI/I2S)

This project has designed and implemented two SPI/I2S modules, both of which have identical functionalities, including I2S and SPI CRC features.

34.1. Introduction to SPI/I2S

The SPI/I2S interface may be configured to support the SPI protocol or to support the I2S audio protocol. The SPI interface works in SPI mode by default, and functions can be switched from SPI mode to I2S mode through software.

The serial peripheral interface (SPI) protocol supports half-duplex, full-duplex and simplex synchronous, serial communication with external devices. This interface may be configured in master mode and provide a communication clock (SCK) for the external slave. The interface is also capable of operating in multimaster configuration. It can be used for a variety of purposes, including two-wire simplex synchronous transmission using one bi-directional data line, and reliable communication using CRC verification.

I2S is also a synchronous serial interface communication protocol. It supports four audio standards, including the Philips I2S standard, the MSB and LSB alignment standards, and the PCM standard. It is in half-duplex communication and can work in both master and slave modes. When it acts as a master, it supplies a clock signal to an external slave through an interface.

34.2. SPI/I2S Main Features

34.2.1. SPI main features

- Supports master or slave mode
- 3-wire full-duplex simultaneous transmission
- 2-wire half-duplex synchronous transmission (with bidirectional data line)
- 2-wire simplex synchronous transmission (no bidirectional data line)
- 8-bit or 16-bit transfer frame format selection
- Support multi-master mode
- 8 Master mode baud rate prescalers (Max $f_{PCLK}/2$)
- Slave mode frequency (Max $f_{PCLK}/2$)
- Both Master and Slave modes can be managed by software or hardware NSS: dynamic change of Master/Slave operating mode
- Programmable clock polarity and phase
- Programmable data order, MSB first or LSB first
- Dedicated transmit and receive flags that can trigger interrupts
- SPI bus busy status flag
- Hardware CRC feature for reliable communication
 - In transmit mode, the CRC value can be transmitted as last byte
 - In full-duplex mode, the last received byte is automatically checked for CRC.
 - CRC error flags
- Support Motorola mode and TI mode
- Can trigger primary mode failure, overload flag that causes interrupt
- Two embedded Rx and Tx FIFOs with DMA capability, depth of four, and width of 16 bits (8 bits when data frame is set to 8 bits)

34.2.2. I2S Main Features

- Half duplex communication (send or receive only)
- Master or slave operations
- 8-bit programmable linear prescaler to reach accurate audio sample frequencies (from 8 kHz to 96 kHz)
- Data format may be 16-bit, 24-bit or 32-bit
- Packet frame is fixed to 16-bit (16-bit data frame) or 32-bit (16-bit, 24-bit, 32-bit data frame) by audio channel
- Programmable clock polarity (steady state)

- Underrun flag in slave transmission mode and Overrun flag in reception mode (master and slave)
- 16-bit register for transmission and reception with one data register for both channel sides
- Support I2S protocols:
 - I2S Philips standard
 - MSB-justified standard (left-justified)
 - LSB-justified standard (right-justified)
 - PCM standard (with short and long frame synchronization on 16-bit channel frame or 16-bit data frame extended to 32-bit channel frame)
- Data direction is always MSB first
- DMA capability for transmission and reception
- Master clock may be output to drive an external audio component. Ratio is fixed at $256 \times FS$ (where FS is the audio sampling frequency)

34.3. SPI functional description

34.3.1. SPI block diagram

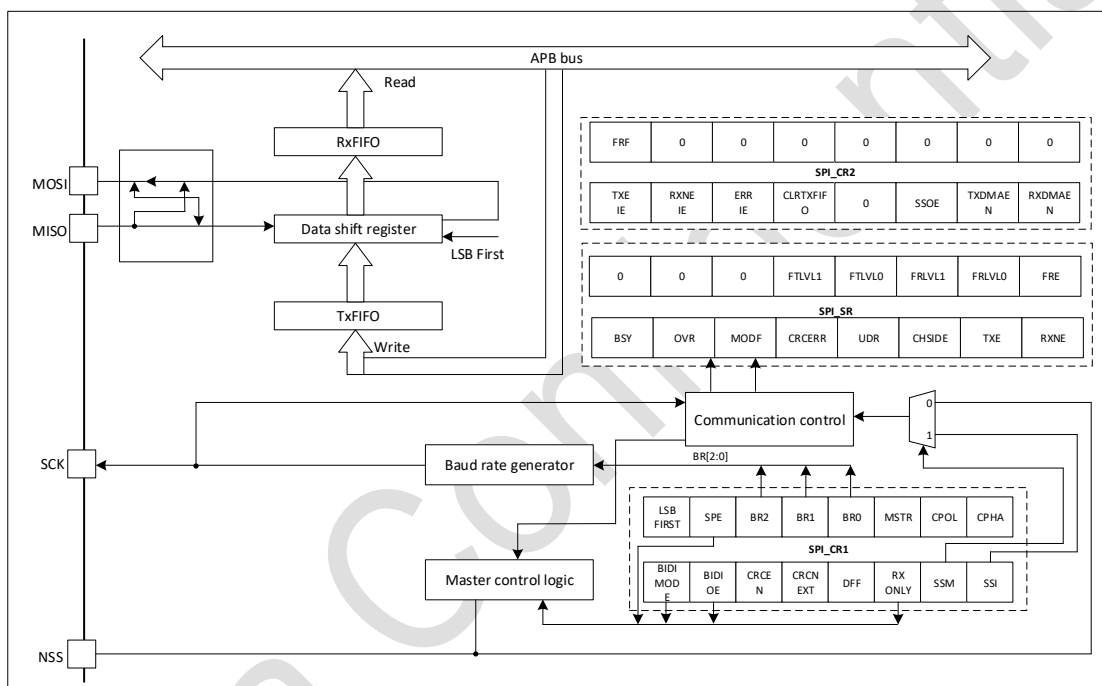


Figure 34-1 SPI block diagram

Four I/O pins are dedicated to SPI communication with external devices.

MISO: Master Input/Slave Output pin. In the general case, this pin is used to transmit data in slave mode and receive data in master mode.

MOSI: Master Output/Slave Input pin. In the general case, this pin is used to transmit data in master mode and receive data in slave mode.

SCK: Serial port clock, which serves as the output of the master and the input of the slave.

NSS: Slave Select. Depending on the SPI and NSS settings, this pin can be used to either:

- Select an individual slave device for communication
- Synchronizing data frames or
- Detect conflicts between multiple hosts

The SPI bus allows the communication between one master device and one or more slave devices. The bus consists of at least two wires: one for the clock signal and the other for synchronous data transfer. Depending on the application scenario, you can choose to add another data line and slave select signal.

34.3.2. Communication between a master and a slave

The SPI allows the MCU to communicate using different configurations, depending on the device targeted and the application requirements. These configurations use either 2-wire, 3-wire (software NSS), or 3-wire, 4-wire (hardware NSS). Communication is always initiated by the host.

34.3.2.1. Full-duplex communication

By default, SPI is configured for full duplex communication. In this configuration, the shift registers of the master and slave are connected together using two unidirectional wires between the MOSI and MISO pins. During SPI communication, data is shifted synchronously on the SCK clock edges provided by the master. The master sends data through the MOSI line and receives data from the slave from the MISO line. When the data frame transmission is completed (all bits are shifted), the information exchange between the master and the slave is completed.

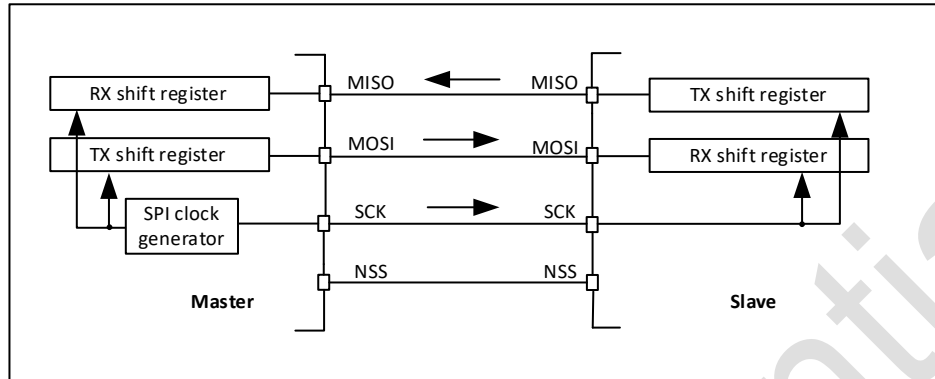


Figure 34-2 Full-Duplex Single-Master/Single-Slave Application

34.3.2.2. Half-duplex communication

By setting the BIDIMODE bit (SPI_CR1 register), SPI can operate in half-duplex mode. In this configuration, one single cross connection line is used to link the shift registers of the master and slave together. During this communication, the data is synchronously shifted between the shift registers on the SCK clock edge in the transfer direction selected reciprocally by both master and slave with the BDIOE bit in their SPI_CR1 registers. In this configuration, the MISO pin of the master and the MOSI pin of the slave are released as GPIO for other applications.

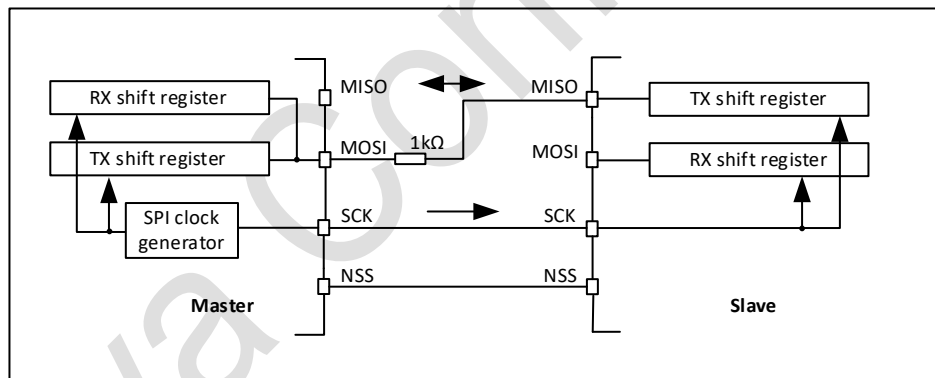


Figure 34-3 Half-duplex single master/slave application

- (1) The NSS pin can be used for hardware control flow between the master and the slave. Optionally, NSS may not be used. Then the flow has to be handled internally.
- (2) In this configuration, the MISO pin of the master and the MOSI pin of the slave can be used as GPIO.
- (3) When the communication direction between two nodes operating in bidirectional mode does not change synchronously, a critical situation will occur. The new transmitter accesses the common data line, while the previous transmitter still maintains the opposite value on the line (the value depends on SPI configuration and communication data). Both nodes then fight while providing opposite output levels on the common line temporary till next node changes its direction settings correspondingly, too. It is recommended that a serial resistor be inserted between the MISO and MOSI pins in this mode to protect the output and limit current flow between them in this case.

34.3.2.3. Simplex communications

The SPI can communicate in simplex mode by setting the SPI to either send-only mode or receive-only mode using the RXONLY (SPI_CR1 register) bit. In this configuration, only one line is used for the transfer between the shift registers of the master and slave. The remaining MISO and MOSI pins pair is not used for communication and can be used as standard GPIOs.

- Send only mode (RXONLY = 0): The configuration is the same as the full duplex mode. The application ignores information on unused input pins. This pin can be used as a standard GPIO.
- Receive-only mode (RXONLY=1): The application can disable the SPI output function by setting the RXONLY bit.
 - In the slave mode configuration, the MISO output is disabled and this pin is used as the GPIO. When the slave select signal is active, the slave continues to receive data from the MOSI pin. Received data events appear depending on the data buffer configuration.
 - In the host mode configuration, the MOSI output is disabled and the pin can be used as a GPIO. As long as the SPI is in the enabled state, the clock signal is continuously generated. The only way to stop the clock is to clear the RXONLY bit or the SPE bit until the incoming mode from the MISO pin ends, and then fill the data buffer structure based on the corresponding configuration.

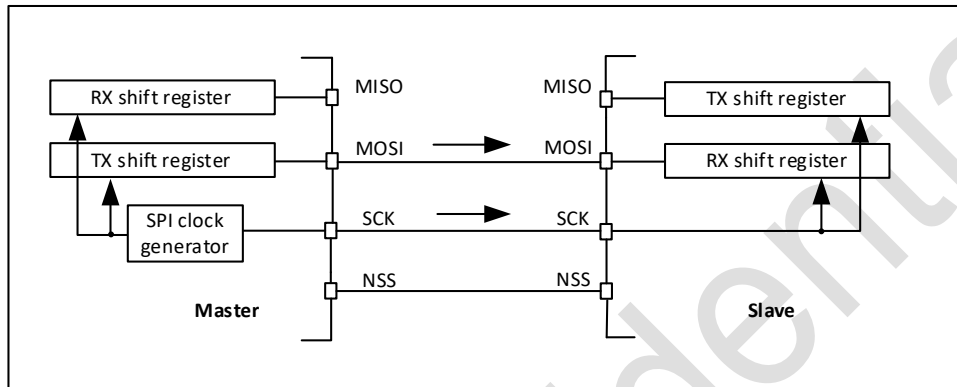


Figure 34-4 Master/slave application of single work order (master send only/slave receive only mode)

- (1) NSS pins can be used for hardware control flow between master and slave. Optionally, NSS may not be used. Then the flow has to be handled internally.
- (2) Capturing unexpected input information on the input of the Rx shift register. In the standard send-only mode, all events related to transmission and reception must be ignored (such as the OVF flag).
- (3) In this configuration, both the MISO pins can be used as GPIOs.

Note: Any simplex communication can be achieved by fixing the direction setting in half-duplex mode (enabling bidirectional mode while the BDIO bit remains unchanged).

34.3.3. Standard multi-slave communication

In a configuration with two or more independent slaves, the master uses GPIO pins for each slave to manage chip line selection. The master must select a slave by pulling down the GPIO level connected to the slave's NSS input. After this operation, communication between the standard master and the dedicated slave is established.

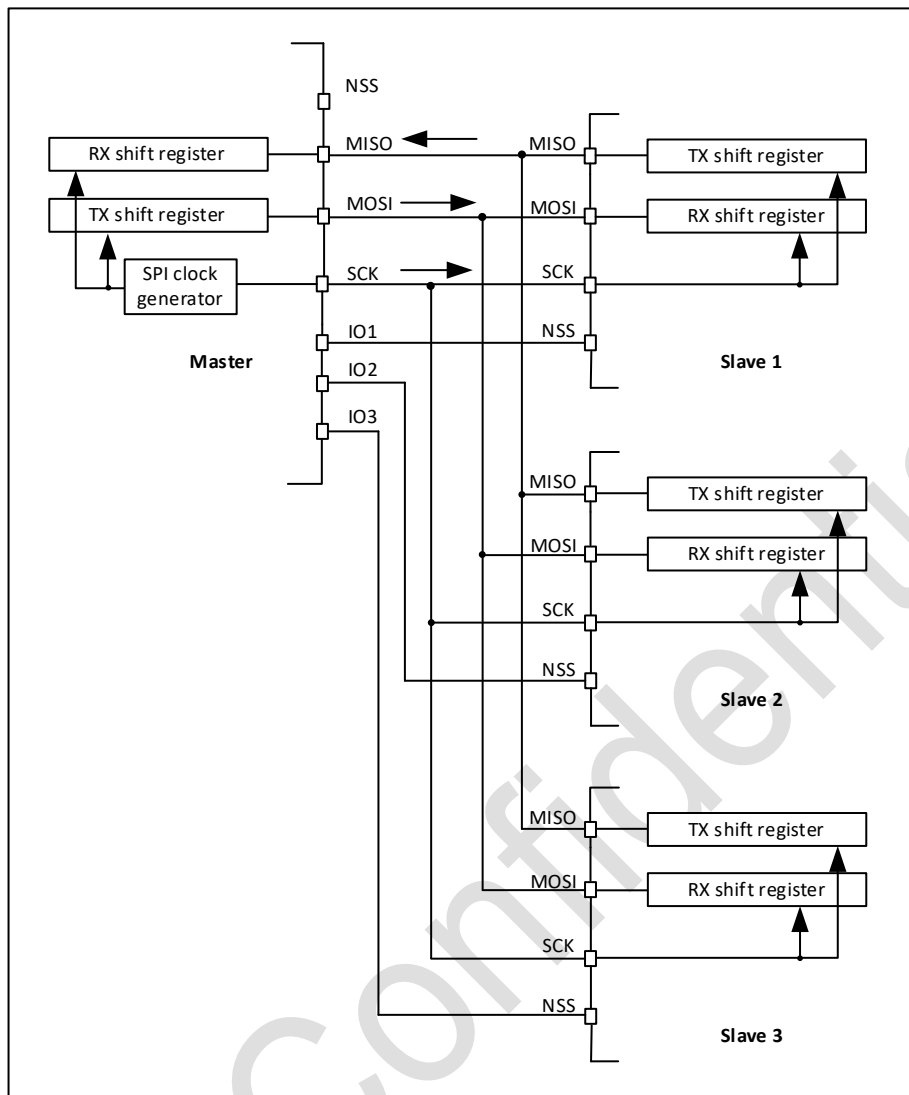


Figure 34-5 Master selects 3 slaves

The NSS pin is not used in this configuration. It has to be managed internally (SSM=1, SSI=1) to prevent any MODF error.

Since the slave MISO pins are connected together, the GPIO of all slave MISO pins must be configured for open-drain multiplexing.

34.3.4. Multi-master communication

If the SPI bus is not used for multi-host functionality, users can use built-in functionality to detect potential conflicts between two nodes trying to control the bus simultaneously. For this detection, the NSS pin is configured in hardware input mode.

The connection of more than two SPI nodes working at this mode is impossible as only one node can apply its output on a common data line at time.

When a node is invalid, it remains in slave mode by default. Once a node is about to take over control of the bus, it switches itself to master mode and then applies an active level to the other node's slave select inputs via a dedicated GPIO pin. After the session is complete, the valid slave select signal will be released, and the node controlling the bus will briefly switch back to passive slave mode, waiting for the next session to begin.

A bus conflict (mode failure MODF event) occurs if two nodes issue their respective control requests simultaneously. Then the user can apply some simple arbitration process (e.g. to postpone next attempt by predefined different time-outs applied at both nodes).

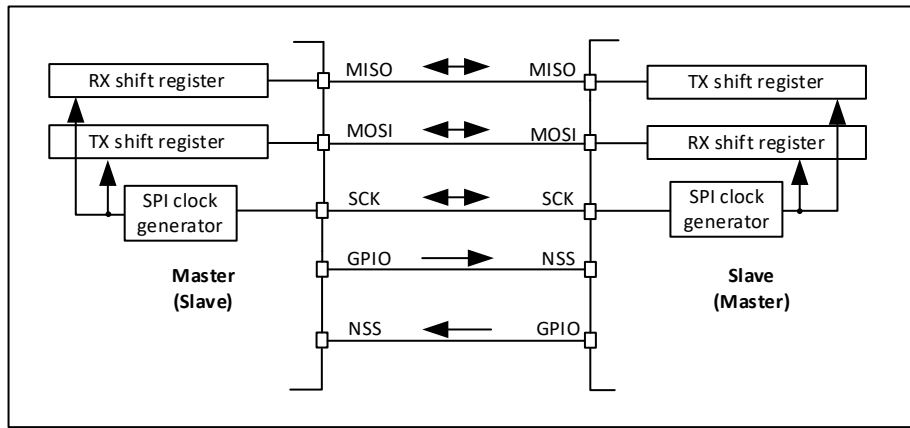


Figure 34-6 Multi-master application

The NSS pin is configured in hardware input mode at both nodes. Its active level enables the MISO line output control as the passive node is configured as a slave.

34.3.5. Slave Selection Interface (NSS)

In slave mode, the NSS works as a standard “chip select” input and lets the slave communicate with the master. In master mode, NSS can be used either as output or input. As an input it can prevent multimaster bus collision, and as an output it can drive a slave select signal of a single slave.

Hardware or software slave selection management can be set using the SSM bit of the SPI_CR1 register:

- **Software NSS Management (SSM = 1):** In this configuration, the slave selection information is internally driven by the value of the SSI bit in the SPI_CR1 register. The external NSS pin is free and available for other applications.
- **Hardware NSS management (SSM = 0):** In this case, there are several possible configurations:
 - **NSS output enabled (SSM = 0, SSOE = 1):** This configuration is used only when the device operates in master mode. The NSS pin is managed by the hardware. As long as SPI is enabled in host mode (SPE = 1), the NSS signal is driven low and remains low until SPI is turned off (SPE = 0). The SPI cannot work in multimaster configuration with this NSS setting.
 - **NSS Output Disabled (SSM = 0, SSOE = 0):** This configuration enables multi-master mode functionality if the MCU is the host on the bus. If the NSS pin is pulled low in this mode, the SPI will enter the master mode fault state and the device will automatically reconfigure in slave mode. In slave mode, the NSS pin serves as the standard chip selection input, and the slave is selected when NSS is low.

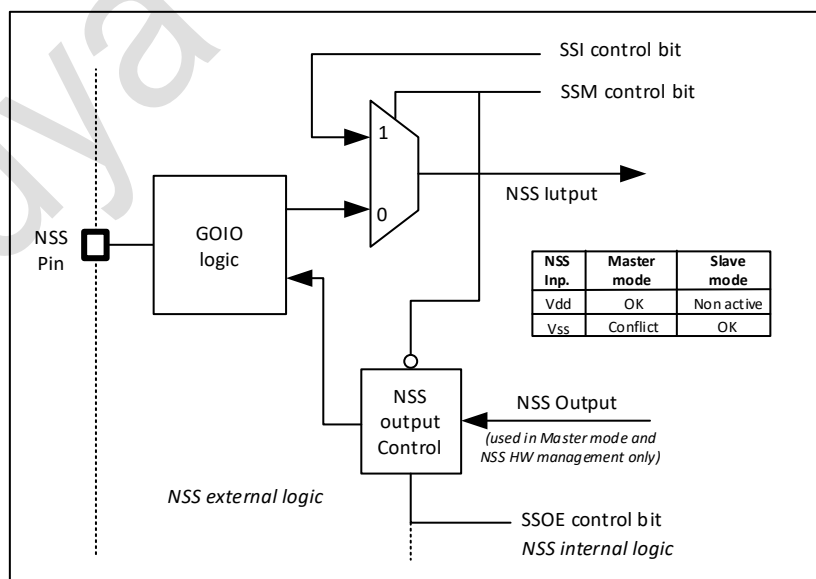


Figure 34-7 Hardware/Software Slave Mode Selection Management

34.3.6. Communication format

During SPI communication, receive and transmit operations are performed simultaneously. The serial clock (SCK) synchronizes the information shift and sampling operation on the data line. The communication format depends on the clock phase, the clock polarity and the data frame format. To be able to communicate together, the master and slave devices must follow the same communication format.

34.3.6.1. Clock phase and polarity controls

Four possible timing relationships can be selected by software through the CPOL and CPHA bits in the SPI_CR1 register. The CPOL (clock polarity) bit controls the idle state value of the clock when no data is being transferred. This bit affects both master and slave modes. If CPOL is reset, the SCK pin is low in the idle state. If CPOL is set to 1, the SCK pin is high in the idle state.

If CPHA position 1 is placed, the first data bit transferred is captured at the second edge of the SCK pin (falling edge if CPOL bit is reset, rising edge if CPOL position 1 is placed). Data are latched on each occurrence of this clock transition type. If the CPHA bit is reset, the first data bit transferred is captured at the first edge of the SCK pin (falling edge if CPOL position 1, rising edge if CPOL bit is reset). Data are latched on each occurrence of this clock transition type.

The combination of CPOL (clock polarity) and CPHA (clock phase) bits is used to select the data capture clock edge.

SPI must be banned (SPE = 0) before CPOL/CPHA changes.

The idle state of the SCK must correspond to the polarity selected by the SPI_CR1 register.

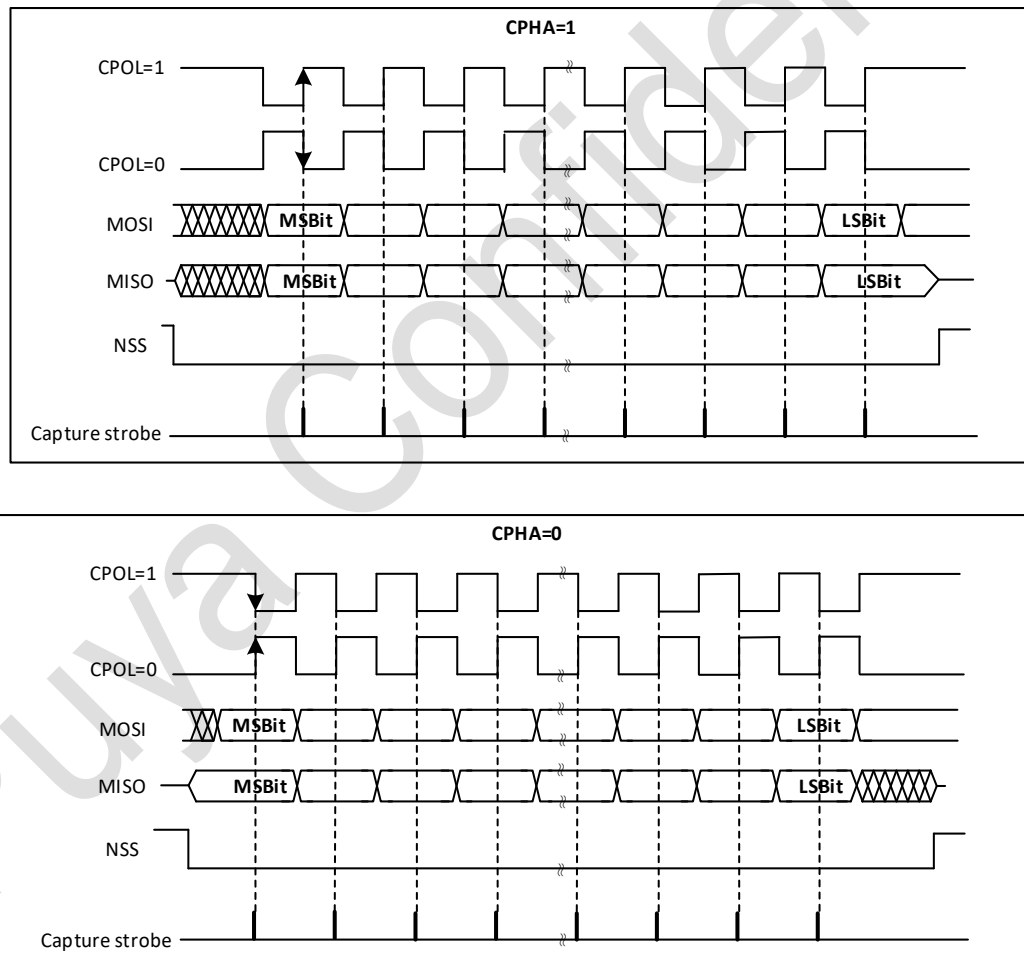


Figure 34-8 Clock/Data Timing Diagram

The order of data bits depends on LSBFIRST bit setting.

34.3.6.2. Data frame format

The SPI shift register can be set to shift out data in a MSB-first or LSB-first manner, depending on the value of the LSBFIRST bit. By using the DFF bit (SPI_CR1 register), the number of bits of the data frame is selected. It can be set 8-bit or 16-bit length and the setting applies for both transmission and reception.

34.3.7. Configuring the SPI

The configuration procedure is almost the same for master and slave. For specific pattern establishment, follow the special chapter introduction. When a standard communication is to be initialized, perform these steps:

1. Write proper GPIO registers: Configure GPIO for MOSI, MISO and SCK pins.
2. Write to the SPI_CR1 register:
 - 1) Configure the serial clock baud rate using the BR[2:0] bits (not required in slave mode)
 - 2) Configure CPOL and CPHA, define the relationship between data transfer and serial clock
 - 3) Select simplex or half-duplex mode by configuring RXONLY or BIDIMODE and BIDIOE
 - 4) Configure LSBFIRST to define the frame format
 - 5) If CRC is required, configure CRCL and CRCEN bits (when the SCK clock signal is idle)
 - 6) Configure SSM and SSI
 - 7) Configure the MSTR bit (in a multi-host NSS configuration, avoid the collision state of NSS if the host is configured to prevent MODF errors)
 - 8) Configure DFF bits, select data frame bits
3. Write to SPI_CR2 register:
 - 1) Configure SSOE (not required for slave mode)
4. Write SPI_CR3 register: configure CRC polynomial if needed
5. Write the appropriate DMA registers (including SYSCFG_CFGR3 and SYSCFG_CFGR4 configuration DMA channels): Configure the SPI TX and Rx channels of DMA

34.3.8. Enable SPI step

It is recommended that the SPI slave be enabled before the master sends the clock. If not, undesired data transmission might occur. The slave's data register must contain the data to be sent in order to start communication with the master (at the first edge of the communication clock; if the clock signal is continuous, before the end of the ongoing communication). Before enabling the SPI slave, the SCK signal must be stabilized to the idle state level corresponding to the selected polarity.

When the SPI is enabled and the TXFIFO is not empty, or the next write operation is made to the TXFIFO, the host in full duplex mode (or send only) begins communication.

In any host receive only mode (RXONLY = 1, or BIDIMODE = 1 and BIDIOE = 0), after SPI is enabled, the host starts communicating and the host immediately starts providing the clock.

For DMA processing, follow the specific sections.

34.3.9. Data transmission and reception

34.3.9.1. RXFIFO and TXFIFO

SPI All data traffic passes through a FIFO with a depth of 4 and a width of 16 bits (8 bits when the data frame is set to 8 bits). This enables the SPI to work in a continuous flow, and prevents overruns when the data frame size is short. Each direction has its own FIFO called TXFIFO and RXFIFO. These FIFOs are used in all SPI modes.

The handling of FIFOs depends on the data exchange mode (duplex, simplex) and data frame format.

A read access to the SPI_DR register returns the oldest value stored in RXFIFO that has not been read yet. A write access to the SPI_DR stores the written data in the TXFIFO at the end of a send queue. Read and write access must be aligned to the data width. The FTLVL [1: 0] and FRLVL [1: 0] bits show the current occupancy levels of the two FIFOs.

A read access to the SPI_DR register must be managed by the RXNE event. This event is triggered when the RXFIFO data is non-empty. When RXNE is cleared, RXFIFO is considered to be empty.

In a similar way, write access of a data frame to be transmitted is managed by the TXE event. This event is triggered when the occupancy level of TXFIFO is less than or equal to half of the total capacity. Otherwise, TXE is cleared and the TXFIFO is considered as full.

In this way, RXFIFO and TXFIFO can both store 4 data frames.

Both TXE and RXNE events can be handled by query or interrupt.

Another way to manage data exchange is to use DMA.

If the next data is received while the RXFIFO is full, an overflow event (OVR flag) occurs. Overflow events can be handled by polling or interrupting.

The BSY bit being set indicates ongoing transaction of a current data frame. When the clock signal is running continuously, in the master, the BSY flag remains set to 1 between data frames, but in the slave, the BSY flag goes low between data frame transmissions for the shortest period of time (one

SPI clock).

In some application scenarios, when writing data to TXFIFO, you can clear the TXFIFO data by setting the CLRTXFIFO bit, so as to write new data to TXFIFO for re-communication.

34.3.9.2. Sequence handling

A few data frames can be passed at single sequence to complete a message. After transmission is enabled, when there is any data in the TXFIFO of the host, the sequence starts and continues as long as there is data in the TXFIFO of the host device. The clock signal is continuously provided by the host until TXFIFO is empty, and then the clock signal stops and waits for other data.

In the receive-only mode, i.e. half-duplex (BIDIMODE = 1, BIDIOE = 0) or simplex mode (BIDIMODE = 0, RXONLY = 1), when SPI is enabled and the receive-only mode is activated, the host immediately starts receiving. The clock signal is provided by the master, and it does not stop until either SPI or receive-only mode is disabled by the master. The master receives data frames continuously up to this moment

While the master can provide all the transactions in continuous mode (SCK signal is continuous) it has to respect slave capability to handle data flow and its content at anytime. When necessary, the master must slow down the communication and provide either a slower clock or separate frames or data sessions with sufficient delays. It should be noted that there is no underflow error signal for the master or slave, and the data from the slave is always processed by the master, even if the slave fails to prepare the data correctly and in time. For slaves, a better approach is to use DMA, especially when the data frame is short and the bus rate is high.

In a multi-slave system, each sequence must be controlled by an NSS pulse, so that only one of the slaves is selected for communication. In a single slave system, there is no need to control the slave by NSS, but it is usually better to provide this pulse at this time to synchronize the slave at the beginning of each data sequence. NSS can be managed by both software and hardware.

When the BSY is at position 1, it indicates that a data frame transaction is being processed. When the performed frame interaction is complete, the RXNE flag is set to 1. After the last bit is sampled, the entire data frame is stored in the RXFIFO.

34.3.9.3. Close SPI step

When shutting down the SPI, you must follow the shutdown steps described in this paragraph. It is important to do this before the system enters a low-power mode when the peripheral clock is stopped. Ongoing transactions can be corrupted in this case. In some modes the disable procedure is the only way to stop continuous communication running.

In full-duplex or send-only mode, the host can complete the interaction when it stops providing the data to be sent. In this case, the clock stops after the last data transaction. In these modes, users must use the standard prohibition process before SPI can be prohibited. When the SPI is transmitted by the host, if one frame interaction is in progress at this time, or the next data frame exists in the TXFIFO, the SPI is turned off at this time, and the state of the SPI is unpredictable.

When the host is in receive-only mode, the only way to stop the continuous clock is to turn off the peripherals (SPE = 0). This must occur in specific time window within last data frame transaction just between the sampling time of its first bit and before its last bit transfer starts (in order to receive a complete number of expected data frames and to prevent any additional “dummy” data reading after the last valid data frame). Specific steps must be followed when turning off SPI in this mode.

After SPI is turned off, received but not read data is always stored in RXFIFO, and this data must be processed after the next time SPI is enabled before a new sequence can be started. To prevent unread data, you need to ensure that RXFIFO is empty when you turn off SPI. You can turn off SPI through the correct turn-off step, or you can turn off SPI (RCC_APBx register) by initializing all SPI registers by software reset by controlling specific registers dedicated to peripheral reset.

The standard prohibition process is based on the BSY status and looking at the FTLVL [1: 0] to ensure that the transfer is complete thoroughly. This check can be done in specific cases, too, when it is necessary to identify the end of ongoing transactions, for example:

- When NSS signal is managed by software and master has to provide proper end of NSS pulse for slave.
- The last data frame or CRC frame is still in transmission, and the interactive data stream from the DMA or FIFO is complete.

The correct prohibition process is (except for receive only mode):

1. Wait until FTLVL[1:0] = 00 (no more data to transmit)
2. Wait until BSY=0 (the last data frame is processed)
3. Disable the SPI (SPE=0).
4. Read data until FRLVL[1:0] = 00 (read all the received data)

For a specific receive-only mode, the correct prohibition process is:

1. During the last data frame transmission, the reception flow is interrupted by disabling SPI (SPE = 0)
2. Wait until BSY=0 (the last data frame is processed).
3. Read data until FRLVL[1:0] = 00 (read all the received data)

34.3.9.4. DMA communication

In order to operate at maximum speed and facilitate the data register read/write processes required to avoid overflows, SPI provides DMA functionality, which employs a simple request/answer protocol. When TXE or RXNE is set, a DMA request will be generated. The Tx and Rx buffers have independent requests.

- When sending, each time TXE is set to 1, a DMA request is generated. The DMA then writes data to the SPI_DR register.
- Upon reception, each time RXNE is set to 1, a DMA request is generated. The DMA then reads the data of the SPI_DR register.

When the SPI is used only to transmit data, it is possible to enable only the SPI Tx DMA channel. In this case, because the received data is not read away, the OVR flag position bit. When the SPI is used only to receive data, it is possible to enable only the SPI Rx DMA channel.

At the time of transmission, when the DMA has written all the data to be transmitted (the TCIF flag bit of the DMA_ISR register is set), you can monitor the BSY flag to ensure that the SPI communication is complete. This is used to prevent the final transmission from being corrupted when SPI is disabled or stop mode is entered. The software must first wait until FTLVL[1:0]=00 and then until BSY=0.

To prevent error events caused by DMA channel management from occurring when communication is started via DMA, the following steps must be followed:

1. Enable DMA Rx buffer (RXDMAEN bit of SPI_CR2) (if Rx DMA is used)
2. Configure the SYSCFG_CFGR3.DMAx_MAP register and select the DMA channel used by SPI; And configure the registers of DMA (refer to the description of DMA module)
3. Enable DMA Tx buffer (at TXDMAEN bit of SPI_CR2 register) (if Tx DMA is used)
4. Configure SPE = 1 to enable SPI

Force closure of communication with the following steps:

1. If DMA is enabled, turn off the DMA module by operating the DMA register (refer to the description of the DMA module)
2. Close SPI via SPI Close Process
3. Close DMA Tx and Rx buffers (if DMA Tx and Rx are used) by clearing TXDMAEN and RXDMAEN (SPI_CR2 registers)

34.3.9.5. time series

This section introduces some typical timings that are valid for queries, interrupts, or DMA. For simplicity, the LSBFIRST=0, CPOL=0 and CPHA=1 setting is used as a common assumption here. Full DMA operation configuration is also not available.

1. After activating NSS and enabling SPI, the slave starts to control MISO; The slave loses control of the MISO when the NSS is released or the SPI is turned off. Slave machines must be provided sufficient time to prepare master-specific data before the transfer begins.
On the host side, SPI peripherals control the MOSI and SCK signals (also NSS signals) only after SPI is enabled. If the SPI is turned off, the SPI peripherals are disconnected from the GPIO, and the level values on these lines depend on the GPIO setting.
2. At the master, BSY stays active between frames if the communication is continuous. On the slave side, the BSY signal typically remains low for at least one clock cycle between data frames.
3. The TXE signal is cleared only if TXFIFO is full.
4. As soon as the TXDMAEN bit is set, the DMA arbitration process begins. The TXE interrupt is generated just after the TXEIE is set. When the TXE signal is active, data transmission to the TxFIFO is started until the TxFIFO becomes full, or the DMA transmission is complete.
5. If all the data to be sent is loaded into the TxFIFO, the DMA Tx TCIF flag will be pulled up before the SPI bus transmission. This flag is always high until the SPI interaction is completed.
6. In data encapsulation mode, TxE and RxNE events occur in pairs, and the access to each read/write FIFO is 16 bits wide until the number of data frames is even. If TxFIFO is 3/4 full, the FTLVL state stops at the FIFO full level. This is why the last odd frame cannot be stored before the TxFIFO becomes 1/2 full. The data frame is stored in the TxFIFO in an 8-bit access mode (automatically realized by the DMA internal signal when LDMA_TX is set to 1).
7. In order to receive the last odd data frame of the data encapsulation mode, the Rx threshold must be changed to 8-bit when the last data frame is processed (automatically implemented by the DMA internal signal when LDMA_Rx is set to 1).

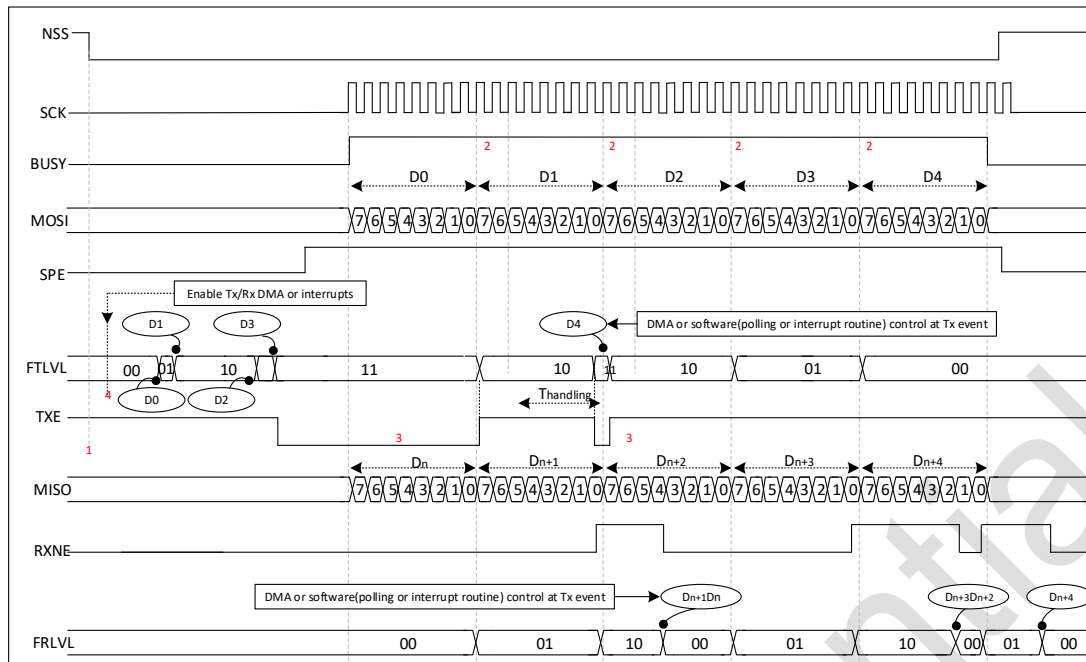


Figure 34-9 main mode full-duplex communication (frame length = 8)

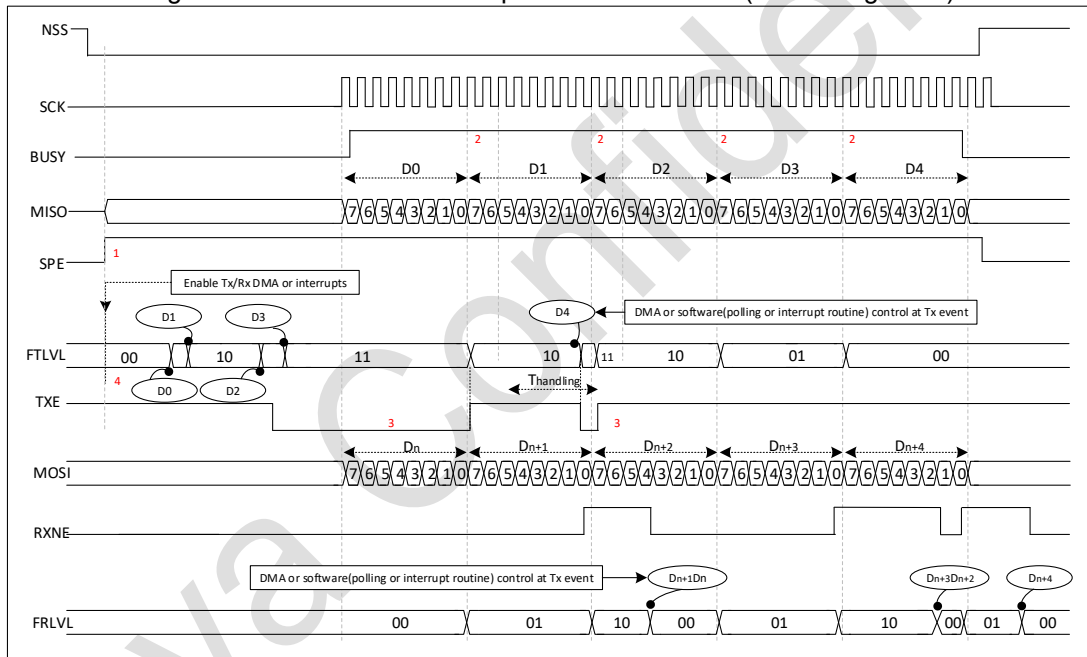


Figure 34-10 Slave mode full-duplex communication (frame length = 8)

34.3.10. Status flag

Three status flags are provided for the application to completely monitor the state of the SPIbus.

34.3.10.1. Send Cache Null Flag (TXE)

When the TXFIFO has enough space to store data to be sent, the TXE flag position bit. TXE flag is linked to the TXFIFO level. The flag bit becomes high and remains high until the TXFIFO occupancy level is less than or equal to 1/2 FIFO depth. If the TXEIE of the SPI_CR2 register is set, an interrupt request is generated. The bit is cleared automatically when the TXFIFO level becomes greater than 1/2.

34.3.10.2. Receive cache non-empty (RXNE)

When data is received, RXNE is set.

If the RXNEIE position of the SPI_CR2 register is bit, an interrupt is generated.

The RXNE is cleared by hardware automatically when the above conditions are no longer true.

34.3.10.3. Busy flag (BSY)

The BSY flag is set and cleared by the hardware (writing this bit has no effect). This flag indicates that SPI is transmitting data (SPI bus is busy).

When it is set to '1', it indicates that the SPI is busy communicating with one exception: in the bidirectional reception mode of the main mode (MSTR = 1, IDIMODE = 1 and BIDIOE = 0), the BSY flag remains low during reception (except during reception of CRC).

Before the software wants to turn off the SPI module and enter the stop low power mode (or turn off the device clock), the BSY flag can be used to detect whether the transmission has ended, which can avoid corrupting the last transmission.

The BSY flag is also useful for preventing write collisions in a multimaster system.

Except for the bidirectional reception mode of the main mode (MSTR = 1, IDIMODE = 1, and BIDIOE = 0), the BSY flag is set to '1' when the transmission starts.

The BSY flag is cleared under any one of the following conditions:

- When the SPI is correctly disabled
- Host mode, when a fault is detected (MODF = 1)
- Host mode, when data sending is complete and no new data is ready to be sent
- In Slave mode, when the BSY flag is set to '0' for at least one SPI clock cycle between each data transfer.

Note: Do not use the BSY flag to handle each data send and receive. It is better to use the TXE and RXNE flags instead.

34.3.11. Error flags

34.3.11.1. Master mode fault (MODF)

Master mode failure (MODF) only occurs when: when NSS is used as the input signal (SSOE = 0), the NSS pin of the host is pulled low in the hardware management mode of the NSS pin; Or when the SSI bit is set to '0' in NSS pin software management mode, the MODF bit is automatically set. Master mode fault affects the SPI interface in the following ways:

- The MODF position is '1'. If the ERRIE bit is set, an SPI interrupt will be generated.
- The SPE bit is cleared. This blocks all output from the device and disables the SPI interface.
- The MSTR bit is cleared, thus forcing the device into slave mode.

Use the following software sequence to clear the MODF bit:

1. Make a read or write access to the SPI_SR register while the MODF bit is set.
2. Then write to the SPI_CR1 register.

In a system with multiple MCUs, in order to avoid conflicts between multiple slaves, the NSS pin of the host must be pulled up first, and then the MODF bit must be cleared. The SPE and MSTR bits can be restored to their original state after this clearing sequence.

As a security, hardware does not allow the SPE and MSTR bits to be set while the MODF bit is set.

Normally, the MODF bit of the slave cannot be set to '1'. However, in a multi-master configuration, a device may be in slave mode with the MODF bit set; At this point, the MODF bit indicates that a multi-master collision may have occurred. The interrupt program may perform a reset or return to the default state to recover from the error state.

34.3.11.2. Overflow (OVF)

An overflow condition occurs when the master or slave receives data, but the RXFIFO does not have enough space to store the received data. This can happen if the software or DMA does not have enough time to read away the previously received data (stored in the RXFIFO).

When an overrun condition occurs, the newly received value does not overwrite the previous one in the RXFIFO. The new data received is ignored, and all data sent next is discarded.

Clearing the OVR bit is done by a read access to the SPI_DR register followed by a read access to the SPI_SR register.

34.3.11.3. CRC error (CRCERR)

This flag is used to verify the validity of the received data when the CRCEN position in the SPI_CR1 register is 1. If the value received in the shift register does not match the value of SPI_RXCRC, the CRCERR flag in the SPI_SR register will be set to 1. The flag is cleared by the software.

34.3.11.4. TI mode frame format error (FRE)

A TI mode frame format error is detected when an NSS pulse occurs during an ongoing communication when the SPI is operating in slave mode and configured to conform to the TI mode protocol.

When this error occurs, the FRE flag in the SPI_SR register is set to 1. The SPI is not disabled when an error occurs, the NSS pulse is ignored, and the SPI waits for the next NSS pulse before starting a new transfer. The data may be corrupted since the error detection may result in the loss of two data bytes.

When the SPI_SR register is read, the FRE flag is cleared. If ERRIE position 1, an interrupt will be

generated when an NSS error is detected. In this case, the SPI should be disabled because data consistency is no longer guaranteed and communications should be reinitiated by the master when the slave SPI is enabled again.

34.3.12. TI mode

34.3.12.1. TI protocol in master mode

The SPI interface is compatible with TI protocol. The SPI can be configured using the FRF bit of the SPI_CR2 register to be compatible with this protocol.

Both the clock polarity and phase are forced to follow the TI protocol regardless of the settings in SPI_CR1. NSS management is also specific to the TI protocol, in which case NSS management cannot be configured through the SPI_CR1 and SPI_CR2 registers (SSM, SSI, and SSOE).

In slave mode, the SPI baud rate prescaler is used to control the moment when the MISO pin is switched to the high impedance state when the current transmission is complete. Any baud rate can be used, making it possible to determine this moment with optimal flexibility. However, the baud rate is generally set to the external master clock baud rate. The release at which the MISO signal becomes high impedance depends on the internal resynchronization and the baud rate value set by the BR [2: 0] bit of the SPI_CR1 register. It is given by the formula:

$$\frac{t_{\text{baud rate}}}{2} + 4 \times t_{\text{pclk}} < t_{\text{release}} < \frac{t_{\text{baud rate}}}{2} + 6 \times t_{\text{pclk}}$$

If the slave detects a misplaced NSS pulse during data frame transmission, the FRE flag will be set to 1.

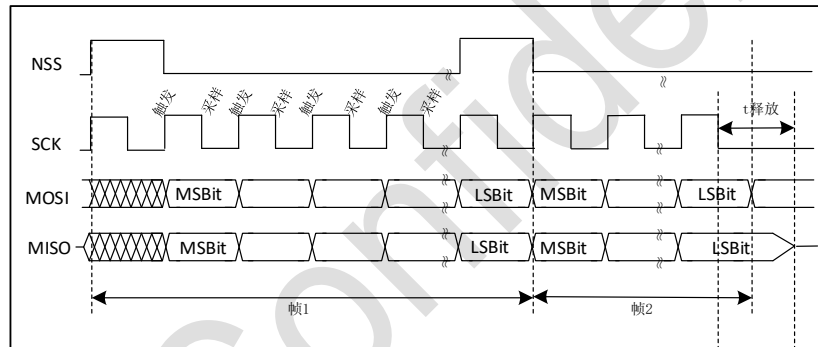


Figure 34-11 TI mode transmission

34.3.13. SPI CRC

To check the reliability of the transmitted and received data, two independent CRC calculators are utilized. SPI provides CRC8 or CRC16 calculation, and CRC8 is selected when the data frame length is 8, while CRC16 is selected when the data frame length is 16.

34.3.13.1. CRC principle

Before SPI is enabled (SPE = 1), CRC calculation is enabled by setting the CRCEN bit in the SPI_CR1 register. CRC values are calculated using odd programmable polynomials on each bit. This calculation is processed on the sample clock edge defined by the CPHA and CPOL bits in the SPI_CR1 register. The calculated CRC value is automatically checked at the end of the data block and its transmission is managed by the CPU or DMA. When it is detected that the CRC calculated internally from the received data does not match the CRC transmitted by the transmitter, a CRCERR flag is set to indicate a data error. The correct procedure to handle the CRC calculation depends on the SPI configuration and the transfer management mode selected.

34.3.13.2. CPU manages CRC transmission

After the communication starts, it will continue until the last data frame in the SPIx_DR register must be sent or received. After that, the CRCNEXT bit in the SPIx_CR1 register must be set to 1 to indicate that the CRC frame transmission will be processed after the currently processed data frame transmission. The CRCNEXT bit must be preceded by 1 at the end of the last data frame transmission. The CRC calculation will freeze during the CRC transmission.

The received CRC is stored in the RXFIFO in the form of data bytes or data words. Therefore, only in CRC mode, the receive buffer must be treated as a 16-bit buffer and only receive one data frame at a time.

The CRC format transmission typically requires an extra data frame at the end of the data transmission.

After the last CRC data is received, an automatic check is performed to compare the received value

with the value in the SPI_RXCRC register. The software must verify the CRCERR flag in the SPI_SR register to determine if the data transfer is corrupted. The software clears the CRCERR flag by writing a "0" to it.

After receiving the CRC, the CRC value is stored in the RXFIFO and must be read in the SPI_DR register to clear the RXNE flag.

34.3.13.3. DMA manages CRC communication

When SPI is enabled and DMA mode with CRC is turned on for communication, CRC is automatically transmitted and received at the end of communication (except for reading CRC data in receive-only mode), and the CRCNEXT bit does not have to be processed by software. The SPI transmit DMA channel counter must be set to the number of data frames to be transmitted, which does not include CRC frames. On the receiver side, the received CRC values are automatically processed by DMA at the end of the transmission, but the user must take care to refresh the CRC information received in the RXFIFO (since this information is always loaded into it). In full duplex mode, the receive DMA channel counter may be set to the number of data frames to be received, including CRC.

In receive-only mode, the DMA receive channel counter should contain only the amount of data that has been transferred and not the CRC calculation. After that, all CRC values must be read back from the FIFO by software based on the full transfer achieved by DMA, since the FIFO is used as a single buffer in this mode.

If a failure occurs during the transmission, the CRCERR flag position 1 in the SPI_SR register is set at the end of the data and CRC transmission.

34.3.13.4. CRC Use Process

The process of enabling CRC is as follows:

- Configure CPOL, CPHA, LSBFIRST, BR, SSM, SSI, and MSTR registers
- Enter a polynomial in the SPI_CRCPR register
- The CRC calculation is enabled by setting the CRCEN bit of the SPI_CR1 register, which also clears the registers SPI_RXCRCR and SPI_TXCRC
- Set the SPE bit of the SPI_CR1 register to start the SPI function
- Start communication and maintain communication until only the last byte or half word remains
- When the last byte or half-word is written into the transmission buffer, the CRCNEXT bit of SPI_CR1 is set to instruct the hardware to transmit the value of CRC after the transmission of the last data is completed. Stop CRC calculation during transmission of CRC values
- When the last byte or half-word is sent, the SPI sends the CRC value and the CRCNEXT bit is cleared. Similarly, the received CRC is compared with the SPI_RXCRCR value, if the comparison does not match, the CRCERR flag bit of the SPI_SR register is set, and when ERRIE of the SPI_CR2 register is set, an interrupt is generated.

Note: After CRC is enabled, CRC calculation will always be performed when receiving or sending data. Therefore, if the data no longer requires CRC calculation, the CRC should be turned off in time.

If SPI is turned off and SPI and CRC are enabled during communication, the following sequence must be followed:

- Disable the SPI
- Clear CRCEN
- CECEN enabled
- Enable the SPI

34.3.14. SPI interrupts

Table 34-1 SPI interrupt request

Interrupt event	Event flag	Enable control bit
Transmit TXFIFO ready to be loaded	TXE	TXEIE
Data received in RXFIFO	RXNE	RXNEIE
Master Mode fault event	MODF	ERRIE
Overrun error	OVR	
TI frame format error	FRE	
CRC check error	CRCERR	

34.4.2. Supports audio protocols

The three-wire bus supports time division multiplexing of audio data on two channels: the left channel and the right channel, but only one 16-bit register is used for transmission or reception. Therefore, when the software writes data to the data register, it must write according to the channel in the current transmission; Similarly, when reading register data, it is necessary to determine which channel the received data belongs to by checking the CHSIDE bit of the register SPI_SR. The left channel always sends data before the right channel (the CHSIDE bit is meaningless under the PCM protocol).

Four data and packet frames are available. Data may be sent with a format of:

- 16-bit data packed in a 16-bit frame
- 16-bit data packed in a 32-bit frame
- 24-bit data packed in a 32-bit frame
- 32-bit data packed in a 32-bit frame

When extending to a 32-bit frame using 16-bit data, the first 16 bits (MSB) are meaningful data and the last 16 bits (LSB) are forced to 0, which requires no software intervention and no DMA request (only one read/write operation is required).

The 24-bit and 32-bit data frames need two CPU read or write operations to/from the SPI_DR register or two DMA operations if the DMA is preferred for the application. For 24-bit data frame specifically, the 8 non-significant bits are extended to 32 bits with 0-bits (by hardware).

For all data formats and communication standards, the highest bit (MSB) is always sent first.

The I2S interface supports four audio standards, which can be selected by setting the I2SSTD [1: 0] bit and the PCMSYNC bit of the register SPI_I2SCFGR.

34.4.2.1. I2S Philips standard

For this standard, the WS signal is used to indicate which channel is being transmitted. This pin is valid for 1 clock cycle before sending the first bit of data (MSB).

The 16/32-bit full-precision timing diagram is shown in the figure below. The sender changes data on the falling edge of the clock signal (CK) and the receiver reads data on the rising edge. The WS signal is also latched on the falling edge of CK.

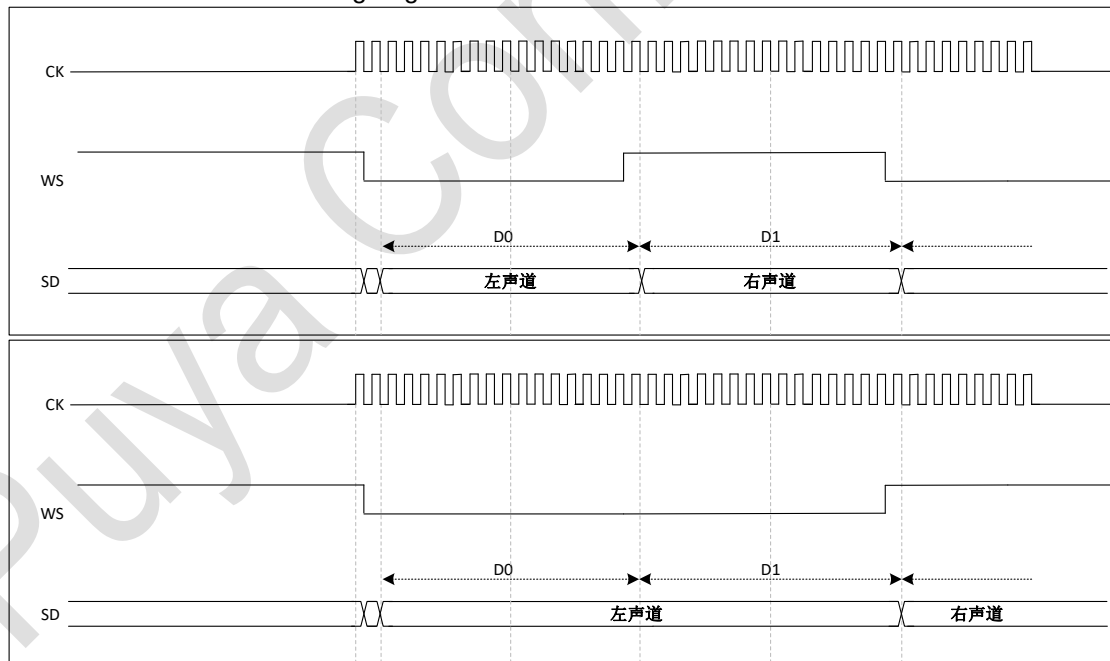


Figure 34-13 I2S Philips protocol waveform (16/32-bit full accuracy, CKPOL = 0)

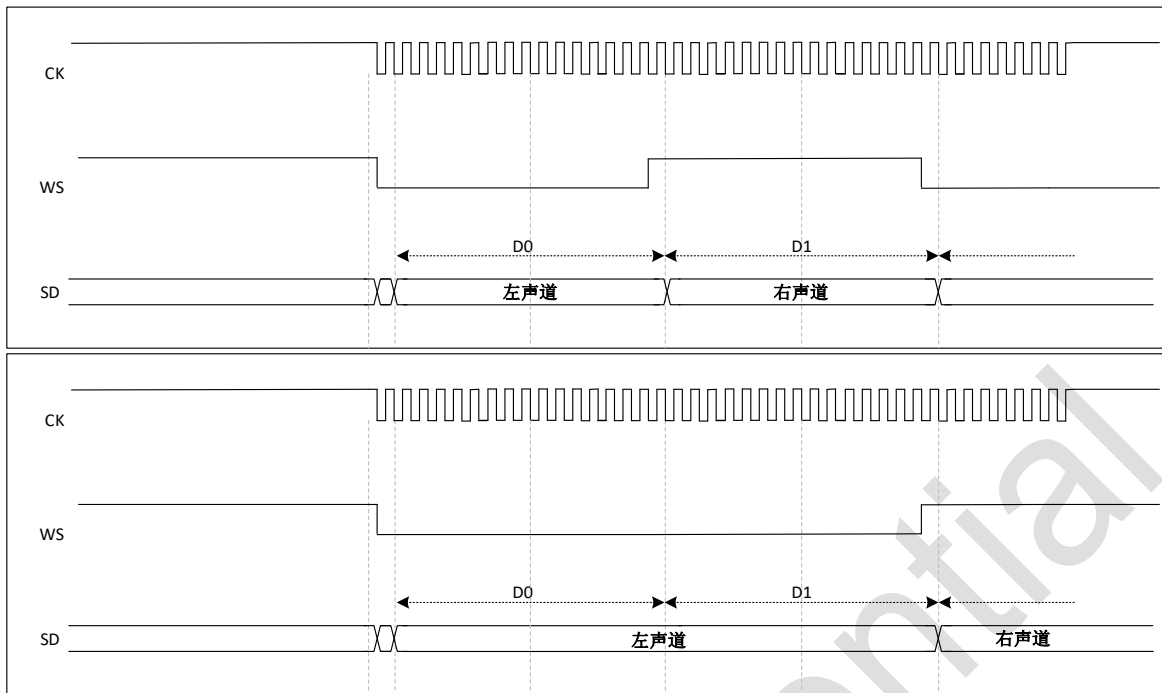


Figure 34-14 I2S Philips protocol waveform (16/32 bit full accuracy, CKPOL = 1)

The 24-bit frame timing diagram is as follows:

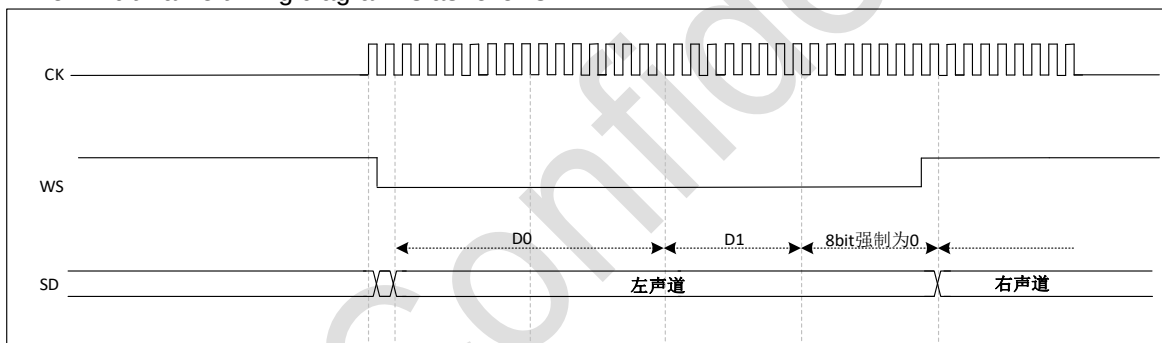


Figure 34-15 I2S Philips protocol waveform (24-bit frame, CKPOL = 0)

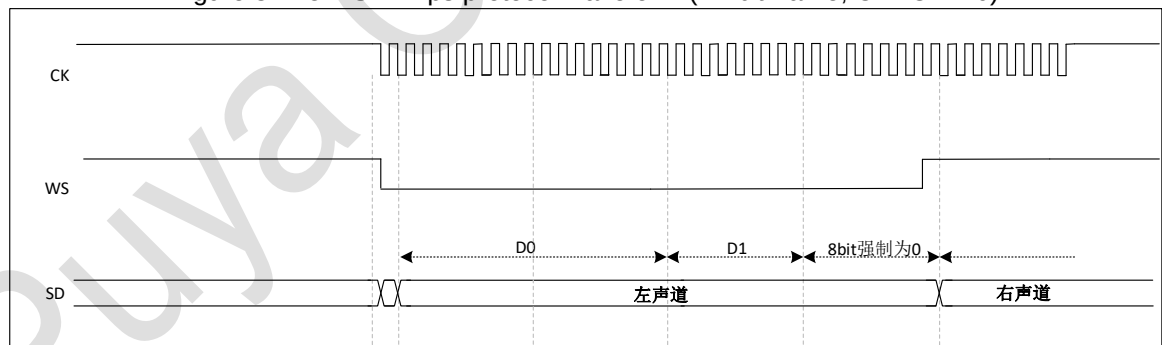


Figure 34-16 I2S Philips protocol waveform (24-bit frame, CKPOL = 1)

This mode needs two write or read operations to/from the SPI_DR register.

- In send mode: If you need to send 0x8EAA33 (24 bits), first write 0x8EAA, second write 0x33xx, lower 8 bits are meaningless
- In receive mode: if you receive 0x8EAA33 (24 bits), receive 0x8EAA for the first time, read 0x3300 for the second time, only the upper 8 bits are meaningful data, and the lower 8 bits are always 0

The following figure shows the timing of 16-bit expansion to 32-bit channel frames:

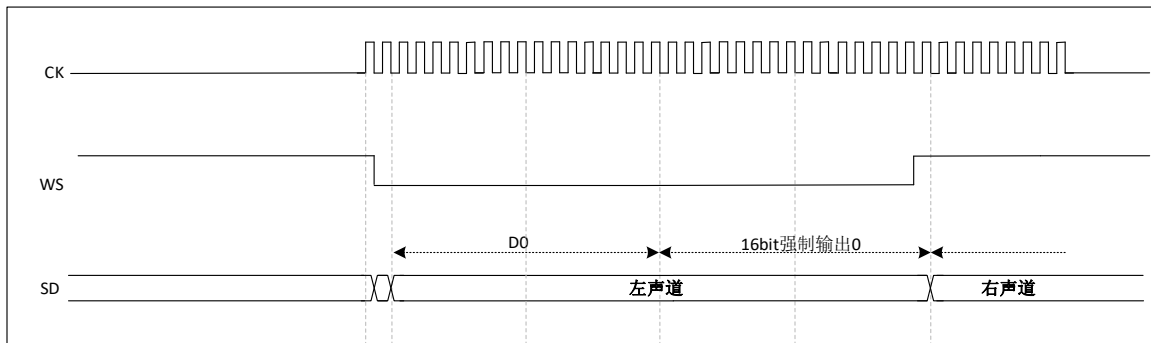


Figure 34-17 I2S Philips protocol waveform (16-bit extension to 32-bit)

When 16-bit data frame extended to 32-bit channel frame is selected during the I2S configuration phase, only one access to the SPI_DR register is required. The 16 remaining bits are forced by hardware to 0x0000 to extend the data to 32-bit format.

If the data to be transmitted or received is 0x76A3 (extended to 32 bits is 0x76A30000), you only need to operate SPI_DR once and write data 0x76A3.

When transmitting, the MSB needs to be written to the register SPI_DR: the flag bit TXE of '1' indicates that new data can be written, and an interrupt can be generated if the corresponding interrupt is allowed. This takes place even if 0x0000 have not yet been sent because it is done by hardware.

On receipt, the flag bit RXNE is set to '1' after receiving a 16-bit half-word higher (MSB), and if the corresponding interrupt is allowed, an interrupt can be generated.

In this way, more time is provided between two write or read operations, which prevents underrun or overrun conditions (depending on the direction of the data transfer).

34.4.2.2. MSB justified standard

Under this standard, the WS signal and the first data bit, which is the most significant bit (MSB), are generated simultaneously.

The 16/32-bit full-precision timing diagram is shown in the figure below. The sender changes data on the falling edge of the clock signal; The receiver is reading the data on the rising edge.

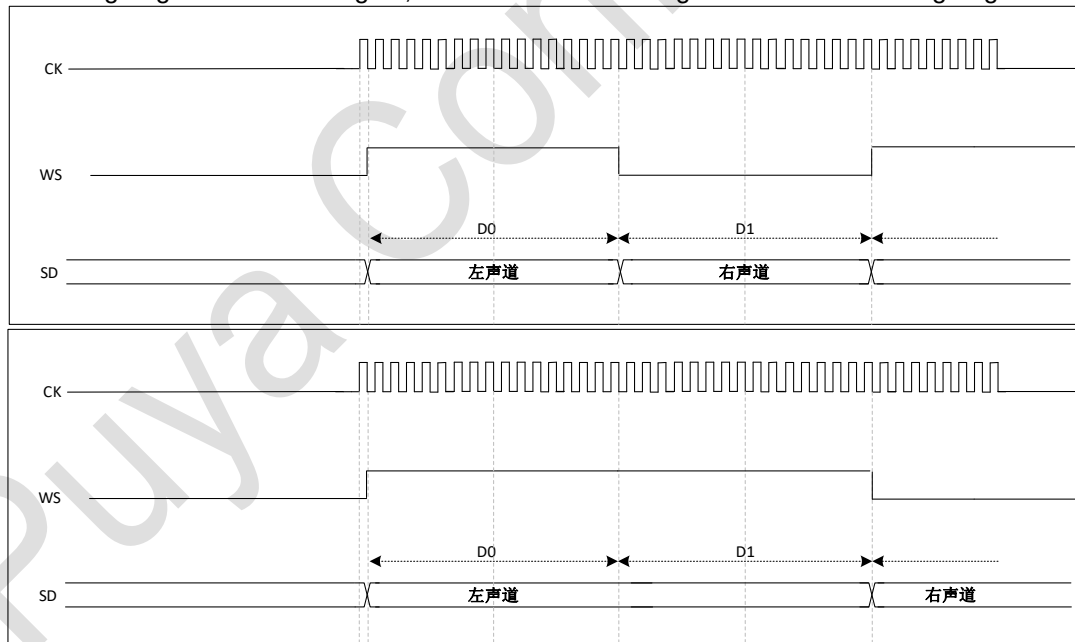


Figure 34-18 I2S MSB Alignment Standard (16/32 full accuracy, CKPOL = 0)

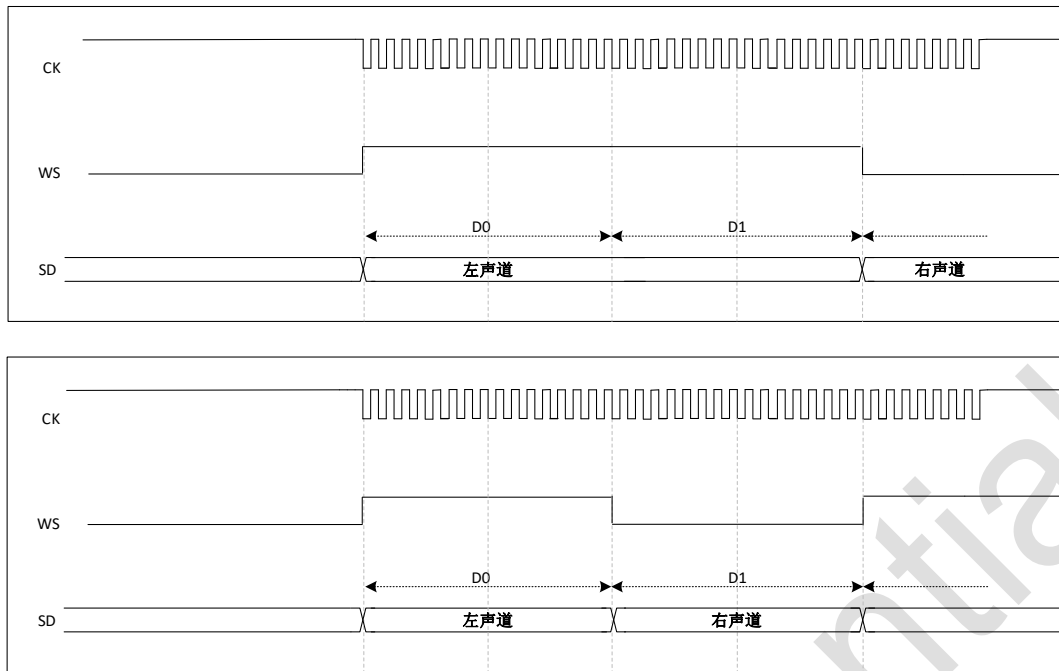


Figure 34-19 I2S MSB Alignment Standard (16/32 full accuracy, CKPOL = 1)
The 24-bit frame timing diagram is as follows:

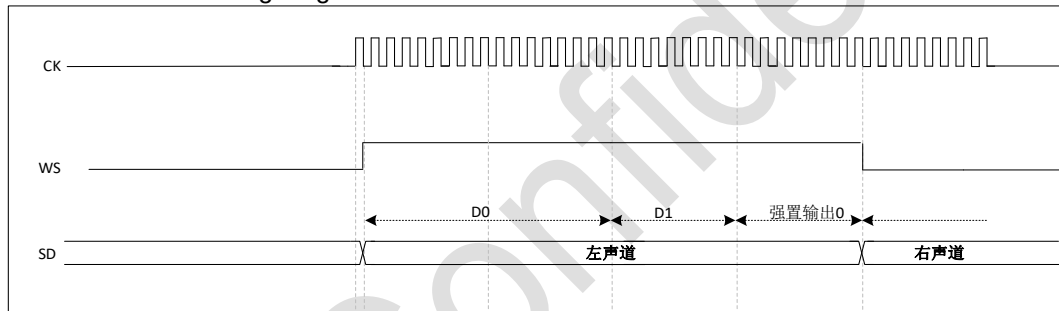


Figure 34-20 I2S MSB Alignment Standard (24-bit frame, CKPOL = 0)

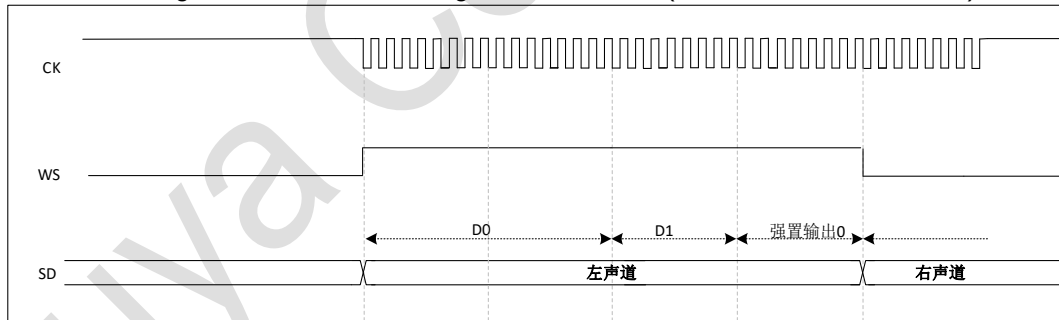


Figure 34-21 I2S MSB Alignment Standard (24-bit frame, CKPOL = 1)
The following figure shows the timing of 16-bit expansion to 32-bit channel frames:

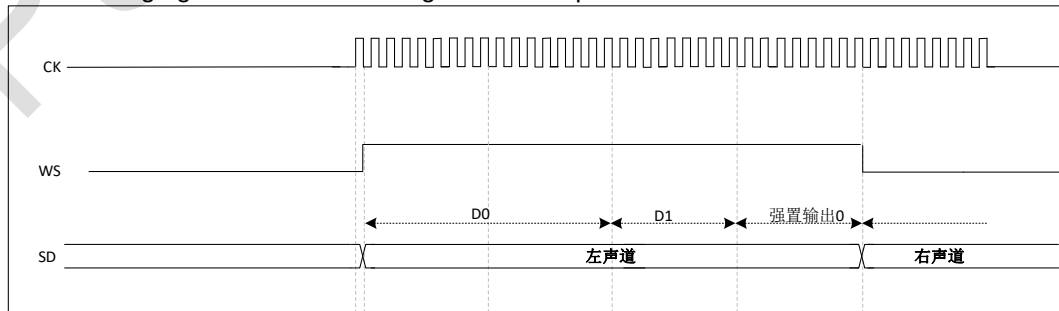


Figure 34-22 I2S MSB Alignment Standard (16-bit extension to 32-bit, CKPOL = 0)

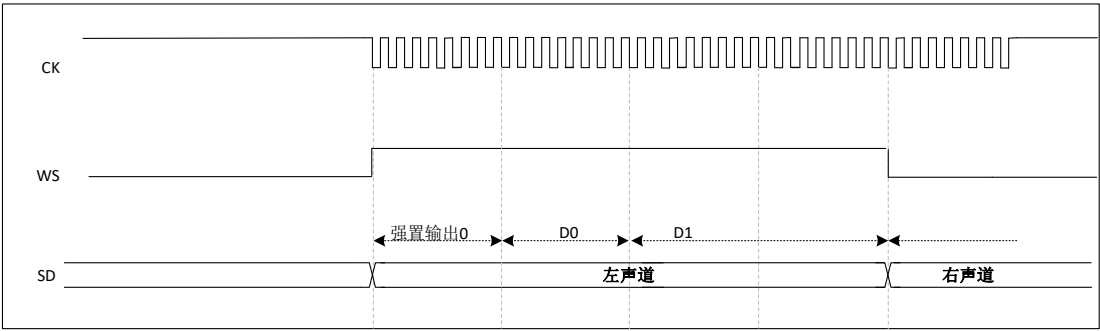


Figure 34-23 I2S MSB Alignment Standard (16-bit extension to 32-bit, CKPOL = 1)

The next TXE event occurs as soon as valid data starts being sent out from the SD pin. Upon reception, an RXNE event occurs once valid data is received (instead of the 0x0000 portion).

34.4.2.3. **LSB justified standard**

This standard is similar to the MSB alignment standard (no difference in 16-bit or 32-bit full-precision frame formats).

The 16/32-bit full-precision timing diagram is shown in the figure below. The sender changes data on the falling edge of the clock signal; The receiver is reading the data on the rising edge.

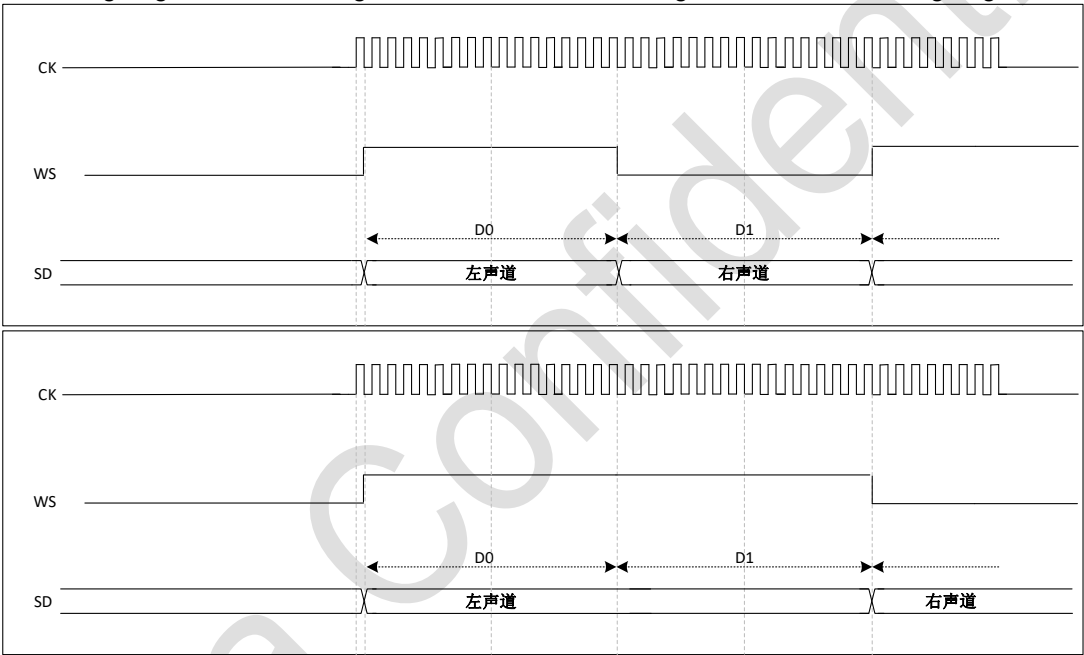


Figure 34-24 I2S LSB Alignment Standard (16/32-bit full accuracy, CKPOL = 0)

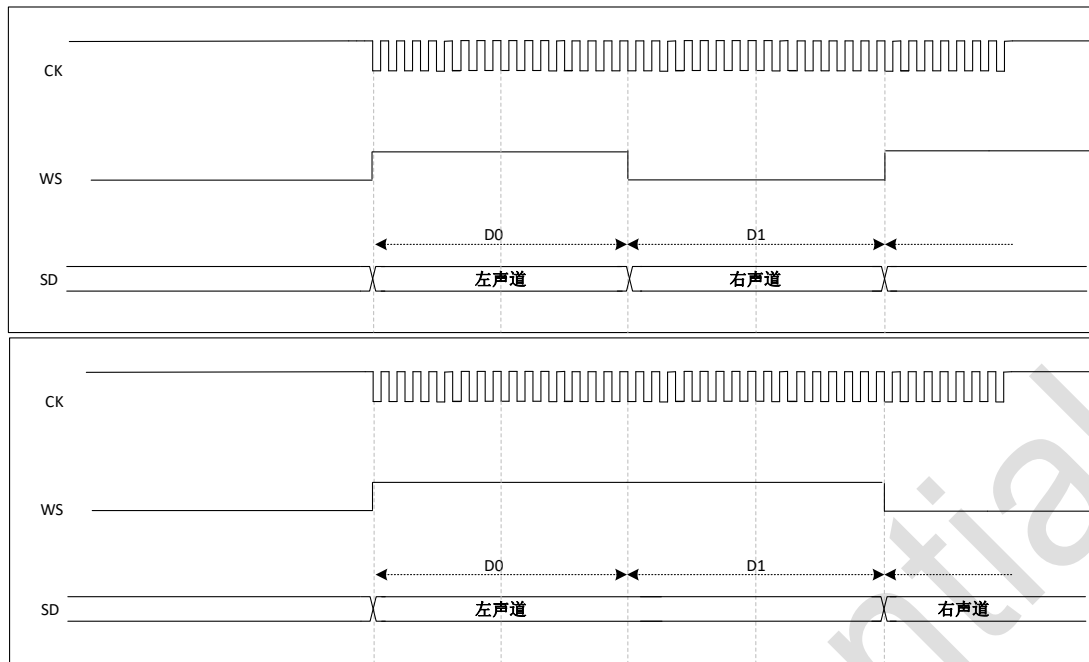


Figure 34-25 I2S LSB Alignment Standard (16/32-bit full accuracy, CKPOL = 1)

The 24-bit frame timing diagram is as follows:

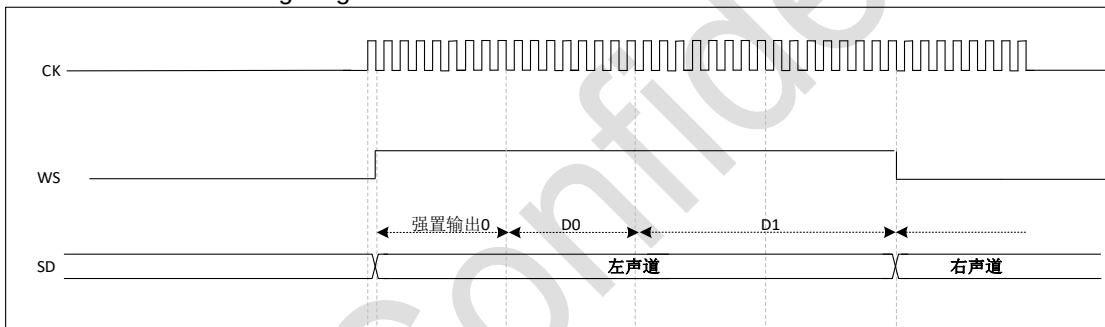


Figure 34-26 I2S LSB Alignment Standard (24-bit frame, CKPOL = 0)

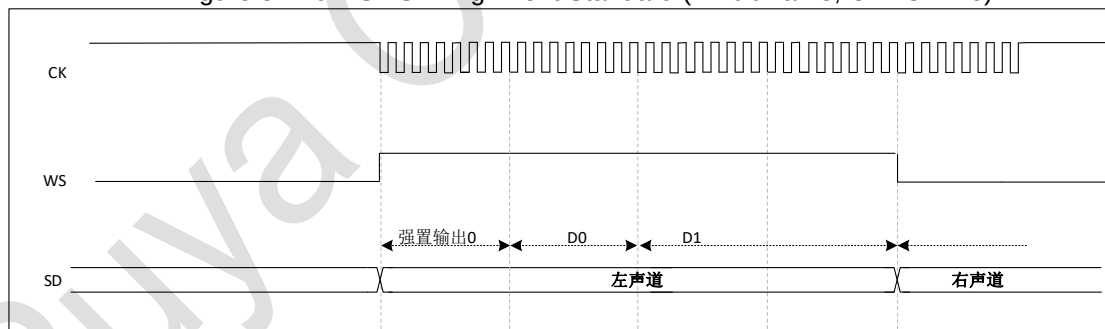


Figure 34-27 I2S LSB Alignment Standard (24-bit frame, CKPOL = 1)

■ In transmission mode:

If data 0x3478AE have to be transmitted, two write operations to the SPI_DR register are required by software or by DMA. Data register 0xXX34 is written for the first time and data register 0x78AE is written for the second time.

■ In reception mode:

If you want to receive data 0x3478AE, you need to read the register SPI_DR once when two consecutive RXNE events occur. The first time I read 0x0034, only the lower 8 bits make sense; Second readout 0x78AE.

The following figure shows the timing of 16-bit expansion to 32-bit channel frames:

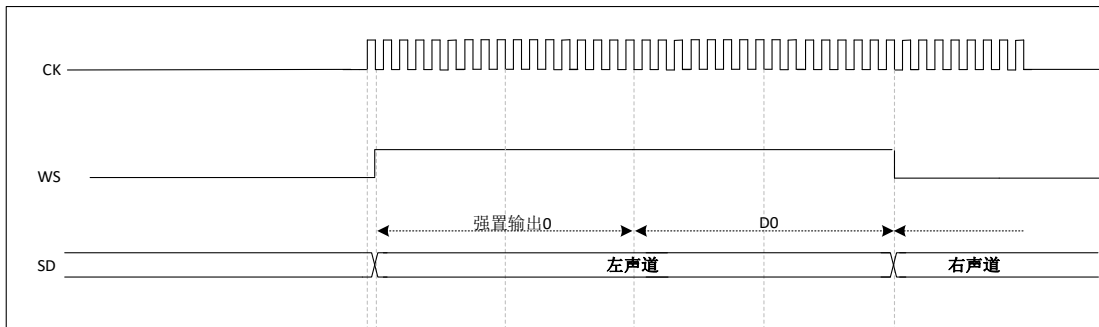


Figure 34-28 I2S LSB Alignment Standard (16-bit extension to 32-bit, CKPOL = 0)

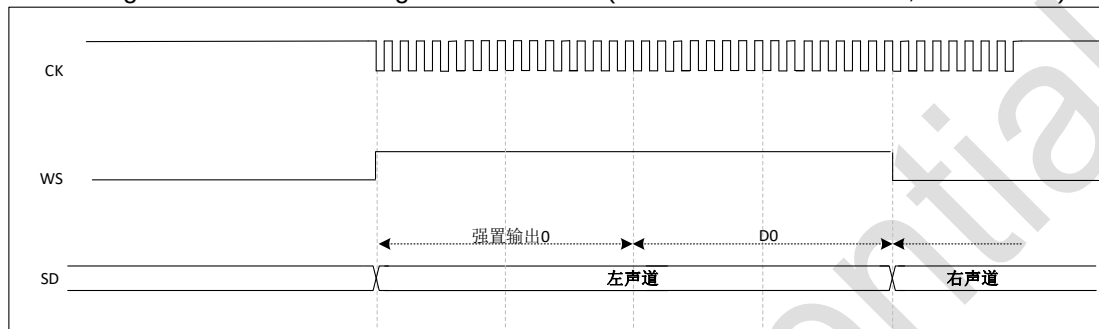


Figure 34-29 I2S LSB Alignment Standard (16-bit extension to 32-bit, CKPOL = 1)

When 16-bit data frame extended to 32-bit channel frame is selected during the I2S configuration phase, only one access to the SPI_DR register is required. At this time, the upper half word (16-bit MSB) extended to 32 bits is set to 0x0000 by hardware.

If the data to be transmitted or received is 0x76A3 (extended to 32 bits is 0x0000 76A3), you only need to operate the SPI_DR register once and write 0x76A3.

When transmitting, if TXE is '1', the user needs to write the data to be transmitted (i.e., 0x76A3). 0x0000, which is used to expand to 32 bits, is partially sent out by the hardware first, and once valid data starts to be sent out from the SD pin, the next TXE event occurs.

Upon reception, an RXNE event occurs once valid data (instead of the 0x0000 portion) is received.

In this way, more time is provided between two write or read operations to prevent underrun or overrun conditions.

34.4.2.4. PCM standard

For the PCM standard, there is no need to use channel-side information. The two PCM modes (short and long frame) are available and configurable using the PCMSYNC bit in SPI_I2SCFGR register.

In PCM mode, the output signals (WS and SD) are sampled at the rising edge of the CK signal. The input signals (WS and SD) are captured at the falling edge of CK.

Note that CK and WS are configured as outputs in main mode.

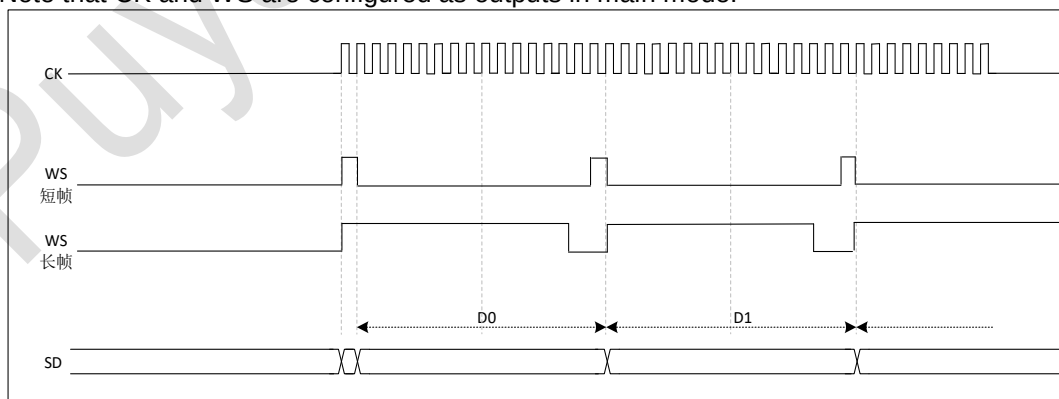


Figure 34-30 I2S PCM standard (16-bit frame, CKPOL = 0)

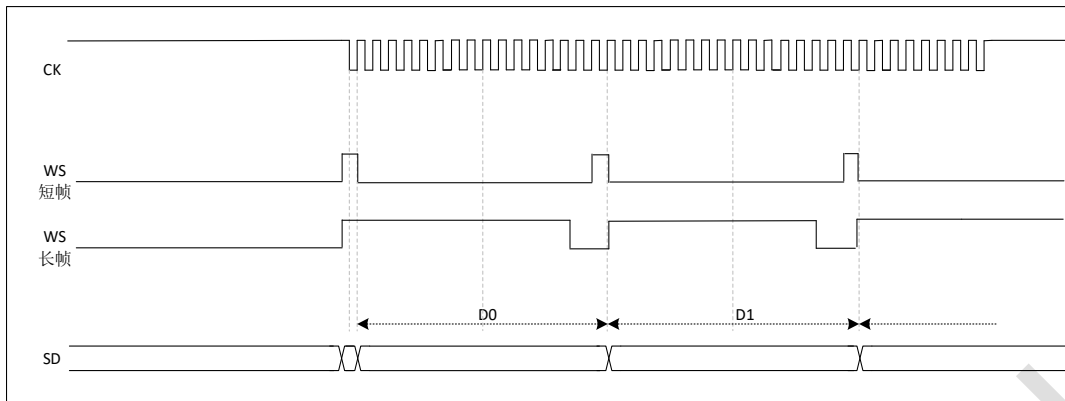


Figure 34-31 I2S PCM standard (16-bit frame, CKPOL = 1)

For long frames, in main mode, the valid time of the WS signal used for synchronization is fixed to 13 bits.

For short frames, the length of the WS signal used for synchronization is only 1 bit.

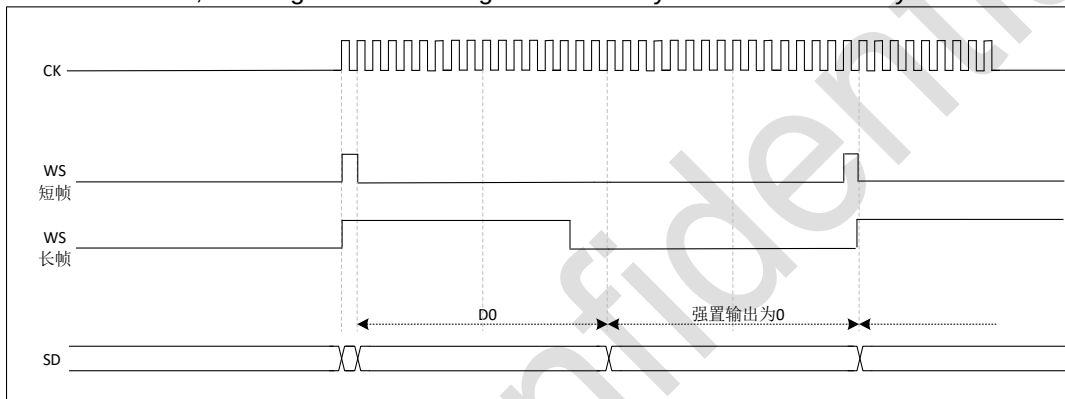


Figure 34-32 I2S PCM standard (16-bit extension to 32-bit, CKPOL = 0)

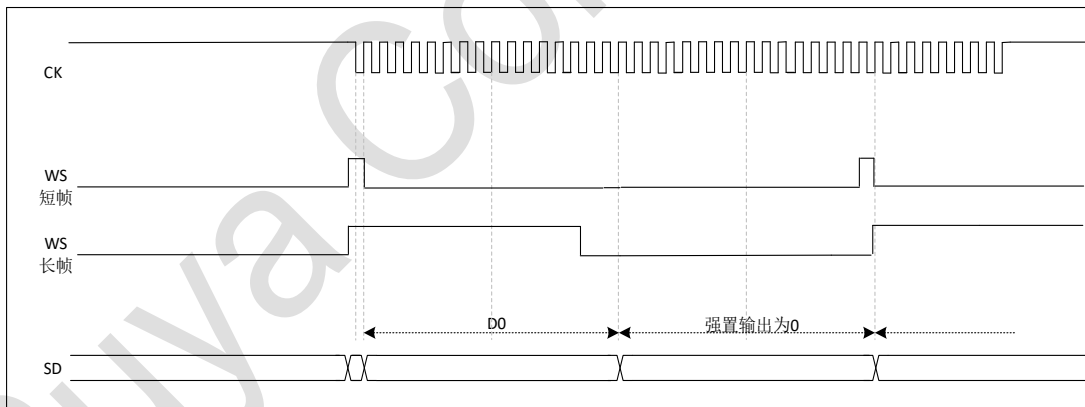


Figure 34-33 I2S PCM standard (16-bit extension to 32-bit, CKPOL = 1)

Regardless of the mode (master or slave) and the synchronization mode (short frame or long frame), even in the slave mode, it is necessary to specify the time difference between two consecutive frames of data and two synchronization signals by setting the DATLEN bit and the CHLEN bit of the SPI_I2 SCFGR register.

34.4.3. Clock generation

The I2S bit rate determines the data flow on the I2S data line and the I2S clock signal frequency. I.e I2S bitrate = number of bits per channel x number of channels x audio sampling frequency.

For a 16-bit audio, left and right channel, the I2S bit rate is calculated as follows:

$$\text{I2S bitrate} = 16 \times 2 \times f_s$$

If the packet length is 32 bits, there is:

$$\text{I2S bitrate} = 32 \times 2 \times f_s$$

The definition of audio sampling frequency is shown in the following figure:



Figure 34-34 Definition of Audio Sampling Frequency

When the master mode is configured, a specific action needs to be taken to properly program the linear divider in order to communicate with the desired audio frequency.

The structure of the I2S clock generator is shown in the following figure:

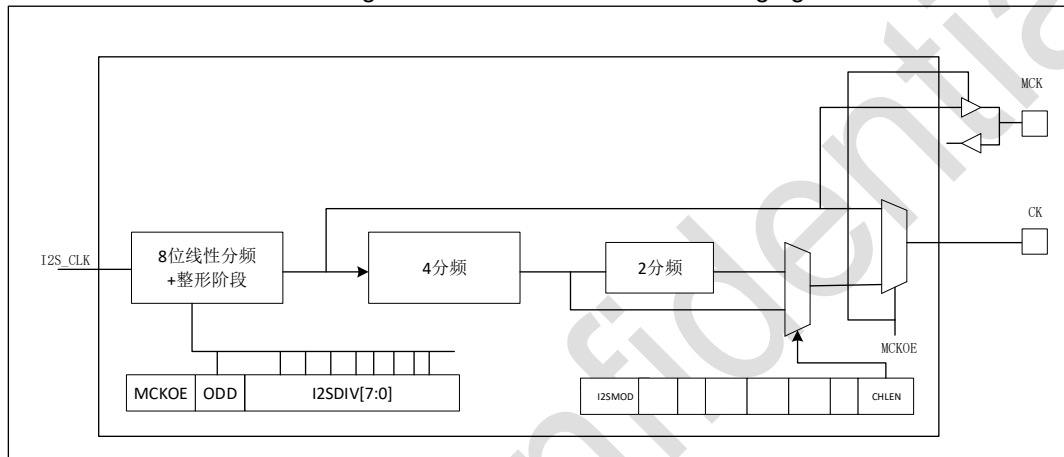


Figure 34-35 Structure of I2S Clock Generator

The I2S_CLK is generated by the RCC module and synchronized with the PCLK.

The sampling frequency of the audio may be 96 kHz, 48 kHz, 44.1 kHz, 32 kHz, 22.05 kHz, 16 kHz, 11.025 kHz, or 8 kHz (or any value within this range). In order to obtain the required frequency, it is obtained by setting the linear frequency divider according to the following formula:

34.4.3.1. I2S mode clock generation

When the master clock is output (the MCKOE bit of the register SPI_I2SPR is '1 '):

$$fs = f_{I2S\_CLK} / [256 \times ((2 \times I2SDIV) + ODD)]$$

When the output of the master clock is disabled (the MCKOE bit of the register SPI_I2SPR is '0 '):

$$\text{When the frame length of the channel is 16 bits, } fs = f_{I2S\_CLK} / [32 \times ((2 \times I2SDIV) + ODD)]$$

$$\text{When the frame length of the channel is 32 bits, } fs = f_{I2S\_CLK} / [64 \times ((2 \times I2SDIV) + ODD)]$$

34.4.3.2. PCM mode clock generation

When the master clock is output (the MCKOE bit of the register SPI_I2SPR is '1 '):

$$fs = f_{I2S\_CLK} / [128 \times ((2 \times I2SDIV) + ODD)]$$

When the output of the master clock is disabled (the MCKOE bit of the register SPI_I2SPR is '0 '):

$$\text{When the frame length of the channel is 16 bits, } fs = f_{I2S\_CLK} / [16 \times ((2 \times I2SDIV) + ODD)]$$

$$\text{When the frame length of the channel is 32 bits, } fs = f_{I2S\_CLK} / [32 \times ((2 \times I2SDIV) + ODD)]$$

Using the standard 8MHz HSE clock to obtain precise audio frequencies, as shown in the table below.

Table 34-2 Audio Frequency Accuracy Using 8MHz HSE

SYSCLK (MHz)	I2SDIV		ODD		MCLK	De- sired Fs (Hz)	Real fs (Hz)		Error	
	16-bit wide	32-bit wide	16-bit wide	32-bit wide			16-bit wide	32-bit wide	16-bit wide	32-bit wide
72	11	6	1	0	None	96000	97826.09	93750	1.90 %	2.34 %
72	23	11	1	1	None	48000	47872.34	48913.04	0.27 %	1.90 %
72	25	13	1	0	None	44100	44117.65	43269.23	0.04 %	1.88 %
72	35	17	0	1	None	32000	32142.86	32142.86	0.45 %	0.45 %

72	51	25	0	1	None	22050	22058.82	22058.82	0.04 %	0.04 %
72	70	35	1	0	None	16000	15957.45	16071.43	0.27 %	0.45 %
72	102	51	0	0	None	11025	11029.41	11029.41	0.04 %	0.04 %
72	140	70	1	1	None	8000	8007.117	7978.723	0.09 %	0.27 %
72	2	2	0	0	Yes	96000	70312.5	70312.5	26.76 %	26.76 %
72	3	3	0	0	Yes	48000	46875	46875	2.34 %	2.34 %
72	3	3	0	0	Yes	44100	46875	46875	6.29 %	6.29 %
72	4	4	1	1	Yes	32000	31250	31250	2.34 %	2.34 %
72	6	6	1	1	Yes	22050	21634.62	21634.62	1.88 %	1.88 %
72	9	9	0	0	Yes	16000	15625	15625	2.34 %	2.34 %
72	13	13	0	0	Yes	11025	10817.31	10817.31	1.88 %	1.88 %
72	17	17	1	1	Yes	8000	8035.714	8035.714	0.45 %	0.45 %

34.4.4. I2S transmission

34.4.4.1. Master mode

The I2S can be configured in master mode. This means that the serial clock is generated on the CK pin as well as the Word Select signal WS. The master clock (MCK) may or may not be output, specifically controlled by the MCKOE bit in the SPI_I2SPR register.

The configuration process is as follows:

1. Set the I2SDIV[7:0] bits in the SPI_I2SPR register to define the serial clock baud rate that matches the audio sampling frequency. The ODD bit in the SPIx_I2SPR register also has to be defined.
2. Select the CKPOL bit to define the steady level for the communication clock. If the master clock MCK needs to be supplied to an external DAC/ADC audio device, the MCKOE position of the register SPI_I2SPR is set to '1', and the values of I2SDIV and ODD are calculated according to different MCK output states.
3. Set the I2SMOD bit in the SPI_I2SCFGR register to '1' to enable the I2S function, set the I2SSTD[1:0] and PCMSYNC bits to select the I2S standard used, and set CHLEN to select the number of data bits per channel. Also set the I2SCFG [1: 0] of the register SPI_I2SCFGR to select the I2S main mode and direction (transmitter or receiver).
4. If needed, enable the required interrupt and DMA functions by configuring the SPI_CR2 register.
5. The I2SE bit in the SPI_I2SCFGR register must be set to '1'.

The pins WS and CK need to be configured to output mode. MCK is also an output, if the MCKOE bit in SPI_I2SPR is set.

Sending process

When one and a half word (16 bits) of data is written into the transmission buffer area, the transmission flow starts.

Lets assume the first data written into the Tx buffer corresponds to the left channel data. When data is moved from the transmission buffer area to the shift register, the flag bit TXE is set to '1', and at this time, the data corresponding to the right channel is written to the transmission buffer area. The CHSIDE flag indicates which channel is to be transmitted. When the TXE flag is set to 1, the CHSIDE flag is meaningful because it is updated when the TXE becomes high.

A full frame has to be considered as a left channel data transmission followed by a right channel data transmission. It is not possible to have a partial frame where only the left channel is sent.

The data half-word is parallel loaded into the 16-bit shift register during the first bit transmission, and then shifted out, serially, to the MOSI/SD pin, MSB first. The TXE flag is set after each transfer from the Tx buffer to the shift register and an interrupt is generated if the TXEIE bit in the SPI_CR2 register is set.

The operation of writing data depends on the selected I2S standard. To ensure a continuous audio data transmission, it is mandatory to write the SPI_DR register with the next data to transmit before the end of the current transmission.

To close I2S by clearing I2SE, you must wait for TXE = 1 and BSY = 0.

Receiving Process

The receiving process is similar to the sending process, except that the receiving mode is selected in step 3.

Whatever the data or channel length, the audio data are received by 16-bit packets. This means that each time the Rx buffer is full, the RXNE flag is set and an interrupt is generated if the RXNEIE bit is set in SPI_CR2 register. Depending on the data and channel length configuration, the audio value received for a right or left channel may result from one or two receptions into the Rx buffer.

Clearing the RXNE bit is performed by reading the SPIx_DR register.

CHSIDE is updated after each reception. It is sensitive to the WS signal generated by the I2S cell. The operation of reading the data depends on the selected I2S standard.

If data are received while the previously received data have not been read yet, an overrun is generated and the OVR flag is set. If the ERRIE bit is set in the SPI_CR2 register, an interrupt is generated to indicate the error.

Turn off I2S

To switch off the I2S, specific actions are required to ensure that the I2S completes the transfer cycle properly without initiating a new data transfer.

Specific operation process:

To switch off the I2S, specific actions are required to ensure that the I2S completes the transfer cycle properly without initiating a new data transfer. The sequence depends on the configuration of the data and channel lengths, and on the audio protocol mode selected.

- 16-bit data extended to 32-bit channel length (DATLEN = 00 and CHLEN = 1) using LSB (low bit) alignment mode (I2SSTD = 10)
 - Wait for the second-to-last (n-1) RXNE=1;
 - Wait for 17 I2S clock cycles (using software delay);
 - Disable I2S (I2SE=0).
- 16-bit data extended to 32-bit channel length (DATLEN=00 and CHLEN=1), using MSB (most significant bit) alignment, I2S or PCM mode (I2SSTD=00, I2SSTD=01, or I2SSTD=11 respectively)
 - Wait for the last RXNE=1;
 - Wait for 1 I2S clock cycle (using software delay);
 - Disable I2S (I2SE=0).
- All other combinations of DATLEN and CHLEN, regardless of the audio mode selected by I2SSTD, turn I2S off using the following manner:
 - Wait for the second-to-last (n-1) RXNE=1;
 - Wait for one I2S clock cycle (using software delay);
 - Disable I2S (I2SE=0).

34.4.4.2. Slave mode

For the slave configuration, the I2S can be configured in transmission or reception mode. The operating mode is following mainly the same rules as described for the I2S master configuration. In slave mode, there is no clock to be generated by the I2S interface. The clock and WS signals are input from the external master connected to the I2S interface. There is then no need, for the user, to configure the clock.

The configuration steps are as follows:

1. Set the I2SMOD bit in the SPI_I2SCFGR register to activate the I2S function; set I2SSTD[1:0] to select the I2S standard; set DATLEN[1:0] to choose the number of data bits; set CHLEN to select the number of bits per channel. The I2SCFG [1: 0] of the set register SPI_I2SCFGR selects the data direction of the I2S slave mode.
2. As needed, configure the SPI_CR2 register to enable the required interrupt and DMA functions.
3. The I2SE bit in the SPI_I2SCFGR register must be set to '1'.

Sending process

When the external host transmits the clock signal, and when the NSS_WS signal requests the transmission of data, the transmission flow starts. The slave must be enabled and written to the I2S data register before the external master can begin communication.

For the I2S, MSB justified and LSB justified modes, the first data item to be written into the data register corresponds to the data for the left channel. When communication starts, data is transferred from the transmit buffer to the shift register, and then the flag bit TXE is set to '1'; At this time, the data item corresponding to the right channel is written into the I2S data register.

The CHSIDE flag indicates which channel is to be transmitted. In the slave mode, the CHSIDE depends on the WS signal from the external master I2S, as compared to the transmission flow of the master mode. This means that the slave has to prepare the first data to send before receiving the clock signal generated by the master. WS assertion corresponds to left channel transmitted first.

Note: When the first clock from the master appears on the CK line, I2SE must be set at least 2 PCLK cycles in advance.

The data half-word is parallel-loaded into the 16-bit shift register (from the internal bus) during the first bit transmission, and then shifted out serially to the MOSI/SD pin MSB first. The TXE flag is set after each transfer from the Tx buffer to the shift register and an interrupt is generated if the TXEIE bit

in the SPI_CR2 register is set. Note that the TXE flag should be checked to be at 1 before attempting to write the Tx buffer. The operation of writing data depends on the selected I2S standard, as detailed in the "Supported Audio Protocols" section.

To ensure a continuous audio data transmission, it is mandatory to write the SPI_DR register with the next data to transmit before the end of the current transmission. If the first clock edge of the next data communication comes before data has been written to the SPI_DR register, the underflow flag will be set to '1' and an interrupt may occur. This indicates to the software that the transferred data are wrong. If the ERRIE position of the SPI_CR2 register is '1', an interrupt will be generated when the UDR flag in the SPI_SR register becomes '1'. In this case, I2S must be turned off and data transfer restarted from the left channel.

To turn off I2S by clearing the I2SE bit, you must first wait for TXE = 1 and BSY = 0.

Receiving Process

The receive flow is similar to the send flow, except that the receive mode is selected by register I2SCFG [1: 0] in step 1 of the "Configuration Flow".

Whatever the data length or the channel length, the audio data are received by 16-bit packets. This means that each time the RX buffer is full, the RXNE flag in the SPI_SR register is set and an interrupt is generated if the RXNEIE bit is set in the SPI_CR2 register. Depending on the data length and channel length configuration, the audio value received for a right or left channel may result from one or two receptions into the RX buffer.

The CHSIDE is updated each time data is received (to be read out from the SPI_DR), which corresponds to the WS signal generated by the I2Sunit. Clearing the RXNE bit is performed by reading the SPI_DR register. The operation of reading data depends on the selected I2S standard.

When the previously received data is not read out and new data is received, overflow is generated, and the flag bit OVR is set to '1'; If the ERRIE bit of the register SPI_CR2 is '1', an interrupt is generated indicating that an error has occurred.

To turn off the I2S function, the I2SE bit needs to be cleared '0' when the last RXNE = 1 is received.

Note: The external host device needs to have the ability to send/receive 16-bit or 32-bit packets through the audio channel.

34.4.5. I2S logo

34.4.5.1. Status flag

Three status flags are provided for the application to fully monitor the state of the I2S bus.

Busy flag bit (BSY)

The BSY flag is set and cleared by the hardware (writing this bit is invalid), and this flag bit indicates the status of the I2S communication layer.

When this bit is '1', it indicates that I2S communication is in progress, with one exception: in primary reception mode (I2SCFG = 11), the BSY flag is always low during reception.

Before the software wants to shut down the I2S module, the BSY flag can be used to detect whether the transmission is finished, which can avoid corrupting the last transmission.

The BSY flag is set when a transfer starts, except when the I2S is in master receiver mode.

The BSY flag is cleared:

- When the transmission ends (except for the main transmit mode, where communication is continuous);
- When the I2S is disabled.

When communication is continuous:

- In master transmit mode, the BSY flag is kept high during all the transfers;
- In slave mode, the BSY flag goes low and lasts for 1 I2S clock cycle between each data item transmission.

Note: Do not use the BSY flag to handle each data transmission or reception. It is better to use the TXE and RXNE flags instead.

Send buffer null flag bit (TXE)

When set, this flag indicates that the Tx buffer is empty and the next data to be transmitted can then be loaded into it. The TXE flag is reset when the Tx buffer already contains data to be transmitted.

When I2S is turned off (I2SE bit is '0'), the flag bit is also '0'.

Receive buffer non-empty flag bit (RXNE)

When set, this flag indicates that there are valid received data in the RX Buffer. It is reset when SPI_DR register is read.

Channel Flag (CHSIDE)

In transmission mode, this flag is refreshed when TXE goes high. It indicates the channel side to which the data to transfer on SD has to belong. In case of an underrun error event in slave transmission mode, this flag is not reliable and I2S needs to be switched off and switched on before resuming the communication.

In reception mode, this flag is refreshed when data are received into SPI_DR. It indicates from which channel side data have been received. If an error occurs (e.g., overflow OVR), this flag bit is meaningless and I2S needs to be turned off and turned on again (and the configuration of I2S should be modified if necessary).

This flag has no meaning in the PCM standard (for both Short and Long frame modes).

When the OVR or UDR flag in the SPI_SR is set and the ERRIE bit in SPI_CR2 is also set, an interrupt is generated. This interrupt can be cleared by reading the SPI_SR status register.

34.4.5.2. Error flags

There are two error flags for the I2S cell.

Underflow flag bit (UDR)

In slave transmission mode, this flag is set when the first clock for data transmission appears while the software has not yet loaded any value into SPI_DR. It is available when the I2SMOD bit in the SPI_I2SCFGR register is set. An interrupt may be generated if the ERRIE bit in the SPI_CR2 register is set.

The UDR bit is cleared by a read operation on the SPI_SR register.

Overflow flag bit (OVR)

This flag is set when data are received and the previous data have not yet been read from the SPI_DR register. As a result, the incoming data are lost. An interrupt may be generated if the ERRIE bit is set in the SPI_CR2 register. In this case, the receive buffer contents are not updated with the newly received data from the transmitter device. A read operation to the SPI_DR register returns the previous correctly received data. All other subsequently transmitted half-words are lost.

Clearing the OVR bit is done by a read operation on the SPI_DR register followed by a read access to the SPI_SR register.

34.4.6. I2S DMA

In I2S mode, DMA works exactly the same way as in SPI mode. There is no difference except that there is no CRC function in I2S mode because there is no data transmission protection mechanism.

34.4.7. I2S interrupts

The I2S interrupt requests are as follows:

Table 34-3 I2S Interrupt

Interrupt event	Event flag	Enable control bit
Transmit buffer empty flag	TXE	TXEIE
Receive buffer not empty flag	RXNE	RXNEIE
Underrun error	OVR	ERRIE
Overrun error	UDR	

34.5. SPI/I2S Register

34.5.1. SPI control register 1 (SPI_CR1)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDIMOD E	BIDIO E	CRCE N	CRCNE XT	DF F	RXONL Y	SS M	SSI	LSBFIR ST	SP E	BR[2:0]			MST R	CPO L	CPH A
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	BIDIMODE	RW	0	Bidirectional data mode enable 0: 2-line unidirectional data mode selected 1: 1-line bidirectional data mode selected Note: This bit is not used in I2S mode.
14	BIDIOE	RW	0	Output enable in bidirectional mode

				This bit combined with the BIDIMODE bit selects the direction of transfer in bidirectional mode. 0: Output disabled (receive only mode) 1: Output enabled (send only mode) In main mode, use the MOSI pin; In slave mode, use the MISO pin. Note: This bit is not used in I2S mode.
13	CRCEN	RW	0	Hardware CRC check enabled. 0: CRC calculation disabled; 1: CRC calculation enabled. Note: This bit can only be written when SPI is disabled (SPE = 0), otherwise an error will occur. Note: This bit is not used in I2S mode.
12	CRCNEXT	RW	0	The next one sends the CRC. 0: Next transmit value is from Tx buffer. 1: Next transmit value is from Tx CRC register. Note: This bit has to be written as soon as the last data is written in the SPI_DR register. Note: This bit is not used in I2S mode.
11	DFF	RW	0	Data frame format 0: 8-bit data frame format is selected for transmission/reception 1: 16-bit data frame format is selected for transmission/reception Note: This bit should be written only when SPI is disabled (SPE = '0') for correct operation. Note: This bit is not used in I2S mode.
10	RXONLY	RW	0	Receive only This bit combined with the BIDIMODE bit selects the direction of transfer in 2-line unidirectional mode. In the configuration of multiple slaves, this position 1 is on the unaccessed slave, so that only the accessed slave has output, thus causing no data collision on the data line. 0: Full-duplex (Transmit and receive) 1: Output disabled (receive-only mode) Note: This bit is not used in I2S mode.
9	SSM	RW	0	Software slave management. When the SSM bit is set, the NSS pin input is replaced with the value from the SSI bit. 0: Disable software slave management 1: Enable software slave management Note: This bit is not applicable to I2S mode and SPI TI mode.
8	SSI	RW	0	Internal slave selection. This bit has an effect only when the SSM bit is set. The value of this bit is forced onto the NSS pin and the IO value of the NSS pin is ignored. Note: This bit is not applicable to I2S mode and SPI TI mode.
7	LSBFIRST	RW	0	Frame format 0: MSB transmitted first 1: LSB transmitted first Note: This bit should not be changed when communication is ongoing. Note: This bit is not applicable to I2S mode and SPI TI mode.
6	SPE	RW	0	SPI enable 0: Peripheral disabled 1: Peripheral enabled Note: This bit is not used in I2S mode.
5:3	BR[2:0]	RW	0	Baud rate control 000: $f_{PCLK}/2$ 001: $f_{PCLK}/4$ 010: $f_{PCLK}/8$ 011: $f_{PCLK}/16$ 100: $f_{PCLK}/32$ 101: $f_{PCLK}/64$ 110: $f_{PCLK}/128$

				111: $f_{PCLK}/256$ Note: This bit should not be changed when communication is ongoing. Note: This bit is not used in I2S mode.
2	MSTR	RW	0	Host selection. 0: Configured as slave 1: Configure as host Note: This bit should not be changed when communication is ongoing. Note: This bit is not used in I2S mode.
1	CPOL	RW	0	Clock polarity 0: In idle state, SCK remains low 1: SCK maintains high level in idle state Note: This bit should not be changed when communication is ongoing. Note: This bit is not applicable to I2S mode and SPI TI mode except in the case of CRC being applied in TI mode.
0	CPHA	RW	0	Clock phase 0: The first clock transition is the first data capture edge 1: The second clock transition is the first data capture edge Note: This bit should not be changed when communication is ongoing. Note: This bit is not applicable to I2S mode and SPI TI mode except in the case of CRC being applied in TI mode.

34.5.2. SPI control register 2 (SPI_CR2)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res.	SSOE	TXDMAEN	RXDMAEN
RW								RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	FRF	RW	0	Frame format: 0: SPI Motorola mode 1: SPI TI mode You can only operate on this bit after SPI is disabled (SPE = 0)
14:8	Reserved	-	-	Reserved
7	TXEIE	RW	0	Tx buffer empty interrupt enable 0: TXE interrupt masked 1: TXE interrupt not masked. Used to generate an interrupt request when the TXE flag is set.
6	RXNEIE	RW	0	RX buffer not empty interrupt enable 0: RXNE interrupt masked 1: RXNE interrupt not masked. Used to generate an interrupt request when the RXNE flag is set.
5	ERRIE	RW	0	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled When CRCERR, OVR, or MODF is 1, an interrupt request is generated.
4	CLRTXFIFO	RW	0	Clear TXFIFO Set by software and reset by hardware 0: No action 1: Clear TXFIFO Note: This bit should be written only when SPI is disabled (SPE = '0') for valid operation.
3	Reserved	-	-	Reserved
2	SSOE	RW	0	SS output enable 0: Disable SS output in main mode, the device can work in multi-host mode

				1: Turn on SS output in main mode, and the device cannot work in multi-host mode. Note: It is not used in I2S mode.
1	TXDMAEN	RW	0	Send buffer DMA enabled. 0: Tx buffer DMA disabled 1: Tx buffer DMA enabled When TXE = 1, a DMA request is issued.
0	RXDMAEN	RW	0	Receive buffer DMA enabled. 0: Rx buffer DMA disabled 1: Rx buffer DMA enabled When TXE = 1, a DMA request is issued.

34.5.3. SPI status register (SPI_SR)

Offset address: 0x08

Reset value: 0x0000 0002

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	FTLVL [1:0]		FRLVL [1:0]		FRE	BSY	OVR	MODF	CRCERR	UDR	CHSIDE	TXE	RXNE
			R	R	R	R	R	R	R	R	RC_W0	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12:11	FTLVL	R	0	FIFO send level. Set and cleared by hardware 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO full (when the FIFO threshold is greater than 1/2, it is considered full) Note: It is not used in I2S mode.
10:9	FRLVL	R	0	FIFO reception level. Set and cleared by hardware 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO full Note: I2S mode and SPI with CRC check are not used in receive mode only.
8	FRE	R	0	FRE: Frame format is wrong. This flag is used for SPI TI slave mode. This flag is set to 1 by the hardware and reset by the software when SPI_SR is read. 0: No frame format error occurred 1: Frame format error occurred
7	BSY	R	0	Busy flag 0: SPI not busy 1: SPI is busy in communication or Tx buffer is not empty
6	OVR	R	0	Overflow flag 0: No overflow error 1: Overflow occurred This flag is set by hardware and reset by a software sequence.
5	MODF	R	0	Mode failure. 0: No mode failure 1: Mode failure occurs This flag is set by hardware and reset by a software sequence. Note: Not applicable in I2S mode.
4	CRCERR	R	0	CRC error flag. 0: The received CRC value matches the value in the SPI_RXCR register; 1: The received CRC value does not match the value in the SPI_RXCR register. This flag is set by hardware and cleared by software writing 0. Note: Not applicable in I2S mode.
3	UDR	R	0	Underflow flag bit. 0: No underflow occurred 1: An underflow error occurred. Hardware set, software sequence cleared. Note: Not applicable in SPI mode.

2	CHSIDE	R	0	Channel control. 0: transmit or receive left channel 1: Transmit or receive the right channel. Note: Not applicable in SPI and I2S PCM modes.
1	TXE	R	1	Transmit buffer empty 0: Tx buffer not empty 1: Tx buffer empty
0	RXNE	R	0	Receive buffer not empty 0: Rx buffer not empty 1: Rx buffer empty

34.5.4. SPI Data Register (SPI_DR)

Offset address: 0x0C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	DR[15:0]	RW	0	Data register. Data received or to be transmitted. The data register serves as an interface between the Rx and Tx FIFOs. When the data register is read, RxFIFO is accessed while the write to data register accesses Tx FIFO. Note: Depending on the DFF bits (data frame width selection), data transmission or reception is 8-bit or 16-bit. For 8-bit data frames, the data registers are transmitted and received based on the right-aligned 8-bit data. When in reception mode, the MSB of the register (SPI_DR[15:8]) is forced to 0. For a 16-bit data frame, the data registers are 16-bit and the entire DR [15: 0] is used for transmission and reception.

34.5.5. SPI CRC polynomial register (SPI_CRCPR)

Offset address: 0x10

Reset value: 0x0000 0007

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CRCPOLY [15: 0]	RW	0x0007	CRC polynomial register. This register contains the polynomial for the CRC calculation. The CRC polynomial (0x0007) is the reset value of this register. Another polynomial can be configured as required. Note: Not applicable in I2S mode. Note: The polynomial value should only be odd. No even values are supported.

34.5.6. SPI Receive CRC Register (SPI_RXCRC)

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXCRC [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

34.5.7. SPI Send CRC Register (SPI_TXCRC)

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	RXCRC[15:0]	R	0	When CRC calculation is enabled in the Receive CRC register, RXCRC [15: 0] contains the CRC value calculated from the received bytes. This register is reset when the CRCEN bit in SPI_CR1 register is written to 1. The CRC is calculated serially using the polynomial programmed in the SPI_CRCPR register. Only the 8 LSB bits are considered when the CRC frame format is set to be 8-bit length. CRC calculation is done based on any CRC8 standard. The entire 16-bits of this register are considered when a 16-bit CRC frame format is selected. CRC calculation is done based on any CRC16 standard. Note: A read to this register when the BSY Flag is set could return an incorrect value. Note: Not applicable in I2S mode.

Offset address: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXCRC [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	TXCRC[15:0]	R	0	When CRC calculation is enabled in the Receive CRC register, RXCRC [15: 0] contains the CRC value calculated from the received bytes. This register is reset when the CRCEN bit in SPI_CR1 register is written to 1. The CRC is calculated serially using the polynomial programmed in the SPI_CRCPR register. Only the 8 LSB bits are considered when the CRC frame format is set to be 8-bit length. CRC calculation is done based on any CRC8 standard. The entire 16-bits of this register are considered when a 16-bit CRC frame format is selected. CRC calculation is done based on any CRC16 standard. Note: A read to this register when the BSY Flag is set could return an incorrect value. Note: Not applicable in I2S mode.

34.5.8. SPI I2S configuration register (SPI_I2SCFGR)

Offset address: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	I2SMOD	I2SE	I2SCFG [1:0]		PCMSYNC	Res.	I2SSTD [1:0]		CKPOL	DTALEN[1:0]		CHLEN
				RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	I2SMOD	RW	0	I2S mode selection. 0: Select SPI mode; 1: Select I2S mode. Note: This bit can only be configured when SPI or I2S is disabled. Note: If the register does not support I2S functionality, it is fixed at 0.
10	I2SE	RW	0	I2S enabled. 0: Disable I2S;

				1: Enable I2S. Note: Not applicable in SPI mode. Note: If the register does not support I2S functionality, it is fixed at 0.
9:8	I2SCFG	RW	0	I2S mode settings. 00: Slave transmit; 01: Slave receive; 10: Master transmit; 11: Master receive. Note: This bit can only be configured when I2S is disabled. Not applicable in SPI mode. Note: If the register does not support I2S functionality, it is fixed at 0.
7	PCMSYNC	RW	0	PCM frame synchronization. 0: Short frame sync; 1: Long frame sync. Note: This bit is only meaningful if I2SSTD = 11 (using the PCM standard). Not applicable in SPI mode. Note: If the register does not support I2S functionality, it is fixed at 0.
6	Reserved	-	-	Reserved
5:4	I2SSTD	RW	0	I2S Standard Select. 00: I2S Philips standard; 01: High byte alignment standard (left-aligned); 10: Low byte alignment standard (right-aligned); 11: PCM standard. Note: For correct operation, these bits should be configured when the I2S is disabled. Not applicable in SPI mode. Note: If the register does not support I2S functionality, it is fixed at 0.
3	CKPOL	RW	0	Invalid state clock polarity. 0: I2S clock inactive state is low level; 1: I2S clock inactive state is high level. Note: For correct operation, these bits should be configured when the I2S is disabled. Not applicable in SPI mode. Note: If the register does not support I2S functionality, it is fixed at 0.
2:1	DATLEN	RW	0	Length of data to be transmitted. 00: 16-bit data length; 01: 24-bit data length; 10: 32-bit data length; 11: Not allowed. Note: For correct operation, these bits should be configured when the I2S is disabled. Not applicable in SPI mode. Note: If the register does not support I2S functionality, it is fixed at 0.
0	CHLEN	RW	0	Channel length (number of data bits per audio channel). 0: 16-bit width; 1: 32-bit width. The write operation of this bit only makes sense if DATLEN = 00, otherwise the channel length is fixed to 32 bits by hardware. Note: For correct operation, these bits should be configured when the I2S is disabled. Not applicable in SPI mode. Note: If the register does not support I2S functionality, it is fixed at 0.

34.5.9. SPI_I2S Prescaler Register (SPI_I2SPR)

Offset address: 0x20

Reset value: 0x0000 0000

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

Res.	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV [7:0]						
						RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	MCKOE	RW	0	Host clock output enabled 0: Disable master clock output; 1: Enable master clock output. Note: For correct operation, these bits should be configured when the I2S is disabled. Use this bit only in I2S host mode. Not applicable in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
8	ODD	RW	0	The odd factor of the prescaler. 0: Actual division factor = I2SDIV * 2; 1: Actual division factor = (I2SDIV * 2) + 1. Note: For correct operation, these bits should be configured when the I2S is disabled. Use this bit only in I2S host mode. Not applicable in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
7:0	I2SDIV	RW	0	I2S linear prescaler. It is disabled to set I2SDIV [7: 0] = 0 or I2SDIV [7: 0] = 1. Note: For correct operation, these bits should be configured when the I2S is disabled. Use this bit only in I2S host mode. Not applicable in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.

35. Voltage reference buffer (V_{REFBUF})

35.1. Introduction to V_{REFBUF}

This chip has a built-in voltage reference buffer (V_{REFBUF}). When using the internal V_{REFBUF} , a $1\mu\text{F}$ capacitor must be externally connected to the PA0 pin, and PA0 must be configured in analog mode. If PA0 is multiplexed as GPIO, the internal V_{REFBUF} cannot be used. It also supports external reference voltage input via PA0 for ADC or DAC usage, in which case PA0 must also be configured in analog mode.

35.2. V_{REFBUF} register

35.2.1. V_{REFBUF} Control Register ($V_{\text{REFBUF\_CR}}$)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	$V_{\text{REFBUF\_EN}}$	$V_{\text{REFBUF\_OUT\_SEL}}[2:0]$		
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3	$V_{\text{REFBUF\_EN}}$	RW	0	V_{REFBUF} enable. 0: V_{REFBUF} disabled 1: V_{REFBUF} enabled
2:0	$V_{\text{REFBUF\_OUT\_SEL}}[2:0]$	RW	0	V_{REFBUF} module output voltage selection (the voltage value represented by this register may vary for different products). 000: 1.024 V 001: 1.5 V 010: 2.048 V 011: 2.5 V 100: 0.6 V Others: Reserved Note: The Trim values corresponding to each voltage are as follows at the Flash address: 0.6V calibration value storage address: 0x1FFF 1C24 1.024V calibration value storage address: 0x1FFF 1C28 1.5V calibration value storage address: 0x1FFF 1C2C 2.048V calibration value storage address: 0x1FFF 1C30 2.5V calibration value storage address: 0x1FFF 1C34

36. Debug support (DBG)

36.1. Debug Introduction

This chip is based on the Cortex®-M0+ CPU, which includes hardware extensions for advanced debugging features. The debug extensions allow the core to halt when fetching instructions (instruction breakpoint) or accessing data (data breakpoint). When stopped, the core's internal state and the system's external state may be examined. Once examination is complete, the core and the system may be restored and program execution resumed.

The debug functions are used by the debug host when connecting and debugging the MCU, with the debug interface being SWD. The debug functionality in the Cortex®-M0+ CPU core is part of the ARM CoreSight design suite.

The Cortex®-M0+ provides integrated on-chip debug support, consisting of the following components:

- SW-DP: Serial Wire
- BPU: Breakpoint Unit
- DWT: Data Watchpoint and Trigger

It also includes debug features dedicated to the device:

- Flexible debug pin assignment, SWIO@PA4/PB6, SWCLK@PA5/PB5
- MCU Debug Box (supports low-power modes, controls peripheral clocks, etc.)

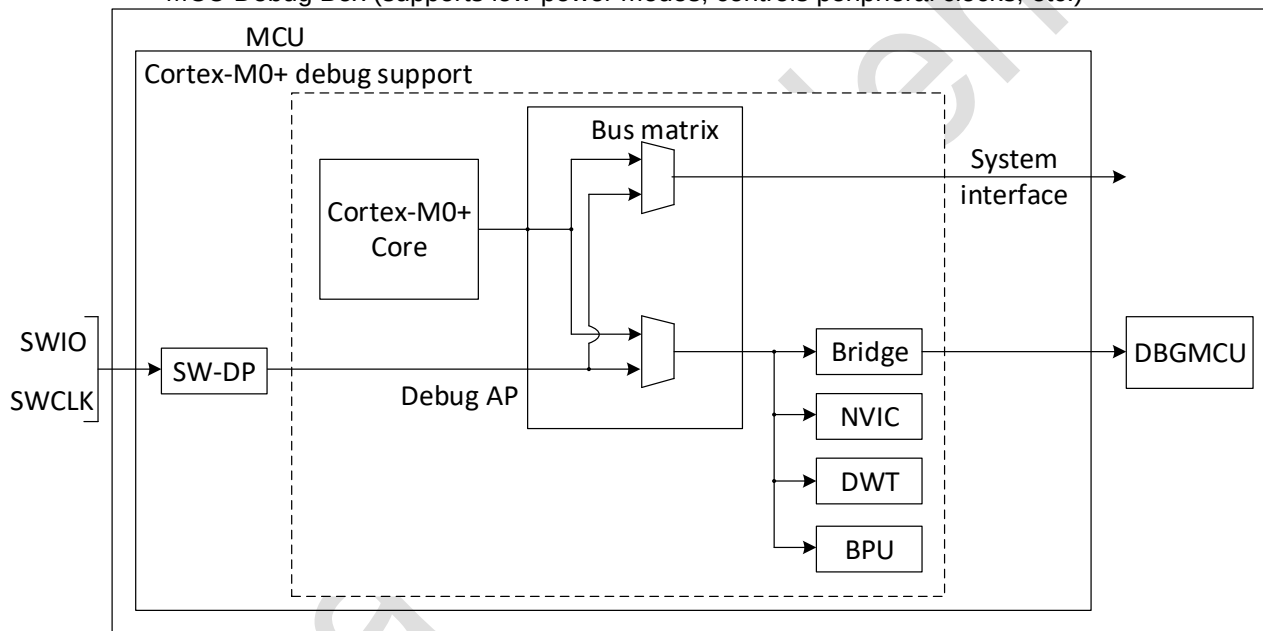


Figure 36-1 DBG block diagram

36.2. Pinout and Debug Port Pins

36.2.1. SWD Port

There are two pins related to debug functionality, which can be used as GPIO alternate functions. For the visibility of these two pins in the package form, please refer to the corresponding datasheet.

Table 36-1 SWD Debug Port

SW-DP Pin Names	SW Debug Disconnect		Pin assignment
	Type	Debug Assignment	
SWDIO	I/O	Serial Wire Data Input/Output	PA4/PB6 Note
SWCLK	I	Serial Wire Clock	PA5/PB5 Note

Note: Supports two sets of SWDIO/SWDCLK mappings. For details, refer to the GPIO chapter.

36.2.2. SW-DP Pin Division

After chip reset (system reset or power-on reset), the pins used for SW-DP are assigned as dedicated pins for immediate use by the debug host.

However, the chip provides the possibility to disable the SWD port and release these pins for GPIO use.

36.2.3. SWD Pin Pull-up/Pull-down

Once the SWD I/O is released by software, the GPIO controller takes control of these two pins. The reset states of the GPIO control registers put the I/Os in the equivalent states:

- SWDIO: input pull-up
- SWCLK: input pull-down

Having embedded pull-ups and pull-downs removes the need to add external resistors.

36.3. Debugger ID

The ID code is stored in the chip memory.

After power-on, the hardware reads the factory configuration bytes of Flash at address 0x1FFF 1FF4 and loads them into the DBG_IDCODE register. The content of this register can be accessed via SWD and software.

36.4. SWD Port

36.4.1. SWD protocol introduction

This is a synchronous serial communication protocol that uses the following two pins:

- SWCLK: Clock signal from the host to the chip
- SWDIO: bidirectional

Using this protocol, two register sets (DPACC register set and APACC register set) can be read and written simultaneously. During data transmission, the LSB is sent first.

For SWDIO bidirectional management, the line must be pulled up on the circuit board (Arm recommends 100 kΩ).

Every time the direction of SWDIO is changed in the protocol, a turnaround time is inserted, during which the line is neither driven by the host nor the target. By default, this turnaround time is one bit time, but it can be adjusted by configuring the SWCLK frequency.

36.4.2. SWD protocol sequence

Each sequence consist of three phases:

- Packet request (8 bits) transmitted by the host
- Acknowledge response (3 bits) transmitted by the target
- Data transfer phase (33 bits) transmitted by the host or the target

Table 36-2 Packet Request (8-bits)

Bit	Pin name	Description
0	Start	Must be "1"
1	ApnDP	0: DP Access 1: AP Access
2	RnW	0: Write Request 1: Read request
4:3	A[3:2]	Address field of the DP or AP registers
5	Parity	Single bit parity of preceding bits
6	Stop	0
7	Park	Not driven by the host. Must be read as "1" by the target because of the pull-up

For detailed descriptions of DPACC and APACC registers, refer to the Cortex®-M0+ TRM.

The packet request is always followed by a turnaround period (default 1 bit), during which neither the host nor the target drives the line.

Table 36-3 ACK Response (3bits)

Bit	Pin name	Description
[2:0]	ACK	001: FAULT 010: WAIT 100: OK

The ACK Response must be followed by a turnaround time only if it is a READ transaction or if a WAIT or FAULT acknowledge has been received.

Table 36-4 DATA Transfer (33bits)

Bit	Pin name	Description
[31:0]	WDATA or RDATA	Write or Read data
32	Parity	Single parity of the 32 data bits

The DATA transfer must be followed by a turnaround time only if it is a READ transaction.

36.4.3. SW-DP State Machine (Reset, Idle State, ID Code)

The State Machine of the SW-DP has an internal ID code which identifies the SW-DP. It follows the JEP-106 standard. This ID code is the default ARM code and is set to 0x0BC11477 (corresponding to Cortex®-M0+).

36.4.4. DP/AP Read-Write Access

- A read access to the DP has not yet been issued: the target response can be sent immediately (if ACK=OK) or delayed (if ACK=WAIT).
- A read access to the AP has been issued. This means that the result of the access is returned on the next transfer. If the next access to be performed is not an AP access, the DP-RDBUFF register must be read to obtain the result.
The READOK flag in the DP-CTRL/STAT register is updated with every AP read access or RDBUFF read request to indicate whether the AP read access was successful.
- The SW-DP has a write buffer (for DP or AP writes), allowing write operations to be accepted even when other operations are still pending. If the write buffer is full, the target acknowledgment response is WAIT. Exceptions are IDCODE reads, CTRL/STAT reads, or ABORT writes, which are accepted even when the write buffer is full.
- Due to the asynchronous clock domains SWCLK and HCLK, two additional SWCLK cycles are required after a write operation (after the parity bit) for the write to take effect internally. These cycles should be applied while driving the line low (idle state). This is particularly important when writing to the CTRL/STAT register to issue a power-up request. If the next transaction (requiring a power-up) occurs immediately, it will fail. SW-DP registers These registers can be accessed when APnDP=0.

36.4.5. SW-DP Registers

These registers can be accessed when APnDP=0.

Table 36-5 SW_DP Registers

A[3:2]	R/W	CTRLSEL bit or SELECT registers	Register	Note
00	Read	-	IDCODE	Manufacturer code set to Arm's default for Cortex®-M0+: 0x0BC11477 (identifies SW-DP)
00	Write	-	ABORT	
01	Read/W rite	0	DP-CTRL/STAT	Purpose: – Request system or debug power-up – Configure transfer operations for AP access – Control compare and verify operations – Read certain status flags (overflow and power-up acknowledgment)
01	Read/W rite	1	WIRE CONTROL	Purpose is to configure the physical serial port protocol (like the duration of the turnaround time)
10	Read	-	READ RESEND	Enables recovery of read data from corrupted debugger transfers without repeating the original AP transfer.
10	Write	-	SELECT	Used to select the current access port and active 4-word register window
11	Read/W rite	-	READ BUFFER	This read buffer is useful because an AP access has been issued (providing the result of the read AP request when executing the next AP transaction). This read buffer captures data in the AP, displayed as the result of the previous read, without initiating a new operation

36.4.6. SW-AP registers

These registers can be accessed when APnDP=1.

There are multiple AP registers, which are addressed by the following combination:

- Shift value A[3:2]
- Current value of the DP SELECT register

Table 36-6 SW_AP registers

Address	A[3:2] value	Description
0x0	00	Reserved, maintain reset value
0x4	01	DP CTRL/STAT register, used as ■ Request a system or debug power-up

		■ Configure the transfer operation for AP accesses ■ Control compare and verify operations ■ Read some status flags (overflow, power-up acknowledges)
0x8	10	DP SELECTION register: Used to select the current access port and active 4-word register window. ■ Bit 31:24: APSEL: Selects current AP ■ Bit 23:8: Reserved ■ Bit 7:4: APBANKSEL: Selects active 4-word register window within current AP ■ Bit 3:0: Reserved
0xC	11	DP RDBUFF register: Used to provide the debugger with the final result after an operation sequence (without requesting a new JTAG-DP operation)

36.5. Core debug registers

The kernel can be debugged through kernel debug registers. These registers are accessed via the debug access port. Consists of the following four registers:

Table 36-7 Core debug registers

Register	Description
DHCSR	32-bit Debug Halt Control and Status Register: This provides status information about the state of the processor enable core debug halt and step the processor
DCRSR	17-bit Debug Core Register Selector Register: This selects the processor register to transfer data to or from.
DCRDR	32-bit Debug Core Register Data Register: This register holds data read from and written to the processor selected by the DCRSR (selector) register.
DEMCR	32-bit Debug Exception and Monitor Control Register: This provides Vector Catching and Debug Monitor Control.

These registers are not reset by a system reset. They are only reset by a power-on reset. For more details, refer to the Cortex®-M0+ TRM.

To Halt on reset, it is necessary to:

- Enable bit 0 (VC_CORRESET) of the Debug Exception and Monitor Control Register
- Enable bit 0 (C_DEBUGEN) of the Debug Halt Control and Status Register

36.6. BPU (Breakpoint Unit)

The Cortex®-M0+ BPU implementation provides 4 breakpoint registers.

36.6.1. BPU functionality

The processor breakpoints implement PC based breakpoint functionality.

Refer to the ARMv6-M ARM and ARM Coresight Component Technical Reference Manual for more information about BPU Coresight identity registers, their addresses, and access types.

36.7. DWT (Data Watchpoint)

The Cortex®-M0+ DWT implementation provides 2 watchpoint registers.

36.7.1. DWT functionality

The processor watchpoints implement PC based watchpoint functionality.

36.7.2. DWT Program Counter Sampling Register

Processors implementing the Data Watchpoint Unit also implement the ARMv6-M optional DWT Program Counter Sampling Register (DWT_PCSR). This register permits a debugger to periodically sample the PC without halting the processor. This provides coarse grained profiling.

The Cortex®-M0+ DWT_PCSR records instructions that passed and failed the condition code.

36.8. MCU Debug Component (DBGMCU)

The MCU Debug Component assists debuggers by providing the following support:

- Low-power modes
- Clock control for timers and watchdogs during breakpoints
- Control whether the I2C SMBUS timeout function operates normally

36.8.1. Low-power mode debugging support

To enter low-power mode, the instruction WFI or WFE must be executed. The MCU enters low-power mode, either by stopping the CPU clock or reducing CPU power consumption.

The CPU is not allowed to turn off FCLK or HCLK during debugging. As these are required for the debugger connection, during a debug, they must remain active. The MCU integrates special means to allow the user to debug software in low-power modes.

For this, the debugger host must first set some debug configuration registers to change the low-power mode behavior:

- In sleep mode: FCLK and HCLK remain active. Consequently, this mode does not impose any restrictions on the standard debug features.
- In stop mode: The DBG_STOP bit must be set by the debugger in advance.

This enables the internal RC oscillator clock (HSI) to provide FCLK and HCLK in stop mode.

36.8.2. Timer/Watchdog/I2C Debugging

During a breakpoint, it is necessary to select how the counter, watchdog, and I2C SMBUS timeout counter should behave:

- Can continue counting during the breakpoint. This is usually required when a PWM is controlling a motor, for example.
- Can stop counting during the breakpoint. This is determined by the watchdog's characteristics.
- Can stop the I2C SMBUS timeout function.

36.9. DBGMCU registers

36.9.1. DBG ID Register (DBG_IDCODE)

Offset address: 0x00

Only 32-bit access supported. Read-only

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

Bit	Name	R/W	Reset Value	Function
31:0	DBG_ID	R	x	Emulator ID

36.9.2. Debug Configuration Register (DBG_CR)

This register configures the MCU's low-power mode during debugging.

It is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

If the debugger host does not support this feature, it is still possible for the user software to write to this register.

Offset address: 0x04

Reset value: 0x0000 0000 (not reset by system reset)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_STOP	DBG_SLEEP
														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved
1	DBG_STOP	RW	0	Debug Stop mode 0: (FCLK = off, HCLK = off). In stop mode, both HCLK and FCLK are turned off. When exiting stop mode, the clock configuration is the same as after power-on reset (system clock is HSI). Consequently, the software must reprogram the clock controller. 1: (FCLK = on, HCLK = on). When entering stop mode, HSI is not turned off, and FCLK and HCLK are generated by HSI. When exiting Stop mode, the software must reprogram the clock controller if the clock control needs to be changed.
0	DBG_SLEEP	RW	0	Debugging sleep mode.

				0: (FCLK ON, HCLK OFF). In Sleep mode, FCLK is provided by the previously configured system clock, and HCLK is turned off. Since sleep mode does not reset the configured clock system, the software does not need to reconfigure the clock after exiting sleep mode. 1: (FCLK=On, HCLK=On). In sleep mode, both the FCLK and HCLK clocks are provided by the previously configured system clock.

36.9.3. DBG APB Peripheral Freeze Register 1 (DBG_APB_FZ1)

This register configures the clock of TIMs, RTC, IWDG, WWDG, and I2C SMBUS peripherals during debug mode. The register is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DBG_LPTIM1_STOP	DBG_LPTIM2_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_I2C2_SMBUS_TIMEOUT	DBG_I2C1_SMBUS_TIMEOUT	Res.	Res.	Res.	Res.	Res.
RW	RW								RW	RW					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	Res.	Res.	Res.	Res.	DBG_TIM7_STOP	DBG_TIM6_STOP	Res.	Res.	DBG_TIM3_STOP	DBG_TIM2_STOP
			RW	RW	RW					RW	RW			RW	RW
Bit	Name				R/W	Reset Value				Function					
31	DBG_LPTIM1_STOP				RW	0				Control bit for LPTIM1 counter clock when CPU is halted 0: Enable 1: Disable					
30	DBG_LPTIM2_STOP				RW	0				Control bit for LPTIM2 counter clock when CPU is halted 0: Enable 1: Disable					
29:23	Reserved				-	-				Reserved					
22	DBG_I2C2_SMBUS_TIMEOUT				RW	0				Control bit for I2C2 SMBUS timeout when CPU is halted 0: Behaves the same as in normal mode 1: Freezes SMBUS timeout					
21	DBG_I2C1_SMBUS_TIMEOUT				RW	0				Control bit for I2C1 SMBUS timeout when CPU is halted 0: Behaves the same as in normal mode 1: Freezes SMBUS timeout					
20:13	Reserved				-	-				Reserved					
12	DBG_IWDG_STOP				RW	0				Control bit for IWDG counter clock when CPU is halted 0: Enable 1: Disable					
11	DBG_WWDG_STOP				RW	0				Control bit for WWDG counter clock when CPU is halted 0: Enable 1: Disable					
10	DBG_RTC_STOP				RW	0				Control bit for RTC counter clock when CPU is halted 0: Enable 1: Disable					
9:6	Reserved				-	-				Reserved					
5	DBG_TIM7_STOP				RW	0				Clocking of TIM7 counter when the core is halted 0: Enable 1: Disable					
4	DBG_TIM6_STOP				RW	0				Clocking of TIM6 counter when the core is halted 0: Enable 1: Disable					

3:2	Reserved	-	-	Reserved
1	DBG_TIM3_STOP	RW	0	Clocking of TIM3 counter when the core is halted 0: Enable 1: Disable
0	DBG_TIM2_STOP	RW	0	Clocking of TIM2 counter when the core is halted 0: Enable 1: Disable

36.9.4. DBG APB Peripheral Freeze Register 2 (DBG_APB_FZ2)

This register configures the clocking of timer counters when the MCU is under debug. The register is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset. Offset address: 0x0C

Power on reset value: 0x0000 0000

Only 32-bit access supported. Read-only

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Re s.	Re s.	Re s.	Re s.	Res.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	DBG_PWM_STOP	DBG_TIM17_STOP	DBG_TIM16_STOP	DBG_TIM15_STOP
												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Re s.	Re s.	Re s.	Re s.	DBG_TIM1_STOP	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Re s.	Res.	Res.	Res.	Res.
				RW											

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19	DBG_PWM_STOP	RW	0	Control bit for PWM counter clock when CPU is halted 0: Enable 1: Disable
18	DBG_TIM17_STOP	RW	0	Clocking of TIM17 counter when the core is halted 0: Enable 1: Disable
17	DBG_TIM16_STOP	RW	0	Clocking of TIM16 counter when the core is halted 0: Enable 1: Disable
16	DBG_TIM15_STOP	RW	0	Clocking of TIM15 counter when the core is halted 0: Enable 1: Disable
15:12	Reserved	-	-	Reserved
11	DBG_TIM1_STOP	RW	0	Clocking of TIM1 counter when the core is halted 0: Enable 1: Disable
10:0	Reserved	-	-	Reserved

37. Revision history

Version	Date	Descriptions
V0.3	2025.10.24	1. Initial version 2. Consistent with the Chinese version number



Puya Semiconductor Co., Ltd.

IMPORTANT NOTICE

Puya reserve the right to make changes, corrections, enhancements, modifications to Puya products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information of Puya products before placing orders.

Puya products are sold pursuant to terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice and use of Puya products. Puya does not provide service support and assumes no responsibility when products that are used on its own or designated third party products.

Puya hereby disclaims any license to any intellectual property rights, express or implied.

Resale of Puya products with provisions inconsistent with the information set forth herein shall void any warranty granted by Puya.

Any with Puya or Puya logo are trademarks of Puya. All other product or service names are the property of their respective owners.

The information in this document supersedes and replaces the information in the previous version.

Puya Semiconductor Co., Ltd. – All rights reserved