DESCRIPTION

The PT32C302 microcontrollers is a series of low-power microcontrollers incorporating a high-performance ARM CortexTM-M0 32-bit RISC core. It operates at a maximum 48 MHz frequency and features up to 64 Kbytes of Flash and up to 8 Kbytes of SRAM. A comprehensive set of power-saving modes allows it to be employed in low-power applications. Extensive range of enhanced peripherals and I/Os are available allowing the device to be adopted in different application. A comprehensive set of power-saving modes allows it to be employed in low-power applications. The Flash program memory can be reprogrammed in-system through the SWD interface.

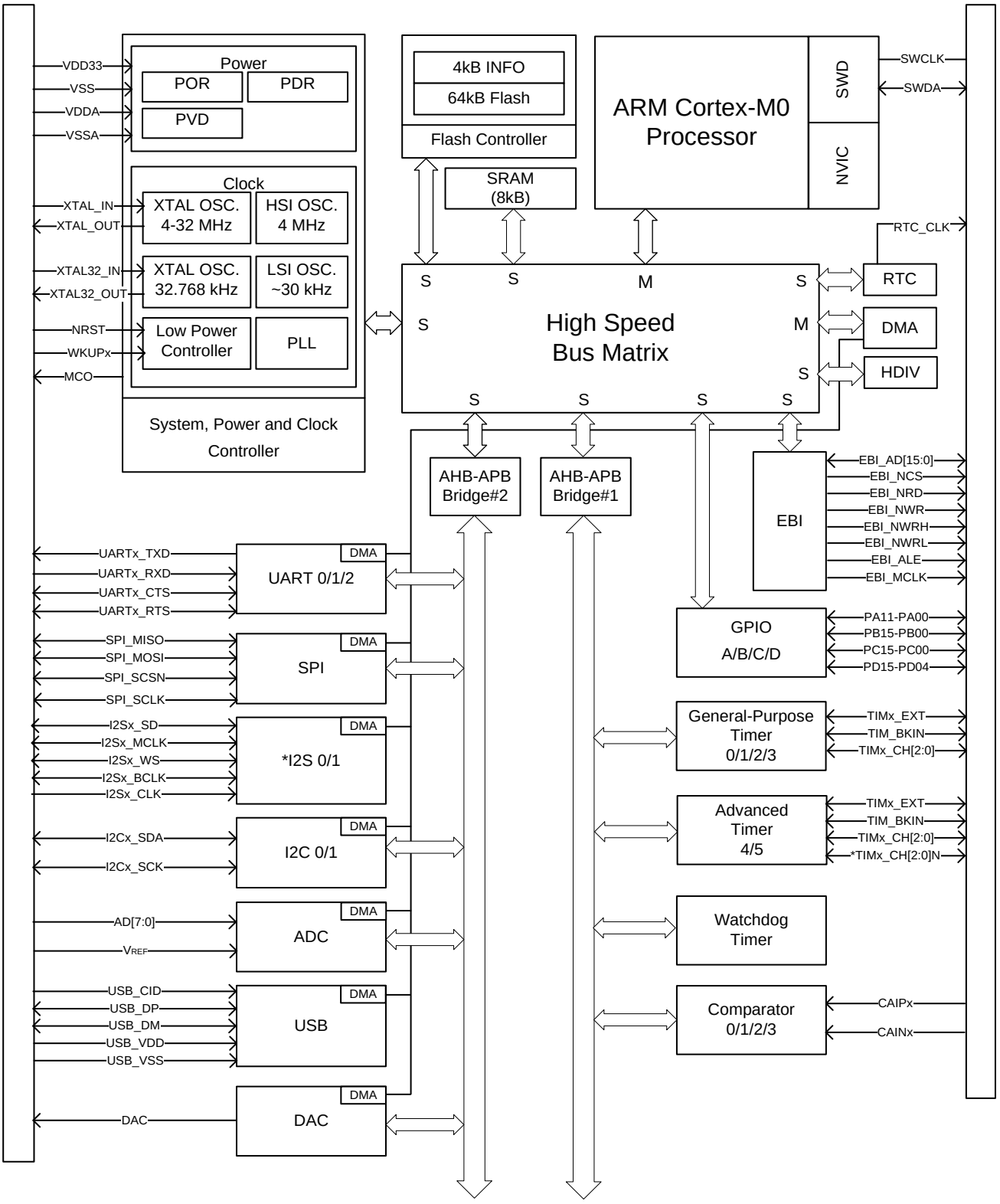
PT32C302 USB MCUs subseries let you easy to implement industry standard USB connectivity. An on-board USB 2.0 full-speed function Host / Device controller with an integrated transceiver and no external crystal or oscillator required for the USB. Elimination of external clock reduces the overall system cost associated with developing a USB device application, make it easy to add USB functionality to the embedded system. In order to support the crystal-less USB device operation, a factory-trimmed internal 4 MHz High Speed clock source is capable of generating the 48MHz clock for USB module and a USB clock recovery circuit is included on this microcontroller.

PT32C302 microcontroller peripheral set also includes an External Bus Interface, 2x UARTs, 2x I2C, 2x I2Ss, 3x SPIs, 4x general-purpose 16-bit timers, 2x advanced 16-bit timers, an RTC, an ADC, a 12-bit DAC and four analog comparator.

FEATURES

- ARM Cortex M0 Processor
- Performance up to 48 MHz
- Flash memory 64K-Byte with 4K-Byte information
- System SRAM 8K-Byte
- Power control
  - POR/PDR
  - PVD
- Low power mode
  - Sleep mode
  - Stop mode
  - Standby mode
- Clock source
  - External crystal 8MHz and 32KHz
  - Intenal trimmable RC 4MHz, and RC32KHz
- Peripherals
  - Up to 56 GPIOs
  - 16-bit general timer * 4
  - 16-bit advance timer *2
  - PWM
  - DMA 6 channel
  - WDT/RTC
  - 8 channel 12-bit ADC
  - 12-bit DAC
  - 4 Comparator with window comparison
  - USB 2.0 full-speed
  - 2 UARTs
  - 2 I^2C with master/slave mode
  - 2 I^2S
  - 3 SPI with variable bit length (4-16 bits)
  - External Bus Interface(EBI)

BLOCK DIAGRAM



CONTENT

DESCRIPTION **1**
FEATURES **1**
BLOCK DIAGRAM **2**
CONTENT **3**
1. ORDER INFORMATION **4**
2. PIN CONFIGURATION **5**
3. PIN DESCRIPTION **8**
3.1 MULTIPLEXING PINS FUNCTION SELECTION 10
3.2 SIGNAL DESCRIPTION 12
4. FUNCTIONAL DESCRIPTION **14**
4.1 SYSTEM AND MEMORY OVERVIEW 14
4.2 ARM ® CORTEX™-M0 CORE 18
4.3 CPU SYSTEM CONTROL 34
4.4 SYSTEM CONTROL (SC) 39
4.5 FLASH CONTROLLER (FC) 83
4.6 GENERAL PURPOSE I/O (GPIO) 87
4.7 UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART) 101
4.8 ANALOG TO DIGITAL CONVERTER (ADC) 129
4.9 DIGITAL TO ANALOG CONVERTER (DAC) 163
4.10 GENERAL PURPOSE TIMERS (GPT) 172
4.11 ADVANCED-CONTROL TIMERS (ADTM) 238
4.12 ANALOG COMPARATOR (AC) 317
4.13 WATCH DOG TIMER (WDT) 328
4.14 REAL TIME CLOCK (RTC) 336
4.15 INTER INTEGRATED CIRCUIT (I2C) 360
4.16 SERIAL PERIPHERAL INTERFACE (SPI) 389
4.17 INTER-IC SOUND (I2S) 426
4.18 EXTERNAL BUS INTERFACE (EBI) 452
4.19 DIRECT MEMORY ACCESS CONTROLLER (DMA) 463
4.20 UNIVERSAL SERIAL BUS (USB) 483
4.21 HARDWARE DIVIDER (HDIV) 541
4.22 APPENDIX A CODE EXAMPLES 545
5. ELECTRICAL CHARACTERISTICS **546**
5.1 MAXIMUM RATINGS 546
5.2 OPERATING CONDITIONS 546
5.3 I/O PIN CHARACTERISTICS 546
5.4 ON-CHIP LOW DROP-OUT (LDO) REGULATOR CHARACTERISTICS 547
5.5 PHASE LOCKED LOOP CHARACTERISTICS 547
5.6 POWER-ON RESET CHARACTERISTICS 548
5.7 NRST CHARACTERISTICS 549
5.8 8 MHZ XTAL CHARACTERISTICS 549
5.9 4 MHZ RCOSC CHARACTERISTICS 550
5.10 32 KHZ XTAL 550
5.11 A/D IO CHARACTERISTICS 551
5.12 TEMPERATURE SENSOR CHARACTERISTICS 551
5.13 ADC+ PGA CHARACTERISTICS 552
5.14 COMPARATOR CHARACTERISTICS 553
5.15 RCOSC_32K CHARACTERISTICS 553
5.16 DAC CHARACTERISTICS 553
5.17 USB PAD CHARACTERISTICS 553
5.18 POWER CONSUMPTION TABLE 554
6. PACKAGE INFORMATION **555**

1. ORDER INFORMATION

Valid Part Number	Package Type	Top Code
PT32C302	LQFP 64	PT32C302
PT32C302	LQFP 48	PT32C302
PT32C302	QFN 24	PT32C302

P T 3 2 C 3 0 2 T G M

Product Line Code
PT : Princeton Technology

MCU Type
08 : 8-bit MCU 8051
32 : 32-bit MCU M0
33 : 32-bit MCU M3
35 : 32-bit MCU M4

Product Type
M : Medical/Motor
C : Connectivity
U : Universal

I/O Type
3 : 3.3V Power/IO
5 : 5V Power/IO

Product Type Sub Family
C : Connectivity
0-1 : USB
3 : CAN/LIN
5 : PLC
6-7 : RF

M : Medical/Motor
0-1 : Medical
2-3 : Motor

Device Variant
2 : Version 2

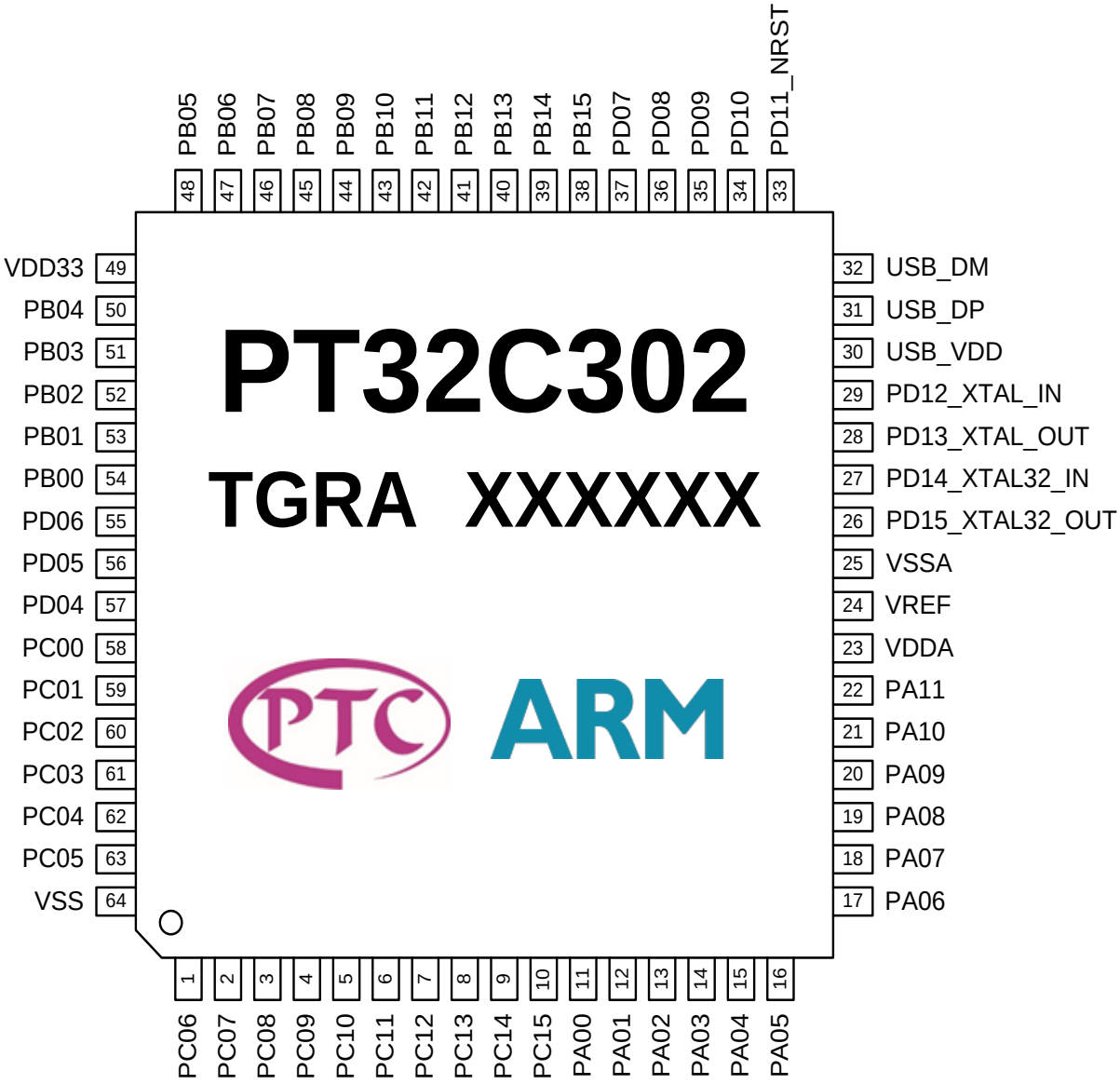
Flash Memory Size
T : 64KB Eflash Memory, 8KB SRAM
H : 32KB Eflash Memory, 4KB SRAM

Package Type
G : LQFP, 7x7 mm^2

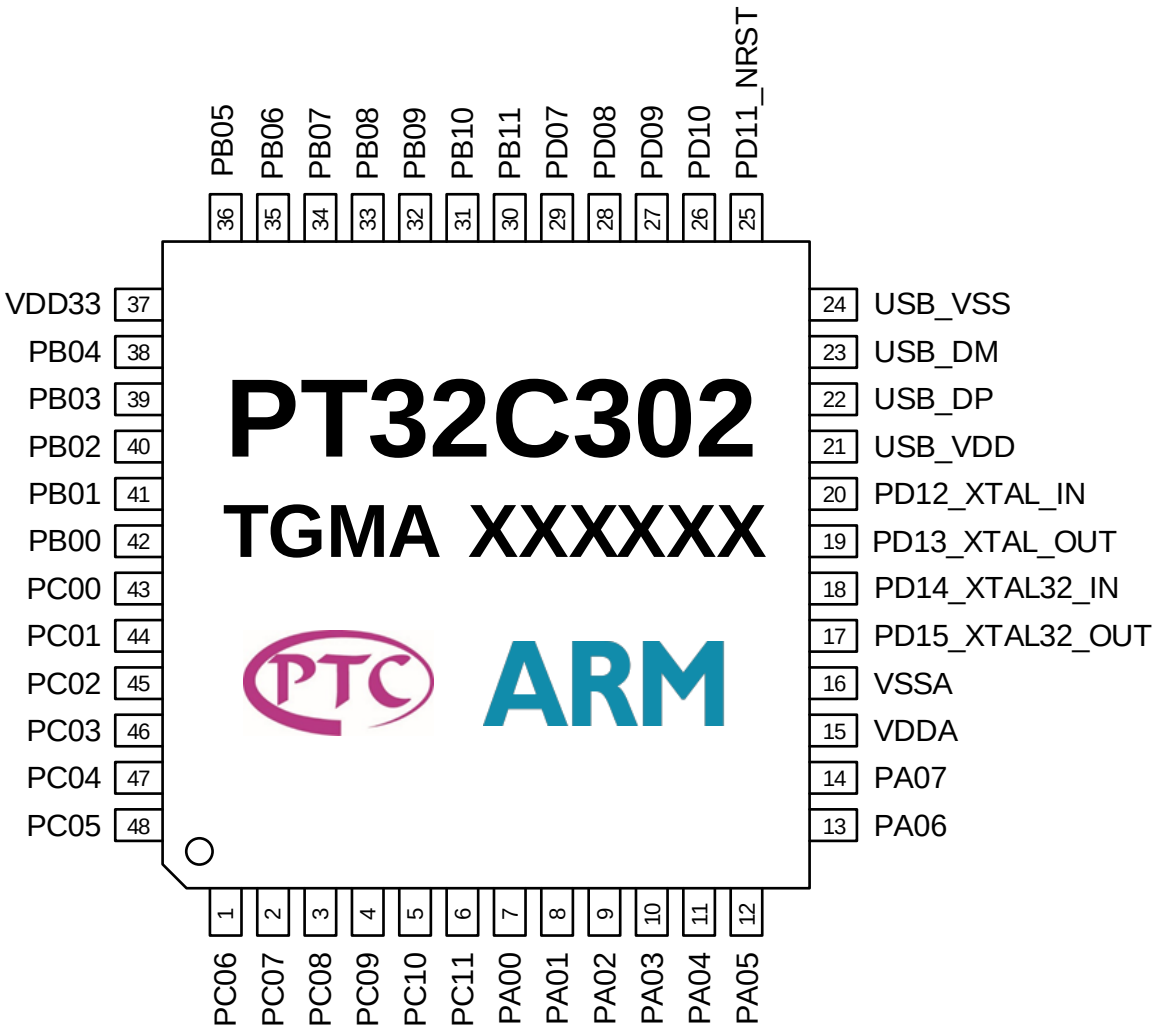
Pin Count
M : 48 pins
R : 64 pins

2. PIN CONFIGURATION

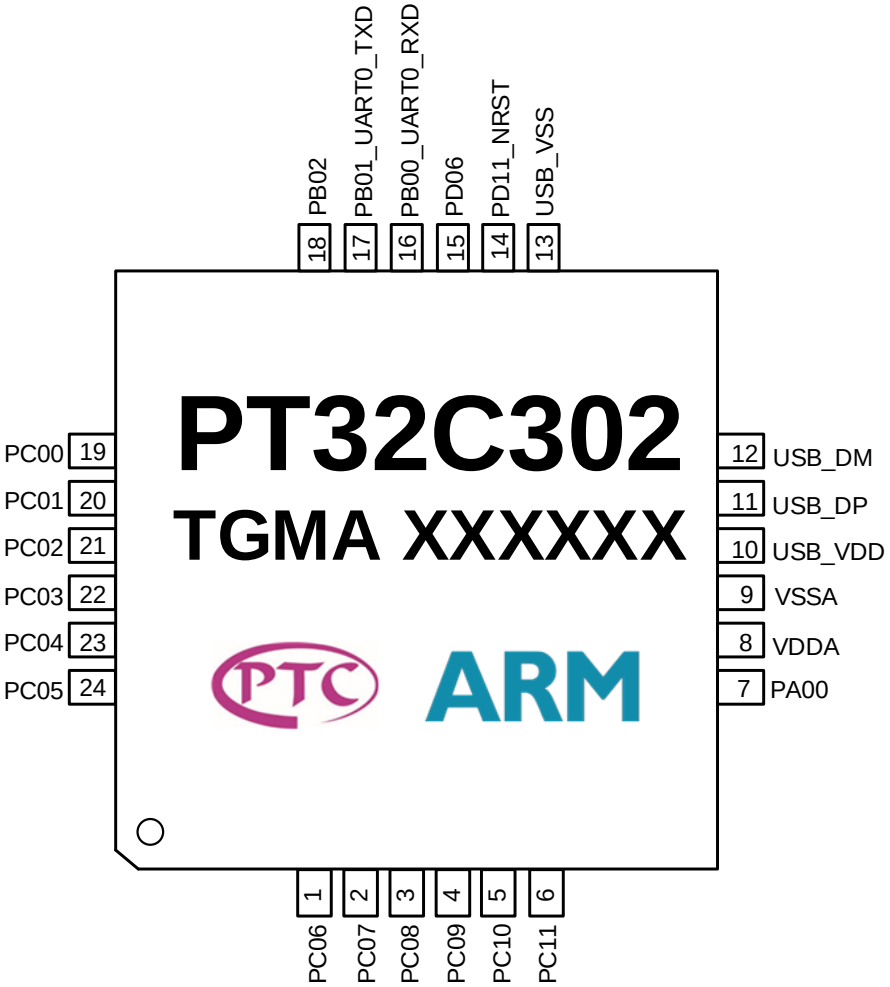
LQFP - 64



LQFP - 48


PB05 36
PB06 35
PB07 34
PB08 33
PB09 32
PB10 31
PB11 30
PD07 29
PD08 28
PD09 27
PD10 26
PD11_NRST 25
VDD33 37
PB04 38
PB03 39
PB02 40
PB01 41
PB00 42
PC00 43
PC01 44
PC02 45
PC03 46
PC04 47
PC05 48
PT32C302
TGMA XXXXXX
ARM
24 USB_VSS
23 USB_DM
22 USB_DP
21 USB_VDD
20 PD12_XTAL_IN
19 PD13_XTAL_OUT
18 PD14_XTAL32_IN
17 PD15_XTAL32_OUT
16 VSSA
15 VDDA
14 PA07
13 PA06
1 PC06
2 PC07
3 PC08
4 PC09
5 PC10
6 PC11
7 PA00
8 PA01
9 PA02
10 PA03
11 PA04
12 PA05

QFN - 24


PB02
PB01_UART0_TXD
PB00_UART0_RXD
PD06
PD11_NRST
USB_VSS
18
17
16
15
14
13
PT32C302
TGMA XXXXXX
PC00 19
PC01 20
PC02 21
PC03 22
PC04 23
PC05 24
12 USB_DM
11 USB_DP
10 USB_VDD
9 VSSA
8 VDDA
7 PA00
1
2
3
4
5
6
PC06
PC07
PC08
PC09
PC10
PC11

3. PIN DESCRIPTION

Each GPIO line can be assigned to one of the peripheral functions. The following table lists out the pin name of all packages and its respective available alternate function.

Pin Name	Pin Type	Description	Pin No.		
			LQFP64	LQFP48	QFN24
PC06	I/O	General Purpose Digital I/O Pin	1	1	1
PC07	I/O	General Purpose Digital I/O Pin	2	2	2
PC08	I/O	General Purpose Digital I/O Pin	3	3	3
PC09	I/O	General Purpose Digital I/O Pin	4	4	4
PC10	I/O	General Purpose Digital I/O Pin	5	2	5
PC11	I/O	General Purpose Digital I/O Pin	6	6	6
PC12	I/O	General Purpose Digital I/O Pin	7		
PC13	I/O	General Purpose Digital I/O Pin	8		
PC14	I/O	General Purpose Digital I/O Pin	9		
PC15	I/O	General Purpose Digital I/O Pin	10		
PA00	I/O	General Purpose Digital I/O Pin	11	7	7
PA01	I/O	General Purpose Digital I/O Pin	12	8	
PA02	I/O	General Purpose Digital I/O Pin	13	9	
PA03	I/O	General Purpose Digital I/O Pin	14	10	
PA04	I/O	General Purpose Digital I/O Pin	15	11	
PA05	I/O	General Purpose Digital I/O Pin	16	12	
PA06	I/O	General Purpose Digital I/O Pin	17	13	
PA07	Supply	Analog Voltage Supply	18	14	
PA08	Ground	Analog Ground	19		
PA09	I/O	General Purpose Digital I/O Pin/Crystal Output	20		
PA10	I/O	General Purpose Digital I/O Pin/Crystal Input	21		
PA11	I/O	General Purpose Digital I/O Pin/Crystal Output	22		
VDDA	I/O	General Purpose Digital I/O Pin/Crystal input	23	15	8
VREF	I/O	General Purpose Digital I/O Pin/Reset	24		
VSSA	I/O	General Purpose Digital I/O Pin	25	16	9
PD15_XTAL32_OUT	I/O	General Purpose Digital I/O Pin	26	17	
PD14_XTAL32_IN	I/O	General Purpose Digital I/O Pin	27	18	
PD13_XTAL_OUT	I/O	General Purpose Digital I/O Pin	28	19	
PD12_XTAL_IN	I/O	General Purpose Digital I/O Pin	29	20	
USB_VDD	I/O	General Purpose Digital I/O Pin	30	21	10
USB_DP	I/O	General Purpose Digital I/O Pin	31	22	11
USB_DM	I/O	General Purpose Digital I/O Pin	32	23	12
USB_VSS	I/O	General Purpose Digital I/O Pin	-	24	13
PD11_NRST	I/O	General Purpose Digital I/O Pin	33	25	14
PD10	I/O	General Purpose Digital I/O Pin	34	26	
PD09	I/O	General Purpose Digital I/O Pin	35	27	
PD08	Supply	Voltage Supply	36	28	
PD07	I/O	General Purpose Digital I/O Pin	37	29	
PB15	I/O	General Purpose Digital I/O Pin	38		

Pin Name	Pin Type	Description	Pin No.		
			LQFP64	LQFP48	QFN24
PB14	I/O	General Purpose Digital I/O Pin	39		
PB13	I/O	General Purpose Digital I/O Pin	40		
PB12	I/O	General Purpose Digital I/O Pin	41		
PB11	I/O	General Purpose Digital I/O Pin	42	30	
PB10	I/O	General Purpose Digital I/O Pin	43	31	
PB09	I/O	General Purpose Digital I/O Pin	44	32	
PB08	I/O	General Purpose Digital I/O Pin	45	33	
PB07	I/O	General Purpose Digital I/O Pin	46	34	
PB06	I/O	General Purpose Digital I/O Pin	47	35	
PB05	I/O	General Purpose Digital I/O Pin	48	36	
VDD33	I/O	General Purpose Digital I/O Pin	49	37	
PB04	I/O	General Purpose Digital I/O Pin	50	38	
PB03	I/O	General Purpose Digital I/O Pin	51	39	
PB02	Supply	Voltage Supply	52	40	18
PB01	I/O	General Purpose Digital I/O Pin	53	41	17
PB00	I/O	General Purpose Digital I/O Pin	54	42	16
PD06	I/O	General Purpose Digital I/O Pin	55		15
PD05	I/O	General Purpose Digital I/O Pin	56		
PD04	I/O	General Purpose Digital I/O Pin	57		
PC00	I/O	General Purpose Digital I/O Pin	58	43	19
PC01	I/O	General Purpose Digital I/O Pin	59	44	20
PC02	I/O	General Purpose Digital I/O Pin	60	45	21
PC03	I/O	General Purpose Digital I/O Pin	61	46	22
PC04	I/O	General Purpose Digital I/O Pin	62	47	23
PC05	I/O	General Purpose Digital I/O Pin	63	48	24
VSS	I/O	General Purpose Digital I/O Pin	64		

3.1 MULTIPLEXING PINS FUNCTION SELECTION

The following tables describes the peripheral signals multiplexed to the GPIO lines. Alternate Function (AF) is enabled by configuring the **GPIOx_AFRL** and **GPIOx_AFRH** registers.

Note (1):

All pins are in AF0 mode by default, with the exception of following cases:

(1) Crystal Oscillator Pinout: Respective pins PD [15:11] are defaulted to the AF8 functionality.

(2) Serial Wire Debug Interface Pinout: Respective PB [6:5] are defaulted to the AF4 functionality

Note (2): expert

(3) The I2S share pins with SPI, those pins can be configured as I2S or SPI protocol depends on user's setting.

Table 3.1-1: Multiplexing Pins Function Selection

Pin Name	Alternate Functions								
	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	AF8
PC06	PC06	UART0_TXD	SPI0_SCLK	SPIS1_SCLK	EBI_AD6	T0PWM0_CCP0	T5PWM0P_CCP0	ADC_TSDA[6]	-
PC07	PC07	UART0_RXD	SPI0_SCSN	SPIS1_SCSN	EBI_AD7	T0PWM1_CCP1	T5PWM0N	ADC_TSDA[7]	-
PC08	PC08	UART0_CTS	SPI0_MOSI	SPIS1_MOSI	EBI_NCS	T0PWM2_CCP2	T5PWM1P_CCP1	ADC_TSDA[8]	-
PC09	PC09	UART0_RTS	SPI0_MISO	SPIS1_MISO	EBI_ALE	T1PWM0_CCP0	T5PWM1N	ADC_TSDA[9]	-
PC10	PC10	UART1_CTS	-	I2C0_SCK	EBI_NRD	T1PWM1_CCP1	T5PWM2P_CCP2	ADC_TSDA[10]	-
PC11	PC11	UART1_RTS	-	I2C0_SDA	EBI_NWR	T1PWM2_CCP2	T5PWM2N	ADC_TSDA[11]	-
PC12	PC12	UART2_CTS	SPIS1_SCLK	-	EBI_AD8	T3PWM0_CCP0	-	-	-
PC13	PC13	UART2_RTS	SPIS1_SCSN	-	EBI_AD9	T3PWM1_CCP1	-	-	-
PC14	PC14	UART1_CTS	SPIS1_MOSI	-	EBI_AD10	T3PWM2_CCP2	-	-	-
PC15	PC15	UART1_RTS	SPIS1_MISO	-	-	TMBREAK	-	-	DAC
PA00	PA00	-	-	-	-	T4PWM0P_CCP0	T5PWM0P_CCP0	-	AD0_CAIP0
PA01	PA01	-	-	-	-	T4PWM1P_CCP1	T5PWM1P_CCP1	-	AD1_CAIN0
PA02	PA02	-	-	-	-	T4PWM2P_CCP2	T5PWM2P_CCP2	-	AD2_CAIP1
PA03	PA03	-	-	-	-	T4PWM0N	T5PWM0N	-	AD3_CAIN1
PA04	PA04	-	SPIS0_SCLK	-	-	T4PWM1N	T5PWM1N	-	AD4
PA05	PA05	-	SPIS0_SCSN	-	-	T4PWM2N	T5PWM2N	-	AD5
PA06	PA06	-	SPIS0_MOSI	-	-	T5PWM0P_CCP0	T4PWM0P_CCP0	-	AD6
PA07	PA07	-	SPIS0_MISO	-	-	T5PWM1P_CCP1	T4PWM1P_CCP1	-	AD7
PA08	PA08	-	-	-	-	T5PWM2P_CCP2	T4PWM2P_CCP2	-	CAIP2
PA09	PA09	-	-	-	-	T5PWM0N	T4PWM0N	-	CAIN2
PA10	PA10	-	-	-	-	T5PWM1N	T4PWM1N	-	CAIP3
PA11	PA11	-	-	-	-	T5PWM2N	T4PWM2N	-	CAIN3
VDDA									
VSSA									
PD15 XTAL32_OUT	PD15	-	-	-	-	-	-	-	XTAL32_OUT
PD14_ XTAL32_IN	PD14	-	-	-	-	-	-	-	XTAL32_IN
PD13_ XTAL_OUT	PD13	-	-	-	-	-	-	-	XTAL_OUT
PD12_ XTAL_IN	PD12	-	-	-	-	-	-	-	XTAL_IN
USB_VDD									
USB_DP									

Pin Name	Alternate Functions								
	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	AF8
USB_DM									
USB_VSS									
PD11_ NRST	PD11	-	-	-	-	-	-	-	NRST
PD10_WKUP	PD10	-	-	-	-	-	-	-	WKUP_V33
PD09	PD09	UART2_TXD	SPI0_SCSN		EBI_NWRH	T1PWM2_CCP2	T0EXTR	MUX_CLK_OUT	WKUP_PD09
PD08	PD08	UART2_RXD	SPI0_SCLK		EBI_NWRL	T1PWM1_CCP1	T1EXTR	RTC_1HZ	WKUP_PD08
PD07	PD07	UART0_RTS	SPIS1_SCLK		EBI_MCLK	T1PWM0_CCP0	T2EXTR	ADC_TSVALID	WKUP_PD07
PB15	PB15	-	-	-	EBI_AD15	TMBREAK	T3EXTR	-	-
PB14	PB14	-	-	-	EBI_AD14	T3PWM2_CCP2	T4EXTR	-	-
PB13	PB13	-	-	-	EBI_AD13	T3PWM1_CCP1	T5EXTR	-	-
PB12	PB12	-	-	-	EBI_AD12	T3PWM0_CCP0	TMBREAK	-	-
PB11	PB11	UART1_TXD	SPIS1_SCSN	I2C1_SCK	-	T3EXTR	TMBREAK	T4EXTR	-
PB10	PB10	UART1_RXD	SPIS1_MOSI	I2C1_SDA	-	T3PWM2_CCP2	T5PWM2N	T4PWM2N	-
PB09	PB09	UART0_CTS	SPIS1_MISO	-	-	T3PWM1_CCP1	T5PWM1N	T4PWM2P_CCP2	WKUP_PB09
PB08	PB08	UART2_TXD	SPI0_MISO	I2C0_SCK	-	T3PWM0_CCP0	T5PWM0N	T4PWM1N	-
PB07	PB07	UART2_RXD	SPI0_MOSI	I2C0_SDA	-	T2PWM2_CCP2	T5PWM2P_CCP2	T4PWM1P_CCP1	-
PB06	PB06	UART1_CTS	SPI0_SCSN	-	SWCLK	T2PWM1_CCP1	T5PWM1P_CCP1	T4PWM0N	-
PB05	PB05	UART1_RTS	SPI0_SCLK	-	SWDA	T2PWM0_CCP0	T5PWM0P_CCP0	T4PWM0P_CCP0	-
VDD									
PB04	PB04	UART1_TXD	SPI0_MISO	-	-	TIM5_CH0N	T4EXTR	-	-
PB03	PB03	UART1_RXD	SPI0_MOSI	-	-	TIM5_CH0	T5EXTR	-	-
PB02	PB02	-	I2S1_CLK	I2S0_CLK	-	USB_CID	TMBREAK	T5EXTR	WKUP_PB02
PB01	PB01	UART0_TXD	-	-	-	T0EXTR	T2EXTR	-	-
PB00	PB00	UART0_RXD	-	-	-	T1EXTR	T3EXTR	-	-
PD06	PD06	-	I2S1_CLK	I2S0_CLK	EBI_AD11	T2PWM0_CCP0	USB_CID	-	WKUP_PD06
PD05	PD05	-	-	-	-	T2PWM1_CCP1	-	-	WKUP_PD05
PD04	PD04	-	-	-	-	T2PWM2_CCP2	-	-	WKUP_PD04
PC00	PC00	UART2_TXD	I2C1_SCK	I2C0_SCK	EBI_AD0	T0PWM0_CCP0	T4PWM0P_CCP0	ADC_TSDA[0]	-
PC01	PC01	UART2_RXD	I2C1_SDA	I2C0_SDA	EBI_AD1	T0PWM1_CCP1	T4PWM0N	ADC_TSDA[1]	-
PC02	PC02	UART1_TXD	SPI0_MISO	SPIS0_SCLK	EBI_AD2	T0PWM2_CCP2	T4PWM1P_CCP1	ADC_TSDA[2]	-
PC03	PC03	UART1_RXD	SPI0_MOSI	SPIS0_SCSN	EBI_AD3	T1PWM0_CCP0	T4PWM1N	ADC_TSDA[3]	-
PC04	PC04	UART1_CTS	SPI0_SCSN	SPIS0_MOSI	EBI_AD4	T1PWM1_CCP1	T4PWM2P_CCP2	ADC_TSDA[4]	-
PC05	PC05	UART1_RTS	SPI0_SCLK	SPIS0_MISO	EBI_AD5	T1PWM2_CCP2	T4PWM2N	ADC_TSDA[5]	-

3.2 SIGNAL DESCRIPTION

The following table describes the details on signals names classified by peripheral.

Table 3.2-1 : Alternate Function Description

Signal Name	I/O	Function Description
Power Supplies		
VDDA	Supply	Analog Domain 3.3 V Power Supply
VREF	Supply	Analog Reference
VDD33	Supply	Digital Domain 3.3V Power Supply
USB_VDD	Supply	USB Power Supply Voltage
VSSA	Ground	Ground
VSS	Ground	Ground Voltage
USB_VSS	Ground	USB Ground
Universal Asynchronous Receiver/Transmitter (UART0, UART1, UART2), x = 0, 1, 2		
UARTx_TXD	O	UART x Data output pins
UARTx_RXD	I	UART x Data Input pins
UARTx_CTSn	I/O	UART x Clear to Send pins
UARTx_RTSn	I/O	UART x Request to Send pins
Serial Wire Debug (SWD)		
SWCLK	I	SWD Clock
SWDA	I/O	SWD Data Input/Output
Inter Integrated Circuit (I2C0, I2C1), x= 0, 1		
I2Cx_SDA	I/O	I2C Data
I2Cx_SCK	I/O	I2C Clock
Universal Serial Bus (USB)		
USB_CID	I	USB Connect ID
USB_DP	I/O	USB Full Speed Data +
USB_DM	I/O	USB Full Speed Data -
Serial Peripheral Interface (SPI)		
SPI_MISO	I/O	SPI Master Input Slave Output
SPI_MOSI	I/O	SPI Master Output Slave Input
SPI_SCSN	I/O	SPI Chip Select
SPI_SCLK	I/O	SPI Clock
Inter-IC Sound (I2S0, I2S1) x = 0, 1		
I2Sx_SD	I/O	I2S Receive/ Transmit Data
I2Sx_MCLK	O	I2S Master Clock
I2Sx_WS	I/O	I2S Word Select
I2Sx_BCLK	I/O	I2S Bit Clock
I2Sx_CLK	I	I2S Clock Source
General Purpose Input/Output (GPIOA, GPIOB, GPIOC, GPIOD)		
PA11-PA00	I/O	GPIO Port A
PB11-PB00	I/O	GPIO Port B
PC11-PC00	I/O	GPIO Port C
PD15-PD04	I/O	GPIO Port D

Signal Name	I/O	Function Description
General-Purpose Timer (TIM0, TIM1, TIM2, TIM3), x = 0, 1, 2, 3		
TIMx_CH0	I/O	Timer X input/output pin 0
TIMx_CH1	I/O	Timer X input/output pin 1
TIMx_CH2	I/O	Timer X input/output pin 2
TIMx_ETR	I	Timer X External Trigger Input
Advanced-control Timer (TIM4, TIM5), x=4,5		
TIMx_CH0	I/O	Timer X input/output pin 0/ Timer X main output 0
TIMx_CH0N	O	Timer X complementary output 0
TIMx_CH1	I/O	Timer X input/output pin 1/ Timer X main output 1
TIMx_CH1N	O	Timer X complementary output 1
TIMx_CH2	I/O	Timer X input/output pin 2/ Timer X main output 2
TIMx_CH2N	O	Timer X complementary output 2
TIM_BKIN	I	Timer break input
TIMx_ETR	I	Timer X External Trigger Inp
Analog to Digital Converter (ADC)		
AD[7:0]	I	ADC Single End Channel Input / *ADC Differential Channel Input Positive or Negative Input
Analog Comparator (AC0, AC1, AC2, AC3), x=0, 1, 2, 3		
CAIPx	I	Comparator x Positive Input
CAINx	I	Comparator x Negative Input
Digital Analog Converter (DAC)		
DAC	O	DAC Output Channel
External Bus Interface (EBI)		
EBI_AD[15:0]	I/O	Address/Data Bus
EBI_NCS	O	Chip Select
EBI_NRD	O	Read Signal
EBI_NWR	O	Write enable
EBI_NWRH	O	High byte write enable
EBI_NWRL	O	Low byte write enable
EBI_ALE	O	Address latch enable
EBI_MCLK	O	External bus base clock
System Control (SC)		
XTAL32_IN	I	32.768K RTC Clock Input
XTAL32_OUT	O	32.768K RTC Clock Output
XTAL_IN	I	High Speed 8MHZ Crystal Clock Input
XTAL_OUT	O	High Speed 8MHZ Crystal Clock Output
NRST	I	System Reset
WKUPx	I	Wakeup Pin
MCO	O	Microcontroller Clock Output
RTC_1Hz	O	RTC 1Hz Clock Pulse Output

4. FUNCTIONAL DESCRIPTION

4.1 SYSTEM AND MEMORY OVERVIEW

The PT32C302 microcontrollers is a series of low-power microcontrollers incorporating a high-performance ARM CortexTM-M0 32-bit RISC core. It operates at a maximum 48 MHz frequency and features up to 64 Kbytes of Flash and up to 8 Kbytes of SRAM. A comprehensive set of power-saving modes allows it to be employed in low-power applications. Extensive range of enhanced peripherals and I/Os are available allowing the device to be adopted in different application. A comprehensive set of power-saving modes allows it to be employed in low-power applications. The Flash program memory can be reprogrammed in-system through the SWD interface.

PT32C302 USB MCUs subseries let you easy to implement industry standard USB connectivity. An on-board USB 2.0 full-speed function Host / Device controller with an integrated transceiver and no external crystal or oscillator required for the USB. Elimination of external clock reduces the overall system cost associated with developing a USB device application, make it easy to add USB functionality to the embedded system. In order to support the crystal-less USB device operation, a factory-trimmed internal 4 MHz High Speed clock source is capable of generating the 48MHz clock for USB module and a USB clock recovery circuit is included on this microcontroller.

PT32C302 microcontroller peripheral set also includes an External Bus Interface, 2x UARTs, 2x I2C, 2x I2Ss, 3x SPIs, 4x general-purpose 16-bit timers, 2x advanced 16-bit timers, an RTC, an ADC, a 12-bit DAC and four analog comparators.

This chapter introduces you to PT32C302 features, its system and memory structure:

- PT32C302 System Architecture
- PT32C302 System Features
- PT32C302 Memory Mapping

4.1.1 PT32C302 MEMORY MAPPING

The system, bus is implemented as a bus matrix. All system bus addresses are fixed and cannot be remapped.

Figure 4.1-1: Memory Mapping

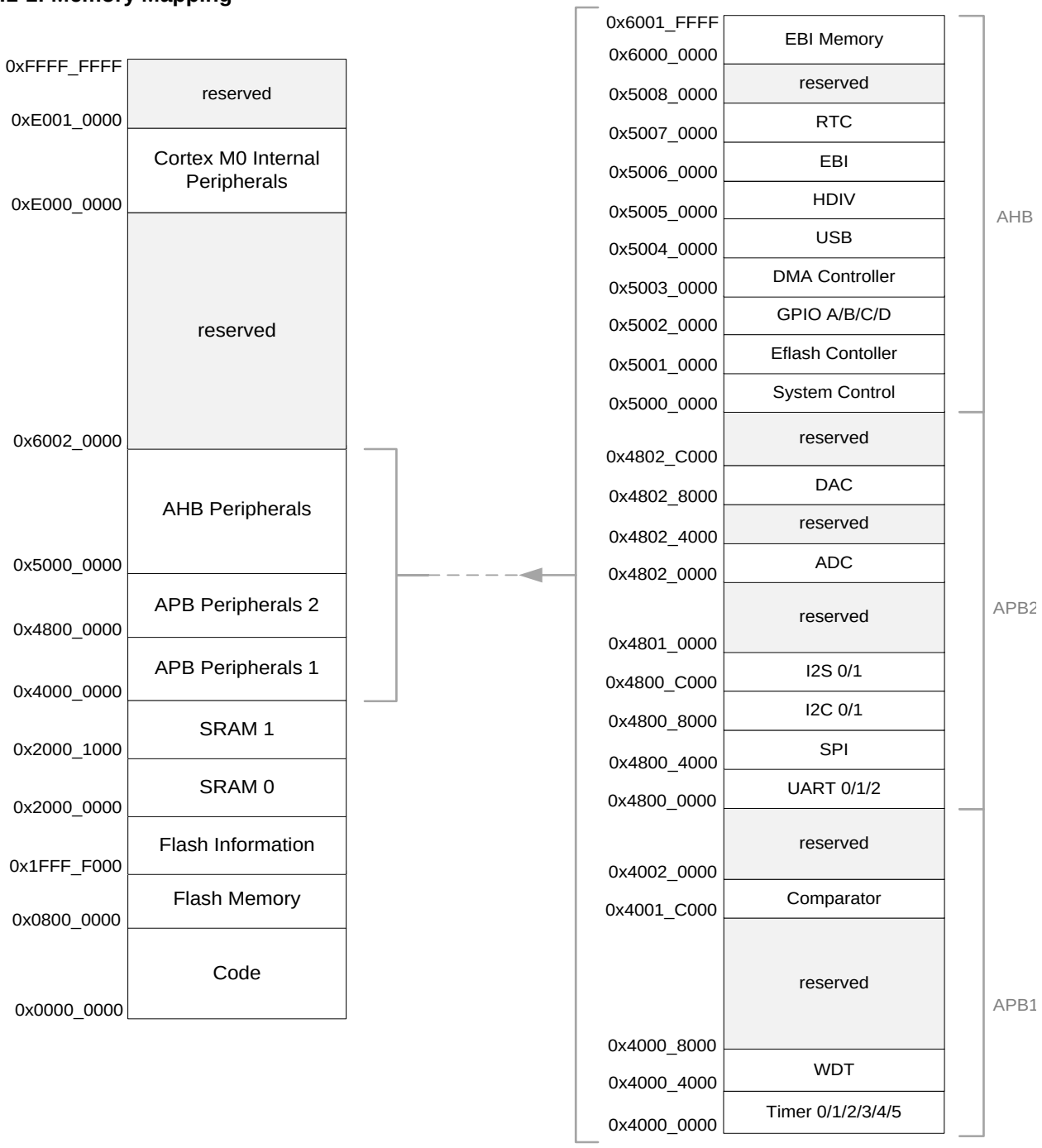


Table 4.1-1: Peripheral register boundary addresses

Boundary address		Depth (Byte)	Peripheral Description	Reference Section	BUS
Start	End				
0x0000_0000	0x1FFF_FFFF	6K	Mask ROM	-	AHB
0x0800_0000	0x0800_FFFF	64K	Embedded Flash Memory Field	4.5	
0x1FFF_F000	0x1FFF_FBFF	3K	Embedded Flash Information Memory Field	4.5	
0x1FFF_FC00	0x1FFF_FFFF	1K	Embedded Flash Information Memory Field	4.5	
0x2000_0000	0x2000_0FFF	4K	System RAM 0	-	
0x2000_1000	0x2000_1FFF	4K	System RAM 1	-	
0x2000_2000	0x3FFF_FFFF	-	Reserved	-	
0x4000_0000	0x4000_3FFF		Timer 0 / 1 / 2 / 3 / 4 / 5 Control Register	4.10	APB1
0x4000_4000	0x4000_7FFF		Watchdog Control Register	4.13	
0x4000_8000	0x4001_BFFF	-	Reserved	-	
0x4001_C000	0x4001_FFFF		Analog Comparator(AC) Control Register	4.12	
0x4002_0000	0x47FF_FFFF	-	Reserved	-	
0x4800_0000	0x4800_3FFF		UART 0/1/2 Control Register	4.7	APB2
0x4800_4000	0x4800_7FFF		SPI Control Register	4.16	
0x4800_8000	0x4800_BFFF		I2C 0/1 Control Register	4.15	
0x4800_C000	0x4801_FFFF	-	I2S 0/1 Control Register	4.17	
0x4801_0000	0x4801_FFFF		Reserved	-	
0x4802_0000	0x4802_3FFF		ADC Control Register	4.8	
0x4802_4000	0x4802_7FFF	-	Reserved	-	
0x4802_8000	0x4802_BFFF		DAC Control Register	4.9	
0x4802_C000	0x4FFF_FFFF		Reserved	-	
0x5000_0000	0x5000_FFFF		System Control Register	4.4	AHB
0x5001_0000	0x5001_FFFF		Embedded Flash Control Register	4.5	
0x5002_0000	0x5002_FFFF		GPIO A/B/C/D Control Register	4.6	
0x5003_0000	0x5003_FFFF		DMA Control Register	4.19	
0x5004_0000	0x5004_FFFF		USB Control Register	4.18	
0x5005_0000	0x5005_FFFF	-	Reserved	-	
0x5006_0000	0x5006_FFFF		EBI Control Register	4.18	
0x5007_0000	0x5007_FFFF		RTC Control Register	4.14	
0x5008_0000	0x5FFF_FFFF		Reserved	-	
0x6000_0000	0x6001_FFFF		External Bus Interface (EBI) Memory Field	-	

Boundary address		Depth (Byte)	Peripheral Description	Reference Section	BUS
Start	End				
0x6002_0000	0xDFFF_FFFF		Reserved	-	
0xE000_0000	0xE00F_FFFF		ARM ® Cortex™-M0 System Timer (SysTick) Control Register	4.2.1	
0xE001_0000	0xFFFF_FFFF	-	Reserved	-	

4.2 ARM ® CORTEX™-M0 CORE

The ARM Cortex™-M0 processor is the smallest and most energy- efficient ARM processor available. It satisfies the demand for ever-lower-cost applications with increasing connectivity. The M0 processor is a configurable, multistage, 32-bit RISC processor.

In PT32C302, this processor configures following features:

- Built-in Nested Vectored Interrupt Controller (NVIC): 32 external Interrupt
- Little-endian
- Integrated system timer - SysTick
- Halting debug support
- Fast multiplier
- Support Serial Wire Debug (SWD) connections.

This chapter provide basic information of the following processor peripherals,

- CPU System Timer Control (SysTick)
- CPU Nested Vectored Interrupt Controller (NVIC)
- CPU System Control

For further information, please refer to:

- ARM Cortex™-M0 Technical Reference Manual
- ARM v6-M Architecture Reference Manual

4.2.1 CPU SYSTEM TIMER CONTROL REGISTER (SYST)

The Cortex™-M0 includes an integrated system timer - **SysTick**, providing a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used as a Real Time Operating System (RTOS) tick timer or as a simple counter.
When the system timer is enabled, it starts counting down from the value in the SysTick Current Value Register (**SYST_CVR**) to 0, and reload (wrap) to the value in the SysTick Reload Value Register (**SYST_RVR**) in the next clock cycle, then decrements on subsequent clocks. Once the counter transitions to 0, the **COUNTFLAG** status bit is set. The **COUNTFLAG** bit clears on reads.

The **SYST_CVR** value is UNKNOWN at reset. Before enabling this feature. Software should write to the register to clear it to zero before enabling the feature. This ensures the timer will count from the **SYST_RVR** value rather than an arbitrary value when it is enabled.

If the **SYST_RVR** is 0, the timer will be maintained with a current value of 0 after it is reloaded with this value. This mechanism can be used to disable the feature independently from the timer enable bit.

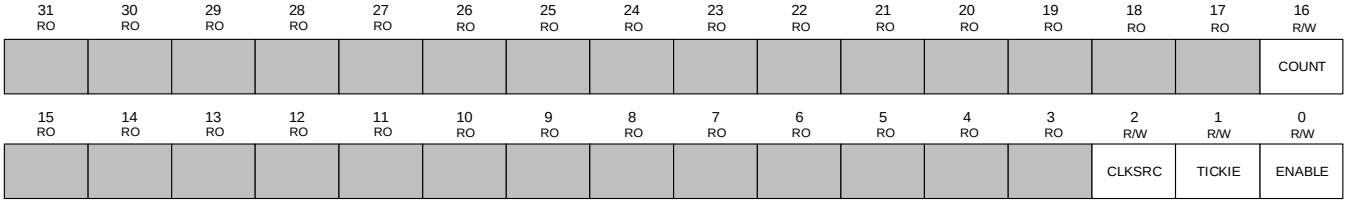
4.2.2 SYST REGISTER MAPS

Base Address: 0xE000_E000

Offset	Symbol	Type	Reset Value	Description	See page
0x0010	SYST_CSR	R/W	0x0000_0000	SysTick Control and Status Register	19
0x0014	SYST_RVR	R/W	-	SysTick Reload Value Register	20
0x0018	SYST_CVR	R/W	-	SysTick Current Value Register	20

4.2.2.1 SYST_CSR - SYSTICK CONTROL AND STATUS REGISTER

The SYST_CSR enables the SysTick features.

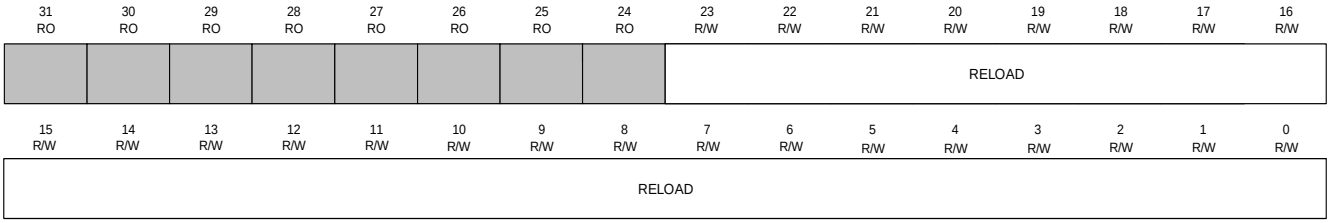


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:17	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	COUNT	R/W	0	Count Flag 0: The SysTick timer has not counted to 0 since the last time this bit was read. 1: The SysTick timer has counted to 0 since the last time this bit was read COUNT is cleared on read or by a write to the Current Value register.
15:3	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	CLKSRC	R/W	0	System Tick Clock Source Selection 0: Clock source is (optional) external reference clock. 1: Core clock used for SysTick.
1	TICKIE	R/W	0	System Tick Interrupt Enable 0: Counting down to 0 does not cause the SysTick exception to be pended. Software can use COUNTFLAG to determine if a count to 0 has occurred. 1: Counting down to 0 will cause the SysTick exception to be pended. Clearing the SysTick Current Value register by a write in software will not cause SysTick to be pended.
0	ENABLE	R/W	0	System Tick Counter Enabled 0: Counter is disabled. 1: Counter operates in a multi-shot manner.

4.2.2.2 SYST_RVR - SYSTICK RELOAD VALUE REGISTER

The SYST_RVR specifies the start value to load into the SYST_CVR.

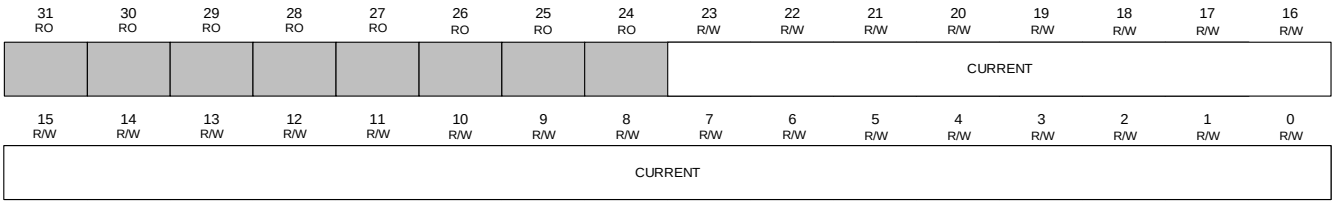


Offset: 0x0014

Bits	Name	Type	Reset	Description
31:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:0	RELOAD	R/W	R0	Reload Value Value to load into the SysTick Current Value Register (SYST_CVR) register when the counter reaches 0.

4.2.2.3 SYST_CVR - SYSTICK CURRENT VALUE REGISTER

The SYST_CVR contains the current value of the SysTick counter.



Offset: 0x0018

Bits	Name	Type	Reset	Description
31:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:0	CURRENT	R/W	0	System Tick Current Value Current counter value. This is the value of the counter at the time it is sampled. The counter does not provide read-modify-write protection. The register is write-clear. A software write of any value will clear the register to 0.

4.2.3 CPU NESTED VECTORED INTERRUPT CONTROLLER (NVIC)

The Cortex™-M0 provides an interrupt controller as an integral part of the exception mode, named as “Nested Vectored Interrupt Controller (NVIC)”, which is closely coupled with the processor core and provides following features:

- Support Nested and Vectored interrupt
- Automatic processor state saving and restoration
- Reduced and deterministic interrupt latency
- 32 maskable interrupts
- 4 levels of priority

The NVIC prioritizes and handles all supported exceptions. All exceptions are handled in “Handler Mode”. This NVIC architecture supports up to 32 discrete interrupts request (IRQ [31:0]) with up to 4 levels of priority. All of the interrupts and most of the system exceptions can be configured to different priority levels. When an interrupt occurs, the NVIC will compare the priority of the new interrupt to the current running one. If the priority of the new interrupt is higher than the current one, the new interrupt handler will override the current handler.

When an interrupt is accepted, the starting address of the interrupt service routine (ISR) is fetched from a vector table in memory. There is no need to determine which interrupt is accepted and branch to the starting address of the correlated ISR by software. While the starting address is fetched, NVIC will also automatically save processor state including the registers “PC, PSR, LR, R0~R3, R12” to the stack. At the end of the ISR, the NVIC will restore the mentioned registers from stack and resume the normal execution. Thus it will take less and deterministic time to process the interrupt request.

The NVIC supports “Tail Chaining” which handles back-to-back interrupts efficiently without the overhead of states saving and restoration and therefore reduces delay time in switching to pending ISR at the end of current ISR. The NVIC also supports “Late Arrival” which improves the efficiency of concurrent ISRs. When a higher priority interrupt request occurs before the current ISR starts to execute (at the stage of state saving and starting address fetching), the NVIC will give priority to the higher one without delay penalty. Thus it advances the real-time capability.

Exceptions Modes and System Interrupt Map

The following table lists the exception models. Software can set four levels of priority on some of these exceptions as well as on all interrupts. The highest user-configurable priority is denoted as “0” and the lowest priority is denoted as “3”. The default priority of all the user-configurable interrupts is “0”. Note that priority “0” is treated as the fourth priority on the system, after three system exceptions “Reset”, “NMI” and “Hard Fault”.

Table 4.2-1: Exception Model

Vector NO.	Exception Name	Priority
1	Reset	-3
2	NMI	-2
3	Hard Fault	-1
4 - 10	Reserved	Reserved
11	SCCall	Configurable
12 - 13	Reserved	Reserved
14	PendSV	Configurable
15	Sys Tick	Configurable
16 - 47	Interrupt(IRQ[31:0])	Configurable

Table 4.2-2: System Interrupt Map

IRQ NO.	Name	Description
31	WAKEUP	CPU Wake Up Interrupt
30	PVD	Power Voltage Detector Interrupt
29-28	Reserved	-
27	USB	USB Interrupt
26	DMA	DMA Interrupt
25	SPI2/I2S1	SPI2/I2S1 Interrupt
24	SPI1/I2S0	SPI1/I2S0 Interrupt
23	TIMER5	TIMER5 Interrupt
22	TIMER4	TIMER4 Interrupt
21	UART 2	UART2 Interrupt
20	I2C1	I2C1 Interrupt
19	Reserved	-
18	ADC	ADC Interrupt
17	COMP	Analog Comparator Interrupt
16-14	Reserved	-
13	TIMER3	TIMER3 Interrupt
12	TIMER2	TIMER2 Interrupt
11	TIMER1	TIMER1 Interrupt
10	TIMER0	TIMER0 Interrupt
9	WDT	Watch Dog Timer Interrupt
8	RTC	RTC Interrupt
7	I2C0	I2C Interrupt
6	SPI0	SPI Interrupt
5	UART1	UART1 Interrupt
4	UART0	UART0 Interrupt
3	GPIO_D	GPIO D Port Interrupt
2	GPIO_C	GPIO C Port Interrupt
1	GPIO_B	GPIO B Port Interrupt
0	GPIO_A	GPIO A Port Interrupt

4.2.4 VECTOR TABLE

When an interrupt is accepted, the processor will automatically fetch the starting address of the interrupt service routine (ISR) from a vector table in memory. For ARMv6-M, the vector table base address is fixed at 0x00000000. The vector table contains the initialization value for the stack pointer on reset, and the entry point addresses for all exception handlers. The vector number on previous page defines the order of entries in the vector table associated with exception handler entry as illustrated in previous section.

Table 4.2-3: Vector Table Format

Vector Table Word Offset	Description
0	SP_main – The Main stack pointer
Vector Number	Exception Entry Pointer using that Vector Number

4.2.5 OPERATION DESCRIPTION

NVIC interrupts can be enabled or disabled by writing to their corresponding Interrupt Set-Enable or Interrupt Clear-Enable register bit-field. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current enabled state of the corresponding interrupts. When an interrupt is disabled, interrupt assertion will cause the interrupt to become Pending, however, the interrupt will not activate. If an interrupt is Active when it is disabled, it remains in its Active state until cleared by reset or an exception return. Clearing the enable bit prevents new activations of the associated interrupt.

NVIC interrupts can be pended/un-pended using a complementary pair of registers to those used to enable/disable the interrupts, named the Set-Pending Register and Clear-Pending Register respectively. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current pended state of the corresponding interrupts. The Clear-Pending Register has no effect on the execution status of an Active interrupt.

NVIC interrupts are prioritized by updating an 8-bit field within a 32-bit register (each register supporting four interrupts). The general registers associated with the NVIC are all accessible from a block of memory in the System Control Space and will be described in next section.

4.2.6 NVIC REGISTER MAPS

Base Address 0xE000_E100

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	NVIC_ISER	R/W	0x0000_0000	IRQ Set Enable Control Register	24
0x0080	NVIC_ICER	R/W	0x0000_0000	IRQ Clear Enable Control Register	24
0x0100	NVIC_ISPR	R/W	0x0000_0000	IRQ Set Pending Control Register	25
0x0180	NVIC_ICPR	R/W	0x0000_0000	IRQ Clear Pending Control Register	25
0x0300	NVIC_IPR0	R/W	0x0000_0000	IRQ0 - IRQ3 Priority Control Register	26
0x0304	NVIC_IPR1	R/W	0x0000_0000	IRQ4 - IRQ7 Priority Control Register	27
0x0308	NVIC_IPR2	R/W	0x0000_0000	IRQ8 - IRQ11 Priority Control Register	28
0x030C	NVIC_IPR3	R/W	0x0000_0000	IRQ12 - IRQ15 Priority Control Register	29
0x0310	NVIC_IPR4	R/W	0x0000_0000	IRQ16 - IRQ19 Priority Control Register	30
0x0314	NVIC_IPR5	R/W	0x0000_0000	IRQ20 - IRQ23 Priority Control Register	31
0x0318	NVIC_IPR6	R/W	0x0000_0000	IRQ24 - IRQ27 Priority Control Register	32
0x031C	NVIC_IPR7	R/W	0x0000_0000	IRQ28 - IRQ31 Priority Control Register	33

4.2.6.1 NVIC_ISER - NVIC IRQ SET ENABLE CONTROL REGISTER

The NVIC_ISER registers enable interrupts, and show which interrupts are enabled.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SENTENA															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	**R/W**	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SENTENA															

Offset: 0x0000

Bits	Name	Type	Reset	Description
31:0	*SETENA*	R/W	0x0	Interrupt enable Enable one or more interrupts. Each bit represents and interrupt number from IRQ0 – IRQ31 0: On a read, indicates the interrupt is disabled. On a write, no effect 1: On a read, indicates the interrupt is enabled On a write, enables the interrupt

4.2.6.2 NVIC_ICER - NVIC IRQ CLEAR ENABLE CONTROL REGISTER

The NVIC_ICER registers disable interrupts, and show which interrupts are enabled.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
CLRENA															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	**R/W**	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
CLRENA															

Offset: 0x0080

Bits	Name	Type	Reset	Description
31:0	*CLRENA*	R/W	0x0	Interrupt disable Disable one or more interrupts. Each bit represents and interrupt number from IRQ0 to IRQ31. 0: On a read, indicates the interrupt is disabled. On a write, no effect 1: On a read, indicates the interrupt is enabled On a write, enables the interrupt

4.2.6.3 NVIC_ISPR - NVIC IRQ SET PENDING CONTROL REGISTER

The ISPR registers force interrupts into the pending state, and show which interrupts are pending.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SETPEND															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SETPEND															

Offset: 0x0100

Bits	Name	Type	Reset	Description
31:0	*SETPEND*	R/W	0x0	Set Interrupt Pending Disable one or more interrupts. Each bit represents and interrupt number from IRQ0 to IRQ31 0: On a read, indicates that the interrupt is not pending. On a write, no effect 1: On a read, indicates that the interrupt is pending. On a write, the corresponding interrupt is set to pending even if it is disabled

4.2.6.4 NVIC_ICPR - NVIC IRQ CLEAR PENDING CONTROL REGISTER

The ICPR registers remove the pending state from interrupts, and show which interrupts are pending.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
CLRPEND															

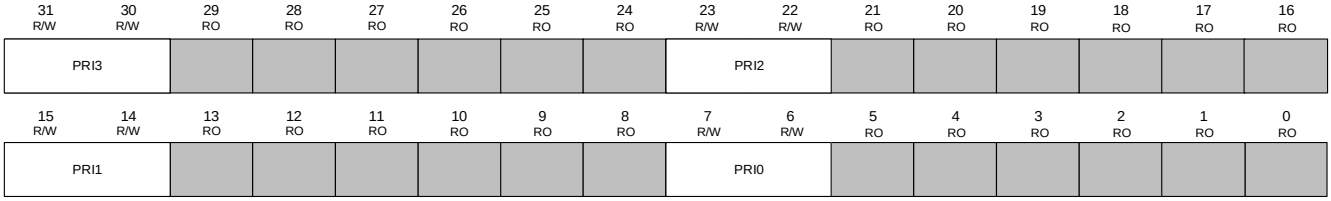
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
CLRPEND															

Offset: 0x0180

Bits	Name	Type	Reset	Description
31:0	*CLRPEND*	R/W	0x0	Set Interrupt Pending Disable one or more interrupts. Each bit represents and interrupt number from IRQ0 to IRQ31 0: On a read, indicates that the interrupt is not pending. On a write, no effect 1: On a read, indicates that the interrupt is pending. On a write, write 1 to clear pending state, so that the corresponding interrupt in no longer pending

4.2.6.5 NVIC_IPR0 - NVIC IRQ0 - IRQ3 PRIORITY CONTROL REGISTER

The IPR0 registers provide an 8-bit priority field for each interrupt and each register holds four priority fields (IRQ [3:0]). While setting each priority field, "0" always denotes the highest priority and "3" always denotes the lowest priority

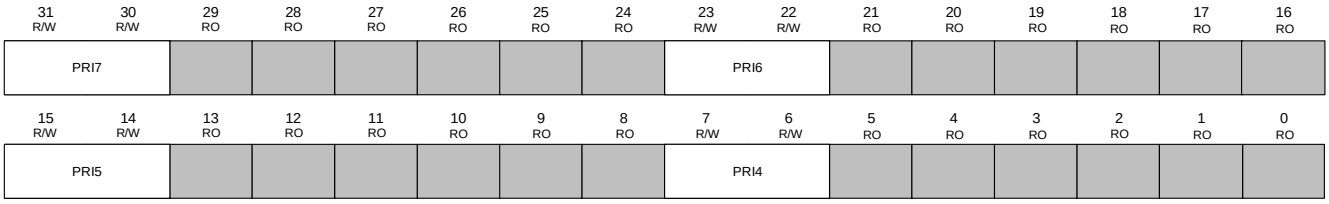


Offset: 0x0300

Bits	Name	Type	Reset	Description
31:30	PRI3	R/W	0x0	Priority of IRQ3
29:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:22	PRI2	R/W	0x0	Priority of IRQ2
21:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:14	PRI1	R/W	0x0	Priority of IRQ1
13:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	PRI0	R/W	0x0	Priority of IRQ0
5:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.2.6.6 NVIC_IPR1 - NVIC IRQ4 - IRQ7 PRIORITY CONTROL REGISTER

The IPR1 registers provide an 8-bit priority field for each interrupt and each register holds four priority fields (IRQ [7:4]). While setting each priority field, "0" always denotes the highest priority and "3" always denotes the lowest priority.

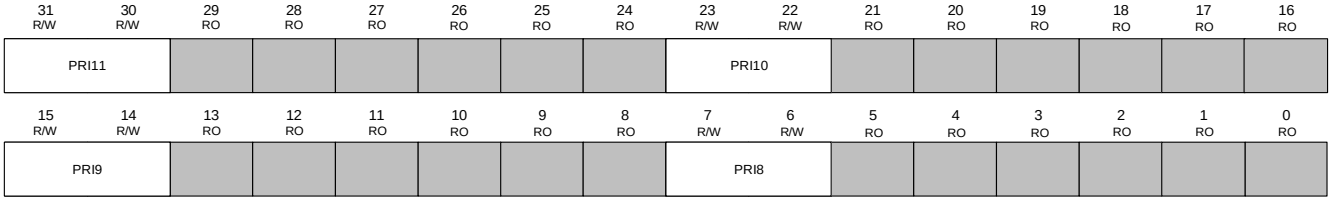


Offset: 0x0304

Bits	Name	Type	Reset	Description
31:30	PRI7	R/W	0x0	Priority of IRQ7
29:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:22	PRI6	R/W	0x0	Priority of IRQ6
21:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:14	PRI5	R/W	0x0	Priority of IRQ5
13:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	PRI4	R/W	0x0	Priority of IRQ4
5:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.2.6.7 NVIC_IPR2 - NVIC IRQ8 - IRQ11 PRIORITY CONTROL REGISTER

The IPR2 registers provide an 8-bit priority field for each interrupt and each register holds four priority fields (IRQ [11:8]). While setting each priority field, "0" always denotes the highest priority and "3" always denotes the lowest priority

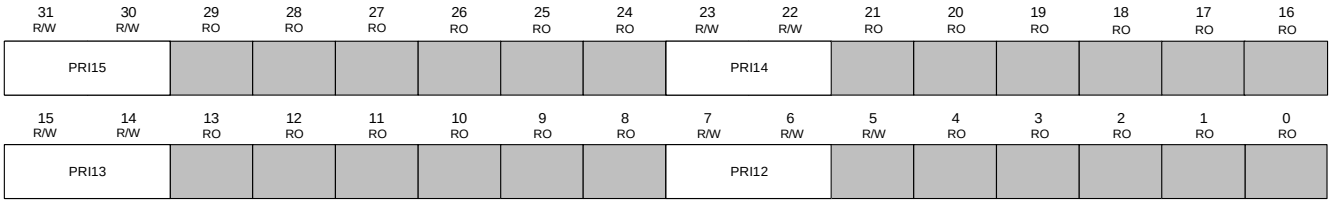


Offset: 0x0308

Bits	Name	Type	Reset	Description
31:30	PRI11	R/W	0x0	Priority of IRQ11
29:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:22	PRI10	R/W	0x0	Priority of IRQ10
21:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:14	PRI9	R/W	0x0	Priority of IRQ9
13:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	PRI8	R/W	0x0	Priority of IRQ8
5:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.2.6.8 NVIC_IPR3 - NVIC IRQ12 - IRQ15 PRIORITY CONTROL REGISTER

The IPR3 registers provide an 8-bit priority field for each interrupt and each register holds four priority fields (IRQ [15:12]). While setting each priority field,“0” always denotes the highest priority and “3” always denotes the lowest priority

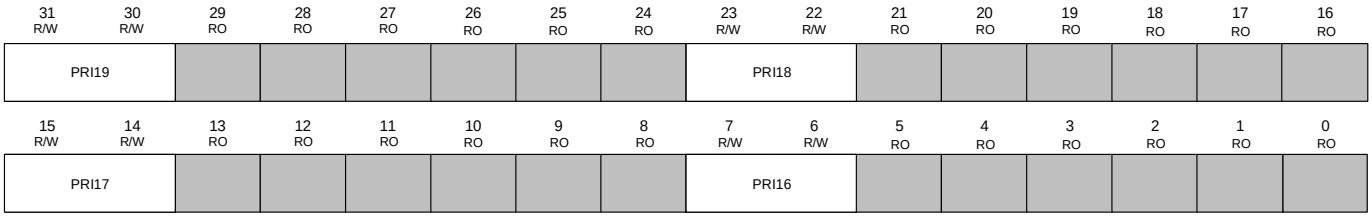


Offset: 0x030C

Bits	Name	Type	Reset	Description
31:30	PRI15	R/W	0x0	Priority of IRQ15
29:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:22	PRI14	R/W	0x0	Priority of IRQ14
21:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:14	PRI13	R/W	0x0	Priority of IRQ13
13:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	PRI12	R/W	0x0	Priority of IRQ12
5:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.2.6.9 NVIC_IPR4 - NVIC IRQ16 - IRQ19 PRIORITY CONTROL REGISTER

The IPR4 registers provide an 8-bit priority field for each interrupt and each register holds four priority fields (IRQ [19:16]). While setting each priority field,“0” always denotes the highest priority and “3” always denotes the lowest priority.

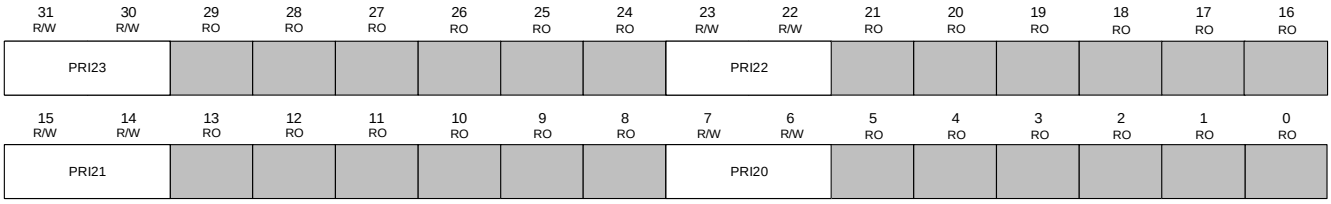


Offset: 0x0310

Bits	Name	Type	Reset	Description
31:30	PRI19	R/W	0x0	Priority of IRQ19
29:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:22	PRI18	R/W	0x0	Priority of IRQ18
21:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:14	PRI17	R/W	0x0	Priority of IRQ17
13:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	PRI16	R/W	0x0	Priority of IRQ16
5:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.2.6.10 NVIC_IPR5 - NVIC IRQ20 - IRQ23 PRIORITY CONTROL REGISTER

The IPR5 registers provide an 8-bit priority field for each interrupt and each register holds four priority fields (IRQ [23:20]). While setting each priority field,“0” always denotes the highest priority and “3” always denotes the lowest priority.

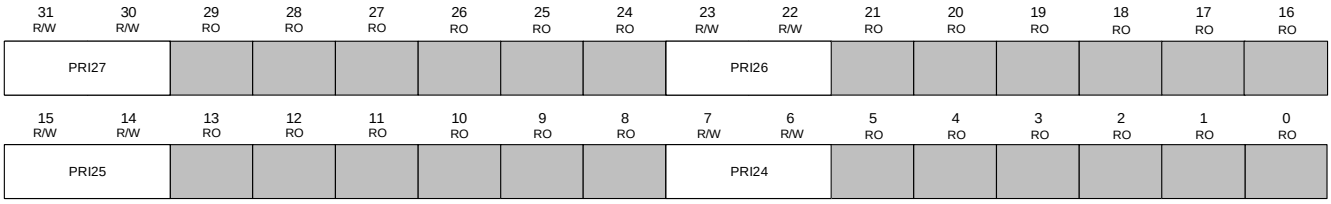


Offset: 0x0314

Bits	Name	Type	Reset	Description
31:30	PRI23	R/W	0x0	Priority of IRQ23
29:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:22	PRI22	R/W	0x0	Priority of IRQ22
21:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:14	PRI21	R/W	0x0	Priority of IRQ21
13:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	PRI20	R/W	0x0	Priority of IRQ20
5:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.2.6.11 NVIC_IPR6 - NVIC IRQ24 - IRQ27 PRIORITY CONTROL REGISTER

The IPR6 registers provide an 8-bit priority field for each interrupt and each register holds four priority fields (IRQ [27:24]). While setting each priority field,“0” always denotes the highest priority and “3” always denotes the lowest priority.

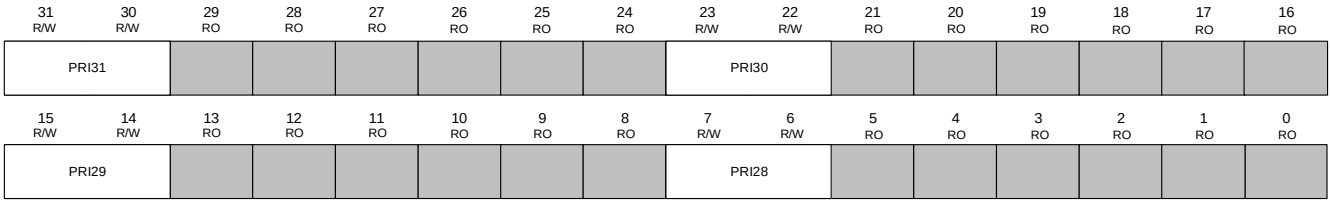


Offset: 0x0318

Bits	Name	Type	Reset	Description
31:30	PRI27	R/W	0x0	Priority of IRQ27
29:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:22	PRI26	R/W	0x0	Priority of IRQ26
21:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:14	PRI25	R/W	0x0	Priority of IRQ25
13:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	PRI24	R/W	0x0	Priority of IRQ24
5:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.2.6.12 NVIC_IPR7 - NVIC IRQ28 - IRQ31 PRIORITY CONTROL REGISTER

The IPR7 registers provide an 8-bit priority field for each interrupt and each register holds four priority fields (IRQ [31:28]). While setting each priority field,“0” always denotes the highest priority and “3” always denotes the lowest priority.



Offset: 0x031C

Bits	Name	Type	Reset	Description
31:30	PRI31	R/W	0x0	Priority of IRQ31
29:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:22	PRI30	R/W	0x0	Priority of IRQ30
21:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:14	PRI29	R/W	0x0	Priority of IRQ29
13:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	PRI28	R/W	0x0	Priority of IRQ28
5:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.3 CPU SYSTEM CONTROL

The Cortex™-M0 status and operating mode control are managed by CPU System Control Registers. Including CPUID, Cortex™-M0 interrupt priority and Cortex™-M0 power management can be controlled through these system control registers.

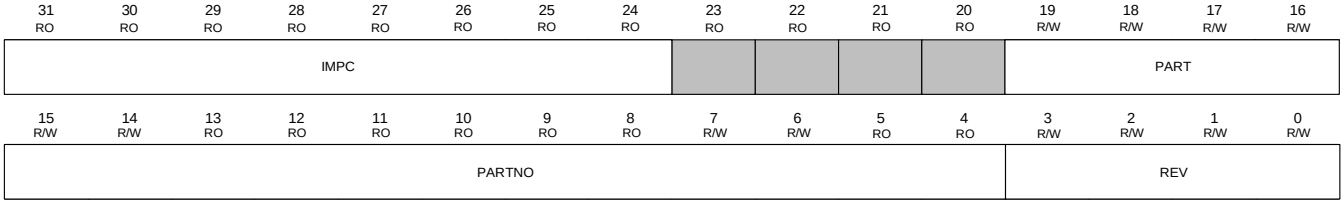
4.3.1 CPU SYSTEM CONTROL REGISTER MAP

Base Address 0xE000_ED00

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	SYS_CPUID	R/W	0x410C_C200	System CPUID Register	34
0x0004	SYS_ICSR	R/W	0x0000_0000	System Interrupt Control and State Register	35
0x000C	SYS_AIRCR	R/W	0xFA05_0000	System Application Interrupt and Reset Control Register	36
0x0010	SYS_SCR	R/W	0x0000_0000	System Control Register	37
0x001C	SYS_SHPR2	R/W	0x0000_0000	System Handler Priority Register 2	38
0x0020	SYS_SHPR3	R/W	0x0000_0000	System Handler Priority Register 3	38

4.3.1.1 SYS_CPUID – CPU ID REGISTER

This register provide identification for the processor.

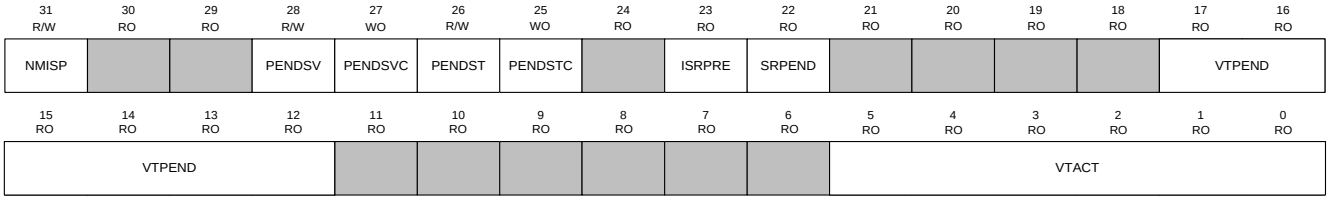


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:24	IMPC	RO	0x41	Implementer Code Assigned By ARM ARM = 0x41
23:20	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
19:16	PART	R/W	0xC	Architecture Of The Processor
15:4	PARTNO	RO	0xC20	Part Number Of The Processor
3:0	REV	R/W	0x0	Revision Number

4.3.1.2 SYS_ICSR - INTERRUPT CONTROL AND STATE REGISTER

This register controls and provides status information.



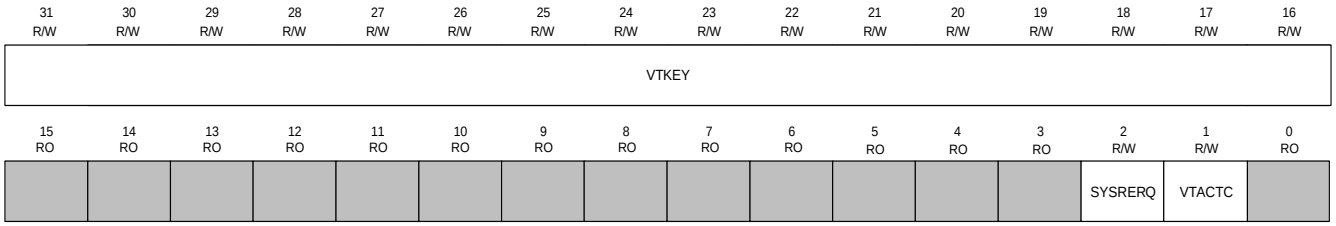
Offset: 0x0004

Bits	Name	Type	Reset	Description
31	NMISP	R/W	0	NMI (Non-maskable Interrupt) Set Pending 0: On a read, indicates an NMI exception is not pending. On a write, no effect. 1: On a read, indicates an NMI exception is pending. On a write, changes the NMI exception state to pending. Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.
30:29	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
28	PENDSV	R/W	0	PendSV Set Pending 0: On a read, indicates a PendSV exception is not pending. On a write, no effect. 1: On a read, indicates a PendSV exception is pending. On a write, changes the PendSV exception state to pending. Only by writing a '1' to this bit can set the PendSC exception state to pending
27	PENDSVC	WO	0	PendSV Clear Pending 0: No effect. 1: Removes the pending state from the PendSV exception
26	PENDST	R/W	0	SysTick Exception Set-Pending Bit 0: On a read, indicates a SysTick exception is not pending. On a write, no effect. 1: On a read, indicates a SysTick exception is pending. On a write, changes the PendSV exception state to pending.
25	PENDSTC	WO	0	SysTick Exception Clear-Pending Bit 0: No effect. 1: Removes the pending state from the SysTick exception
24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23	ISRPRE	RO	0	Debug Interrupt Handling 0: The release from halt does not take an interrupt 1: A pending exception will be serviced On Exit From The Debug Halt State.
22	SRPEND	RO	0	Interrupt Pending Flag, Excluding NMI And Faults 0: Interrupt is not pending 1: Interrupt is pending

Bits	Name	Type	Reset	Description
21:18	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
17:12	VTPEND	RO	0x0	Interrupt Pending Vector Number 0: No pending exceptions. Non-zero: Exception number of the highest priority pending enabled exception.
11:6	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5:0	VTACT	RO	0x0	Interrupt Pending Vector Number This field contains the active exception number. 0: Thread mode. Non-zero: Exception number of the currently active exception.

4.3.1.3 SYS_AIRCR - APPLICATION INTERRUPT AND RESET CONTROL REGISTER

This register sets or returns interrupt control data.

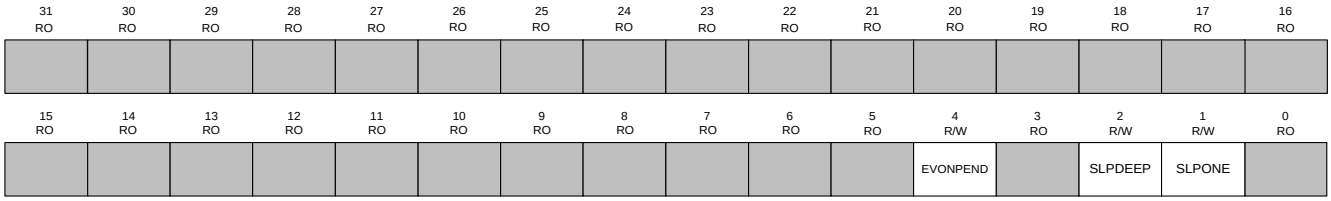


Offset: 0x000C

Bits	Name	Type	Reset	Description
31:16	VTKEY	R/W	0xFA05	Register Access Key When writing to this register, the VTKEY field need to be set to 0x05FA, otherwise the write operation would be ignored. The VTKEY filed is used to prevent accidental write to this register from resetting the system or clearing of the exception status.
15:3	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	SYSRERQ	R/W	0	System Reset Request 0: Do not request a rest 1: Request a reset
1	VTACTC	R/W	0	Clear Active NMI / Fault Reserved for debug use. When writing to the register, user must write 0 to this bit, otherwise behavior is unpredictable.
0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.3.1.4 SYS_SCR - SYSTEM CONTROL REGISTER

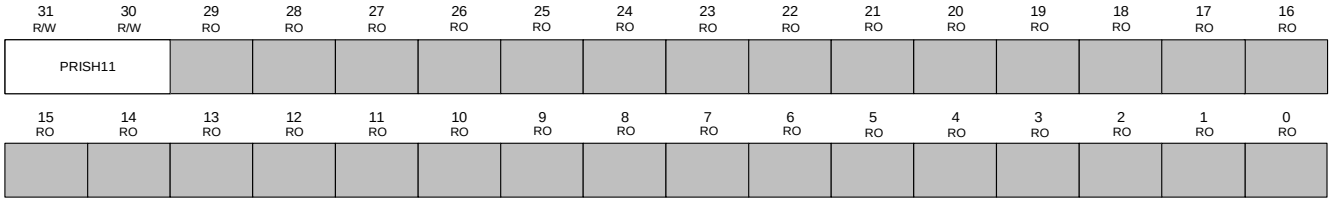
The SCR controls features of entry to and exit from low power state.



Offset: 0x0010

Bits	Name	Type	Reset	Description
31:5	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	EVONPEND	R/W	0x0	Send Event On Pending Bit 0: Only enabled interrupts or events can wake the processor up. (Disabled interrupts are not included) 1: All enabled interrupts, event and disabled interrupts can wake the processor up. When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects the next WFE. The processor also wakes up on execution of an SEV instruction or an external event.
3	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	SLPDEEP	R/W	0	Deep Sleep and Sleep Mode selection Controls whether the processor uses sleep or deep sleep as its low power mode: 0: Sleep mode. 1: Deep Sleep mode.
1	SLPONE	R/W	0	Sleep-On-Exit Enable 0: Do not sleep when returning to Thread mode. 1: Enter Sleep or Deep Sleep when returning from ISR to Thread mode. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.
0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

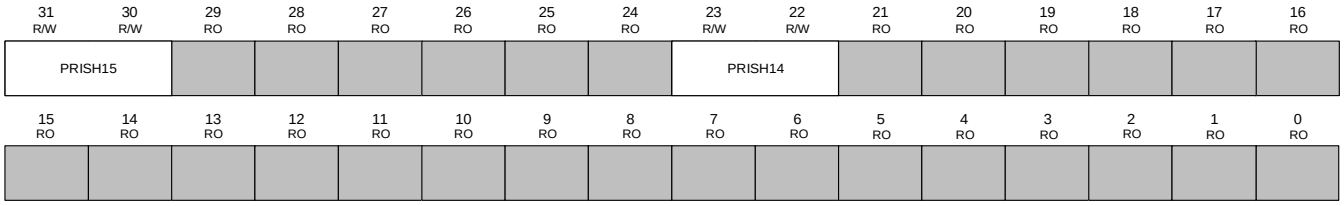
4.3.1.5 SYS_SHPR2 - SYSTEM HANDLER PRIORITY REGISTER 2



Offset: 0x001C

Bits	Name	Type	Reset	Description
31:30	PRISH11	R/W	0	Priority of System Handler 11 - SVCall “0” denotes the highest priority and “3” denotes the lowest priority
29:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.3.1.6 SYS_SHPR3 - SYSTEM HANDLER PRIORITY REGISTER 3



Offset: 0x0020

Bits	Name	Type	Reset	Description
31:30	PRISH15	R/W	0x0	Priority of System Handler 15 - SysTick “0” denotes the highest priority and “3” denotes the lowest priority
29:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:22	PRISH14	R/W	0x0	Priority of System Handler 14 - PendSV “0” denotes the highest priority and “3” denotes the lowest priority
21:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.4 SYSTEM CONTROL (SC)

System control configures the overall operation of the device and provides information about the device. SC in PT32U301 configures following features:

- Device Identification
- Booting Configuration
- ICE Protection
- Peripheral Management
- System Tick Calibration
- Clock Control
- Power Control
- Low-power modes
- Backup Control

4.4.1 FUNCTIONAL DESCRIPTION

4.4.1.1 DEVICE IDENTIFICATION

The **SC_PID0** and **SC_PID1** registers related to device identification provide software with information on the microcontroller, such as Flash / SRAM memory space, product quality level, packaging and product version. In addition, the **SC_PID1** register's content such as package type *PKG*, temperature test range *TEMP* and quality grade *QUAL* are set by programming the Eflash Info (0x7F4 - 0x7F7). The Eflash will be set by Mask ROM boot up. User may add extra identification for the microcontroller via the **SC_UID0** and **SC_UID1** in later development.

4.4.1.2 ICE PROTECTION

The user area of the Flash memory can be protected against read by untrusted code. The read protection is activated by assigning specified value to an address in embedded flash information block; the protection is therefore activate after Power-on reset (POR). User is able to check the protection status by the bit *ICE* in register **SC_BK_SCTRL**.

4.4.1.3 BOOTING CONFIGURATION

In the PT32C302, four different boot modes can be selected using memory remapping register (**SC_REMAP)**. The *REMAP* bit of the **SC_REMAP** register controls the boot modes, as shown in the following table.

Table 4.4-1: Boot Modes

Boot mode configuration REMAP bit	Mode*
0x0	**Flash Information** is selected as boot space.
0x1	**SRAM** is selected as boot space.
0x2	**Main Flash Memory** is selected as boot space.
0x3	**ROM** is selected as boot space

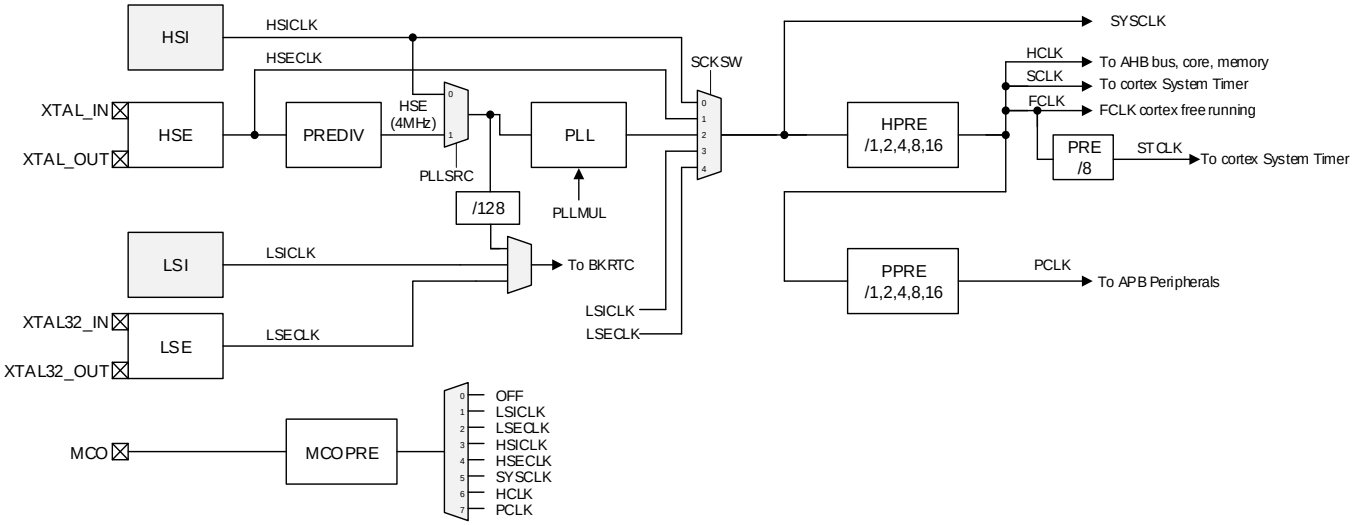
*: The boot mode configuration is sampled in a power-on reset or a system reset.

Depending on the selected boot mode, Flash Information, SRAM, Main Flash memory and ROM is accessible as follows:

- Boot from Flash Information: the flash information memory is aliased in the boot memory space (0x0000 0000), but still accessible from its original memory space (0x1FFF F000).
- Boot from SRAM: the SRAM is aliased in the boot memory space (0x0000 0000), but still accessible from its original memory space (0x2000 0000).
- Boot from Main Flash Memory: the main flash memory is aliased in the boot memory space (0x0000 0000),[A1] but still accessible from its original memory space (0x0800 0000). User can select each 4k Byte of flash main block as base address(refer to)
- Boot from ROM: Normal booting mode, the ROM memory is aliased in its original memory space (0x0000 0000).

4.4.1.4 CLOCK SYSTEM

Figure 4.4-1: Clock Tree



CLOCK SOURCE

There are multiple clock sources for use in the microcontroller:

- HSI Clock
  - High-Speed Internal Clock (HSI) signal is generated from an internal 4 MHz RC oscillator and be used directly as a system clock or for PLL input or RTC clock source.
- HSE Clock
  - High-Speed External Clock (HSE) signal is generated from an external crystal oscillator. This clock can be used for PLL input or RTC clock source.
- LSI Clock
  - Low-Speed Internal Clock (LSI) signal is generated from an internal RC oscillator whose frequency is around 30 kHz.
- LSE Clock
  - The Low-Speed External (LSE) crystal is a 32.768 kHz Low Speed External crystal oscillator. It provides a highly accurate clock source to the RTC for clock/calendar or other timing functions.

CLOCK DISTRIBUTION

The clock system on the PT32C302 consists of:

SYSTEM CLOCK - SYSCLK

All oscillators can be used as a source for the system clock (SYSCLK). Various clock sources can be used to drive the SYSCLK controlled by bit *SCKSW* in System clock configuration register (**SC_CK_CONF.).**

- 4 MHz HSI RC oscillator clock
- PLL clock
- HSE oscillator clock
- LSE 32.768kHz oscillator clock

The SYSCLK is derived by any of the above sources. After booting and reset, the HSI oscillator is selected as SYSCLK's clock source. When any of the above clock sources is used as system clock, it is not possible to stop it.

To switch the clock source, user need to first confirm if the target clock source is ready from clock stable after startup delay or PLL locked. The respective clock source status bit (Ready flag) is shown in SC_CK_STAT register. If a clock source which is not yet ready is selected, the switch will occur when the clock source becomes ready.

It is recommended to switch on the HSI clock all the time for safety precaution in case any malfunction of the external clock source.

REAL TIME COUNTER CLOCK- RTCCLK

There are three clock source can be used to derive the RTC clock (RTCCLK) controlled by bit *CLKSEL* in RTC clock control register **(RTC_CTRL).**

- PLL reference CLK divided by 128
- LSE 32.768kHz oscillator clock
- LSI RC oscillator clock

For detailed description of RTC, please refer to Real Time Clock (RTC).

HCLK, PCLK

The AHB clock (HCLK) is derived from System clock through configurable prescaler and used to clock the CPU, memory and peripherals mapped on AHB bus. The APB clock (PCLK) is derived from AHB clock though configurable prescaler and used to clock the peripherals mapped on APB bus.

STCLK

Adding a 1/8 SYSCLK frequency as a PT32C302 STCLK option. Users can choice CLK by configuring the CLKSOURCE of SYST_CSR. When setting CLKSOURCE as 0, it means that system frequency CLK 1/8 is set as CLK of SystemTick. As CLKSOURCE is 1, system frequency is configured as SystemTick CLK. Refer to M0 manual for SystemTick settings.

4.4.1.5 POWER

VOLTAGE REGULATOR

The Backup Domain includes the RTC, and Backup registers. It is powered by VDD power source 2.0 V-3.6V. The Backup Register and LSI Oscillator allow user to save the maximum power by switching off the internal regulator.

VOLTAGE REGULATOR

A voltage regulator is embedded in PT32C302 supplying the internal digital power domain. The device requires 2V-3.6V operating supply voltage. The Real-Time Clock (RTC) and backup register can be powered even the regulator is off.

The voltage regulator is always enabled after Reset. It works in three different modes depending on the application modes.

- Run mode

  Regulator supplies full power to the internal digital domain (core, memories and digital peripherals).

- Stop mode

  Regulator supplies low-power to the digital domain, preserving contents of registers and SRAM

- Standby Mode

  Regulator is powered off. The contents of the registers and SRAM are lost except for the Standby circuitry and the Backup domain.

RESET CONTROL

In PT32C302, a system reset is generated when a power reset or PIN reset occurs.

PIN RESET

A reset can be caused by a detected falling edge on NRST line (i.e. *PD11_NRST* in PT32C302). The system reset that is generated by NRST is different from Power Reset, as PIN reset will not erase the content stored in Backup Domain while power reset will.

Note: PD11_NRST pin can also be used to wake up the system in Standby/Stop mode. Refer to Low Power Mode for detailed description.

POWER RESET

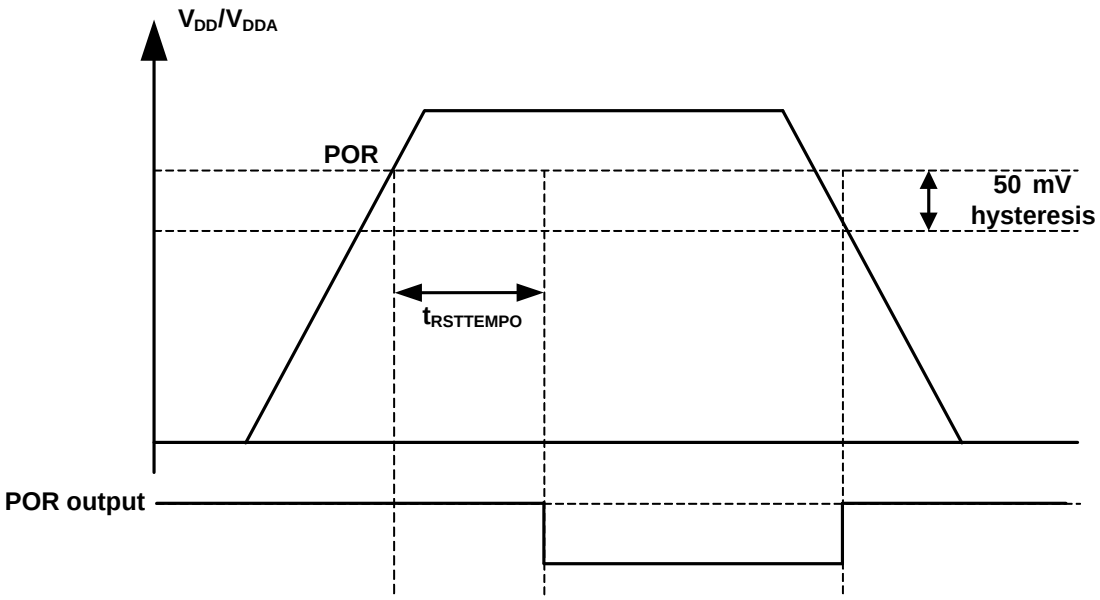
A power reset is caused by the power supply dropping below a level needed to keep the processor running. This device has three ways to monitor and reset the device:

- Power On Reset (POR)
- Power Down Reset (PDR)
- Programmable Voltage Detector (PVD)

Power on reset (POR) / Power Down Reset (PDR)

The device has an integrated POR/PDR circuitry that allows proper operation starting from/down to 2 V. The device remains in Reset mode when VDD/VDDA is below a specified threshold, VPOR/PDR, without the need for an external reset circuit. For more details concerning the power on/power down reset threshold, refer to the Electrical Characteristics section.

Figure 4.4-2: Power on reset and power down reset waveform

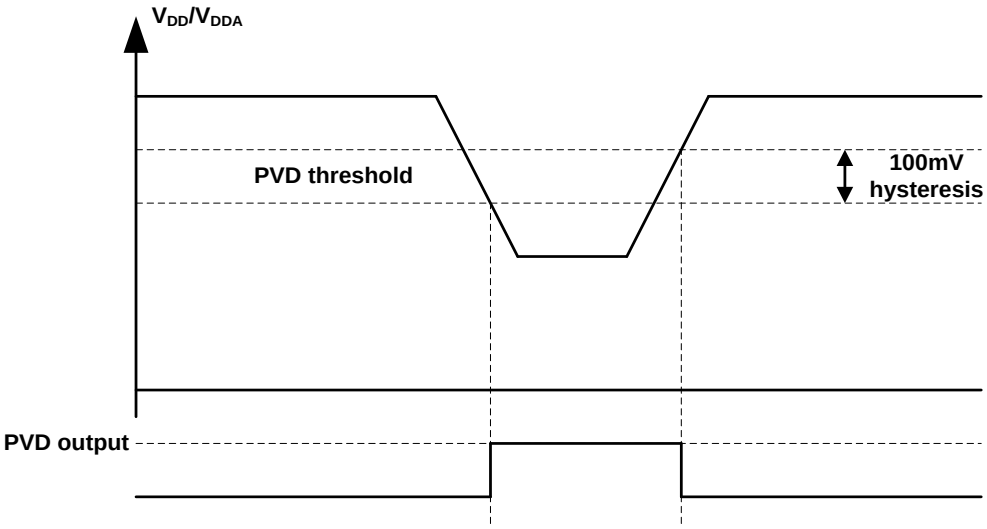


In PT32C301, the Power-on reset (NRST) must be provided externally (pin PD11_NRST). To guarantee proper power on and power down reset to the device, the PD11_NRST pin must be held low until V_{DD33} is stable or before turning off the supply. When V_{DD33} is stable, the reset state can be excited by putting the PD11_NRST pin in high impedance. The PD11_NRST pin has an internalpull-up connected to V_{DD}.

Programmable voltage detector (PVD)

PVD is used to monitors the V_{DD} / V_{DDA} power supply by comparing it to a threshold selected by the *VOLT* bits in the PVE Detection Register (**SC_PVD_DET**). The PVD is enabled by setting the *DETEN* bit in (**SC_PVD_DET**).

Figure 4.4-3: PVD thresholds



4.4.1.6 LOW POWER MODE

By default, the microcontroller is in Run mode after a system or a power Reset. Several low-power modes are available to save power when the CPU does not need to be kept running.
For example, when waiting for an external event. It is up to the user to select the mode that gives the best compromise between low-power consumption, short startup time and available wakeup sources.

The device features three level of low-power modes:

- Level 0 : Sleep mode (CPU clock off, all peripherals including Cortex® -M0 core peripherals like NVIC, etc. are kept running)
- Level 1 : Stop mode (all clocks are stopped, except the RTC clock source)
- Level 2 : Standby mode (1.8V domain powered-off)

Table 4.4-2: Low-power mode summary

Mode (Level)	Entry	Wakeup	Effect on 1.8V Domain clocks	Effect on 3.3V Domain clocks	Voltage regulator (LDO)
Sleep (Level 0)	WFI	Any interrupt	FCLK is on, other clock's status are user-configurable.	None	ON
Stop (Level 1)	*SLPDEEP* bit + WFI	Any External WKUP pin or internal wake up events (RTC, PVD)	All 1.8V domain Clocks OFF	HSI and HES oscillation OFF	Low Power Mode
Standby (Level 2)	*SLPDEEP* bit + WFI	Any External WKUP pin or internal wake up events (RTC, PVD)			OFF

In addition, the power consumption in Run mode can be reduce by one of the following means:

- Slowing down the system clocks
- Gating the clocks to the APB and AHB peripherals when they are unused.
- Putting flash memory into sleepmode

SLOWING DOWN SYSTEM CLOCKS

In Run mode the speed of the system clocks (SYSCLK, HCLK, and PCLK) can be reduced by programming the prescaler registers. These prescalers can also be used to slow down peripherals before entering Sleep mode

PERIPHERAL CLOCK GATING

In Run mode, the AHB clock (HCLK) and the APB clock (PCLK) for individual peripherals and memories can be stopped at any time to reduce power consumption. To further reduce power consumption in Sleep mode, the peripheral clocks can be disabled prior to executing the WFI instructions

ENABLING FLASH MEMORY SLEEP MODE

The flash memory (Main block and information block) can be put into sleepmode to reduce power usage under all low power mode. User write 1 to the bit *SLEEP* bit in the Flash Controller Command Register **FC_CMD** to enable the flash memory sleep mode. Once this bit is set, the flash memory will enters sleepmode as soon as the device enter low power modes.

LEVEL 0 - SLEEP MODE

Entering Level 0

Level 0 (Sleep Mode) is entered by executing the WFI (Wait for Interrupt) instructions.

In level 0, all I/O pins remain in the same state as in the Run mode. To allow specific peripheral keep running under sleep mode, user should first configure register **SC_SLP_APB**/**SC_SLP_AHB** to enable the clock to the peripheral before entering sleep mode. When clock gating is enabled, the peripheral will keep running when microcontroller is in Sleep mode. In the Sleep mode, all I/O pins remain in the same state as in the Run mode.

Exiting Level 0

Any peripheral interrupt acknowledged by the nested vectored interrupt controller (NVIC) can wake up the device from Sleep mode.

LEVEL 1 - STOP MODE

Entering Level 1

Level 1 (Stop Mode) is based on the Cortex®-M0 deepsleep mode. The system is entering stop mode as soon as it meets the following conditions:

- Level 1 is selected by configuring **SC_BK_LPCTRL**.
- WFI is received.
- *SLPDEEP* bit is received.

The voltage regulator (LDO) is configured as in low-power mode. All clocks in 1.8V domain, except RTC clock, are switched off. PLL, HSI and HSE oscillator are switched off. Peripheral that build in 1.8V domain preserve all data when entering stop mode.

Exiting Level 1

The device is waken up from the stop mode when a rising/falling edge on one of the enabled WKUP pins, an external reset (NRST pin), PVD reset and RTC wakeup event occurs.

While the voltage regulator is operating at low-power mode, an additional startup delay is incurred when waking up from Stop mode. The wake up event status flag in the Backup Status Register (SC_BK_STATUS) indicates that which wake up event has waken MCU. After awakening from Stop mode, user should write 1 to CWUF bit in **SC_BK_LPCTRL** to clear the wake up event flag.

WKUP Pins Configuration:

Note that these external wake up pins are edge sensitive, and user can configure whether these pins are rising edge sensitive or falling edge sensitive (except PD11_NRST which must be falling edge sensitive) by configuring the respected *WUEDGEx* bit in the Backup Low Power Sleep Controller Register (**SC_BK_LPCTRL**) register.

LEVEL 2 - STANDBY MODE

Entering Standby mode

Level 2 (Standby Mode) achieve the lowest power consumption. It is based on the Cortex®-M0 **deepsleep** mode, with the voltage regulator disabled. The regulator power domain is consequently powered off. The PLL, the HSI oscillator and the HSE oscillator are also switched off. SRAM and register contents are lost except for registers in the Backup domain and Standby circuitry.

The system is entering standby mode as soon as it meets the following conditions:

- Level 2 is selected by configuring **SC_BK_LPCTRL**.
- WFI is received.
- *SLPDEEP* bit is received.

Exiting Standby mode

The device wakes up from the stop mode when a rising/falling edge on one of the enabled WKUP pins, an external reset (NRST pin), PVD reset and RTC wakeup event occurs. All registers are reset after waking up from Standby except backup domain.

After exiting Standby mode, program execution restarts in the same way as after a Reset. The wake up event status flag in **SC_BK_STATUS** indicates that which wake up event wakes up the MCU. After waking up from Standby mode, user should set *CLEAR* bit in **SC_BK_LPCTRL** to clear wake up flag.

WKUP Pins Configuration:

Note that these external wake up pins are edge sensitive, and user can configure whether these pins are rising edge sensitive or falling edge sensitive by configuring the respected *WUEDGEx* bit in the Backup Low Power Sleep Controller Register (**SC_BK_LPCTRL**) register. In Standby mode, the 1.8 V domain is powered off. While an external wakeup pins is used to wake the system up from this mode, it must be connected to pullup/pulldown resistor according to the wakeup event setting. Note that the NRST pin (PD11_NRST) is forced in input pull up configuration which does not require any external resistor.

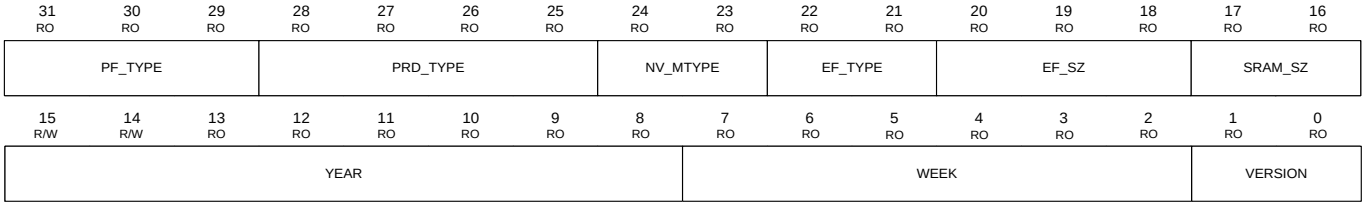
4.4.2 SYSTEM CONTROL REGISTER MAP

Base Address: 0x5000_0000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	SC_PID0	RO	0x058D_10C0	Product ID0 number Register	51
0x0004	SC_PID1	R/W	0x0000_00E0	Product ID1 number Register	52
0x0008	SC_UID0	R/W	0x0000_0000	User define field ID0	53
0x000C	SC_UID1	R/W	0x0000_0000	User define field ID1	53
0x0010	SC_REMAP	R/W	0x0000_0F36	Internal remap by software configuration	54
0x0018	SC_STCALIB	R/W	0x0100_0148	System Tick Timer Calibration Value	55
0x001C	SC_NVM_ACR	R/W	0x0000_0040	NVM (FLASH) Access Control Register	56
0x0020	SC_GCLK_APB	R/W	0x0000_0000	APB Peripheral Clock Gating Enable Register	57
0x0024	SC_GCLK_AHB	R/W	0x0000_0000	AHB Peripheral Clock Gating Enable Register	59
0x0028	SC_RST_APB	R/W	0x0000_0000	APB Peripheral Reset Request Register	60
0x002C	SC_RST_AHB	R/W	0x0000_0000	AHB Peripheral Reset Request Register	62
0x0030	SC_CK_CTRL	R/W	0x0000_0001	Clock Control Register	64
0x0034	SC_CK_CONF	R/W	0x0000_0000	Clock Configuration Register	65
0x0038	reserved	N/A	0x0000_0000	Reserved	-
0x003C	SC_CK_STAT	RO	0x0000_0000	Clock Status Register	67
0x0040	SC_BK_CTRL	R/W	0x0000_0000	Backup Control Register	69
0x0044	reserved	N/A	0x0000_0000	Reserved	-
0x0048	SC_PWR_CTRL	R/W	0x0000_0000	Power Control Register	69
0x004C	SC_PVD_DET	R/W	0x0000_000E	PVD Detection	70
0x0050	SC_SLP_APB	R/W	0x0000_0000	Into Sleep APB Peripheral Clock Gating Enable Register	71
0x0054	SC_SLP_AHB	R/W	0x0000_0000	Into Sleep AHB Peripheral Clock Gating Enable Register	73
0x0058	SC_DSLP_APB	R/W	0x0000_0000	Into Deep Sleep APB Peripheral Clock Gating Enable Register	75
0x005C	SC_DSLP_AHB	R/W	0x0000_0000	Into Deep Sleep AHB Peripheral Clock Gating Enable Register	77
0x0060	reserved	N/A	0x0000_0000	Reserved	-
0x0064	reserved	N/A	0x0000_0000	Reserved	-
0x0068	reserved	N/A	0x0000_0000	Reserved	-
0x006C	SC_OSC_TRIM	R/W	0x0080_0000	Ring OSC Hardware Trimming Register	78
0x0070	reserved	N/A	0x0000_0000	Reserved	-
0x0074	reserved	N/A	0x0000_0000	Reserved	-
0x0078	reserved	N/A	0x0000_0000	Reserved	-
0x007C	reserved	N/A	0x0000_0000	Reserved	-
0x0080	SC_BK_LPCTRL	R/W	0x00C0_0000	Backup Low Power Sleep Controller Register	79
0x0084	SC_BK_STATUS	RO	0x0000_0000	BackUp Status Register	80
0x0088	SC_BK_SCTRL	R/W	0x0010_8000	BackUp System Control Register	81
0x008C	reserved	N/A	0x0000_0000	Reserved	-
0x0090	SC_BK_REG0	R/W	0x0000_0000	Backup 0 Register	82
0x0094	SC_BK_REG1	R/W	0x0000_0000	Backup 1 Register	82

Offset	Symbol	Type	Reset Value	Description	See page
0x0098	SC_BK_REG2	R/W	0x0000_0000	Backup 2 Register	82
0x009C	SC_BK_REG3	R/W	0x0000_0000	Backup 3 Register	82
0x00A0	SC_BK_REG4	R/W	0x0000_0000	Backup 4 Register	82
0x00A4	SC_BK_REG5	R/W	0x0000_0000	Backup 5 Register	82
0x00A8	SC_BK_REG6	R/W	0x0000_0000	Backup 6 Register	82
0x00AC	SC_BK_REG7	R/W	0x0000_0000	Backup 7 Register	82

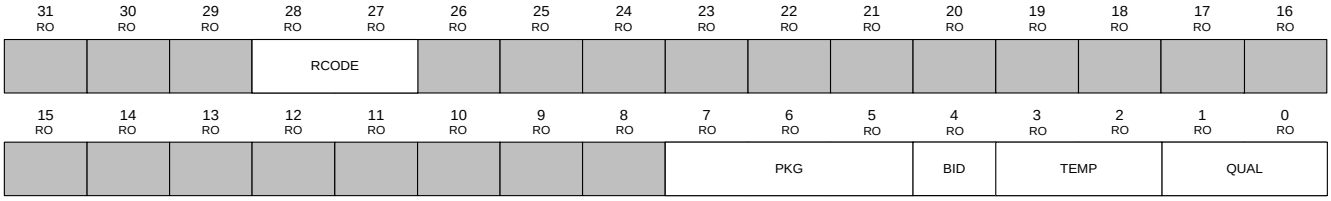
4.4.2.1 SC_PID0 - PRODUCT ID0 NUMBER REGISTER



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:29	*PF_TYPE*	RO	000	Product Type 0x0 : Cortex-M0 MCU Platform 0x1 : Cortex-M3 MCU Platform 0x2 : Cortex-M4 MCU Platform 0x3 : ASIC 0x4 : PL
28:25	*PRD_TYPE*	RO	0010	Product Type 0x0 : General Propose MCU 0x1 : BLDC Application 0x2 : USB Application 0xE : EMC Application 0xF : TSC Application
24:23	*NVM_TYPE*	RO	0x3	0x0 : OTP 0x1 : MTP 0x2 : EEPROM 0x3 : Embedded Flash
22:21	*EF_TYPE*	RO	0x0	Eflash Type 0: Single Embedd Flash Memory 1: Dual Embedd Flash Memory
20:18	*EF_SZ*	RO	0x3	Embedded Flash Size 0x0 : 8K 0x1 : 16K 0x2 : 32K 0x3 : 64K Byte 0x4 : 128 Kbyte(Reserved)
17:16	*SRAM_SZ*	RO	0x1	SRAM Size 4X(SRAM_SZ+1) K Byte
15:8	*YEAR*	RO	16	Release Year e.g. 20(16)…
7:2	*WEEK*	RO	52	Release Week by year
1:0	*VERSION*	RO	0x0	Major Revision 0x0: Revision A (initial device) 0x1: Revision B (first base layer revision) 0x2: Revision C (second base layer revision) 0x3: Revision D (last revision)

4.4.2.2 SC_PID1 - PRODUCT ID1 NUMBER REGISTER

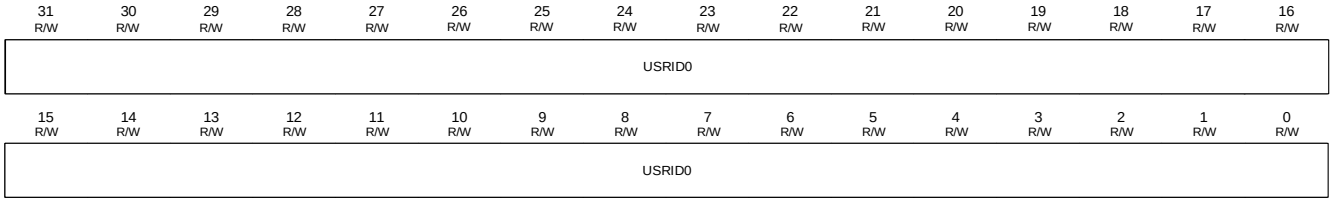


Offset: 0x0004

Bits	Name	Type	Reset	Description
31:29	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
28:27	*RCODE*	RO	0x0	Rom Code Version 0x0: Revision A (initial device) 0x1: Revision B (first base layer revision) 0x2: Revision C (second base layer revision) 0x3: Revision D (last revision)
26:8	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:5	*PKG*	R/W	0x7	Package Type 0x0: 8-pin SOP package 0x1: 16-pin SSOP package 0x2: 20-pin SSOP package 0x3: 24-pin SSOP package 0x4: 32-pin QFN package 0x5: 48-pin LQFP package 0x6: 64-pin LQFP package 0x7: other special package type
4	*BID*	R/W	0	Bounding Wire Type 0: Gold 1: Copper
3:2	*TEMP*	R/W	0x0	Temperature Range 0x0: Commercial temperature range (0°C to 70°C) 0x1: Industrial temperature range (-40°C to 85°C) 0x2: Extended temperature range (-40°C to 125°C)
1:0	*QUAL*	R/W	0x0	Qualification Status 0x0: Engineering Sample (unqualified) 0x1: Pilot Production (unqualified) 0x2: Fully Qualified (only FT) 0x3: Fully Qualified (CP/FT)
1:0	*QUAL*	RO	0x0	Qualification Status 0x0: Engineering Sample (unqualified) 0x1: Pilot Production (unqualified) 0x2: Fully Qualified (only FT) 0x3: Fully Qualified (CP/FT)

4.4.2.3 SC_UID0 - USER DEFINE FIELD ID0 REGISTER

The SC_UID0 let user further defining the product.

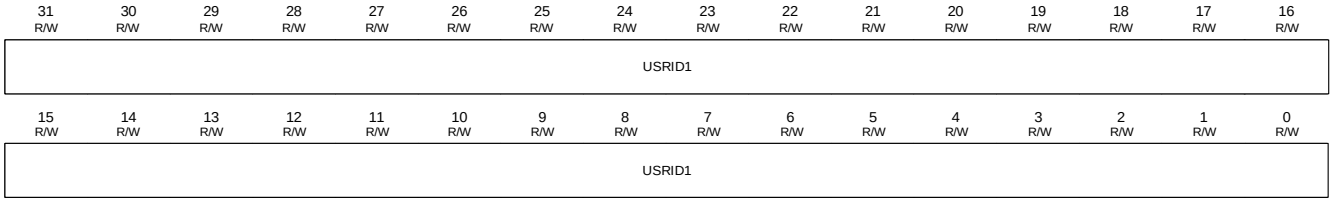


Offset: 0x0008

Bits	Name	Type	Reset	Description
31:0	*USR_ID0*	R/W	0x0	User define field ID0

4.4.2.4 SC_UID1 - USER DEFINE FIELD ID1 REGISTER

The SC_UID0 let user further defining the product.

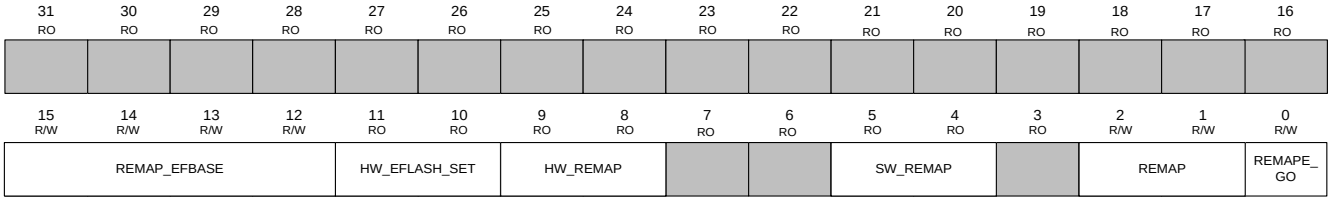


Offset: 0x000C

Bits	Name	Type	Reset	Description
31:0	*USR_ID1*	R/W	0x0	User define field ID1

4.4.2.5 SC_REMAP - MEMORY REMAP BY SOFTWARE CONFIGURATION REGISTER

This register provide software to change the location of the boot up process, such as Eflash, SRAM.

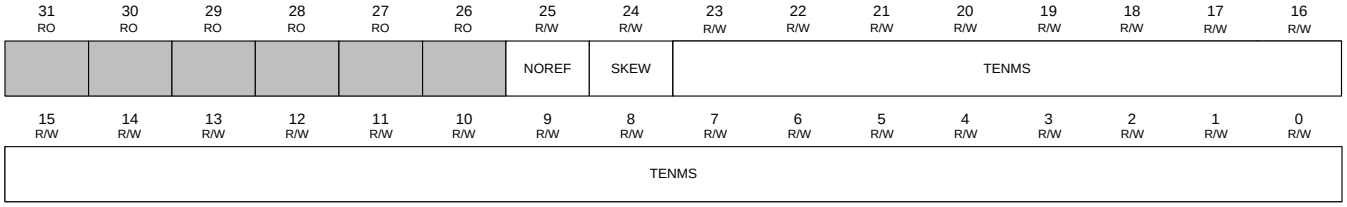


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:12	*REMAP_EFBASE*	R/W	0	Remap Eflash Base ADDR Selection This field is available only when *REMAP* = 0x2. Refer to Main Flash Memory Remap for details. The Eflash memory can be accessed starting from: 0x0: First 4k Byte location 0x1: Second 4k Byte location 0x2: Third 4k Byte location … 0x7: Eighth 4k Byte location.
11:10	*HW_EFLASH_SET*	RO	0x3	Maximum EFlash Size 0x0 : 16K 0x1 : 32K 0x3 : 64K
9:8	*HW_REMAP*	RO	0x3	External Hardware Remap Status Register 0x2 : Eflash 0x3 : Mask ROM
7:6	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5:4	*SW_REMAP*	RO	0x3	Internal Remap Selected Status Register 0x0 : Eflash Information 0x1 : SRAM 0x2 : Eflash Memory 0x3 : Mask ROM
3	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:1	*REMAP*	R/W	0x3	Internal Remap Select Register 0x0 : Eflash Information 0x1 : SRAM 0x2 : Eflash Memory 0x3 : Mask ROM
0	*REMAPE_GO*	R/W	0	Internal Remap Enable If set to1, enter into REMAP Flow. It will be cleared after finish REMAP Flow.

4.4.2.6 SC_STCALIB - SYSTEM TICK TIMER CALIBRATION VALUE

The SC_STCALIB register indicates the SysTick calibration properties

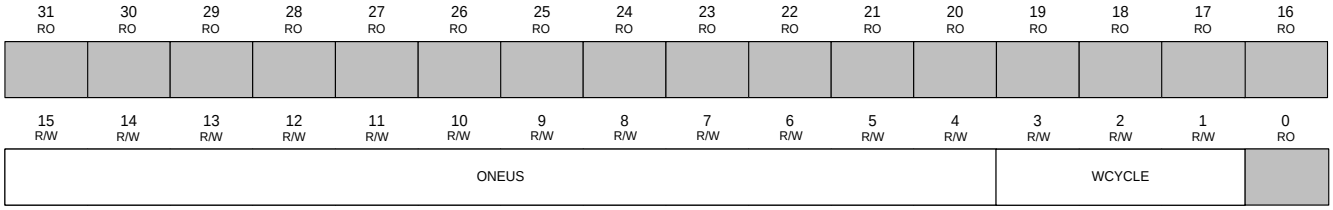


Offset: 0x0018

Bits	Name	Type	Reset	Description
31:26	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
25	*NOREF*	R/W	0	SysTick Timer Reference 0: An external reference clock is needed 1: Indicating System Tick always adopt core clock for counting
24	*SKEW*	R/W	1	1: *TENmS* bits field is not accurate.
23:0	*TENMS*	R/W	0x148	Ten millisecond calibration value. The value is MCU design dependent.

4.4.2.7 SC_NVMACR - NVM (FLASH) ACCESS CONTROL REGISTER

The **SC_NVMACR** register is the control register for Flash program/erase operations. This register selects whether an erase or program operation can be performed and is used to start the program or erase cycle.



Offset: 0x001C

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:4	*ONEUS*	R/W	0x4	Clock divider setup to divide system clock to tick a 1-us pulse, for example, if your system clock is 48MHz(20.8ns), then you have to set the register to 48 in decimal. About the role of SC_NVMACR. ONEUS, mainly for when Embedded Flash Memory in Program or Erase process. The controller can based on this setting, to do the Flash programing or erasing.
3:1	*WCYCLE*	R/W	0x0	Waiting cycle to read data from embedded flash **HCLK Frequency (*ƒ*)** / ***WCYCLE* Value** *ƒ* ≤ 24MHz / 0x0 24MHz < *ƒ* ≤ 48 MHz / 0x1 48 MHz < *ƒ* ≤ 72 MHz / 0x2 *Note: The highest frequency of HCLK in PT32C302 is 72 MHz, therefore, the value of WCYCLE should not be larger than 0x2.*
0	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.4.2.8 SC_GCLK_APB - APB PERIPHERAL CLOCK GATING ENABLE REGISTER

The SC_GCK_APB register enables the clocks of individual APB peripheral blocks;

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RO	RO	RO	RO	R/W	R/W	RO	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
				DAC	COMP		ADC			TIM5	TIM4	TIM3	TIM2	TIM1	TIM0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RO	RO	RO	RO	R/W	RO	R/W	R/W	R/W	R/W	RO	R/W	RO	R/W	R/W	R/W
				WDT		I2S1	I2S0	I2C1	I2C0		SPI		UART2	UART1	UART0

Offset: 0x0020

Bits	Name	Type	Reset	Description
31:28	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
27	*DAC*	R/W	0	DAC Clock Gating Control. This bit controls the clock gating for DAC module.
26	*COMP*	R/W	0	Comparator Clock Gating Control. This bit controls the clock gating for Comparator module.
25	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
24	*ADC*	R/W	0	ADC Clock Gating Control. This bit controls the clock gating for ADC module.
23:22	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
21	*TM5*	R/W	0	Timer 5 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 5.
20	*TM4*	R/W	0	Timer 4 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 4.
19	*TM3*	R/W	0	Timer 3 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 3.
18	*TM2*	R/W	0	Timer 2 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 2.
17	*TM1*	R/W	0	Timer 1 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 1.
16	*TM0*	R/W	0	Timer 0 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 0.
15:12	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	*WDT*	R/W	0	WDT Clock Gating Control. This bit controls the clock gating for WDT module.
10	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9	*SPIS1*	R/W	0	SPI2/I2S1 Clock Gating Control This bit controls the clock gating for SPI module 2 and I2S1 module.

8	*SPIS0*	R/W	0	SPI1/I2S0 Clock Gating Control This bit controls the clock gating for SPI module 1 and I2S0 module.
7	*I2C1*	R/W	0	I2C1 Clock Gating Control This bit controls the clock gating for I2C module 1.
6	*I2C0*	R/W	0	I2C0 Clock Gating Control This bit controls the clock gating for I2C module 0.
5	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*SPI0*	R/W	0	SPI0 Clock Gating Control This bit controls the clock gating for SPI module 0.
3	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*UART2*	R/W	0	UART2 Clock Gating Control This bit controls the clock gating for UART module 2.
1	*UART1*	R/W	0	UART1 Cloczk Gating Control This bit controls the clock gating for UART module 1.
0	*UART0*	R/W	0	UART0 Clock Gating Control This bit controls the clock gating for UART module 0.

4.4.2.9 SC_GCLK_AHB - AHB PERIPHERAL CLOCK GATING ENABLE REGISTER

The SC_GCK_AHB register enables the clocks to individual peripheral blocks which connect to AHB Bridge.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R/W	R/W	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	R/W	RO	RO	RO	RO
RTC	EBI				HDIV	USBPHY	USBCTRL				DMA				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
												GPIOD	GPIOC	GPIOB	GPIOA

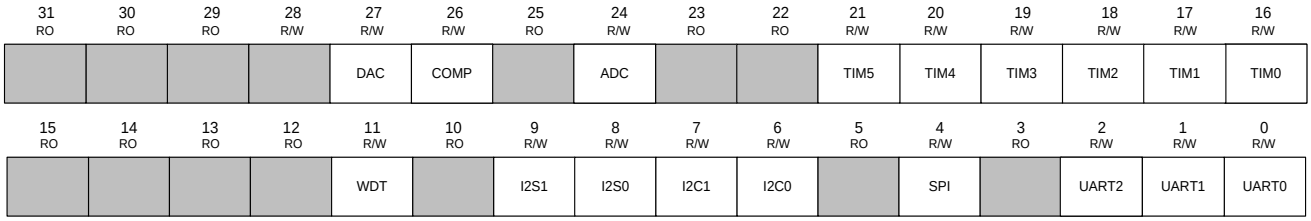
Offset: 0x0024

Bits	Name	Type	Reset	Description
31	*RTC*	R/W	0	Real-Time Clock (RTC) Clock Gating Control. This bit controls the clock gating for RTC module.
30	*EBI*	R/W	0	External Bus Interface (EBI) Clock Gating Control. This bit controls the clock gating for EBI module.
29:27	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26	*HDIV*	R/W	0	HDIV Clock Gating Control. This bit controls the clock gating for HDVI module.
25	*USBPHY*	R/W	0	USB PHY Clock Gating Control. This bit controls the clock gating for USB PHY module.
24	*USBCTRL*	R/W	0	USB Clock Gating Control. This bit controls the clock gating for USB module.
23:21	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
20	*DMA*	R/W	0	DMA Clock Gating Control. This bit controls the clock gating for DMA module.
19:4	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*GPIOD*	R/W	0	GPIO_D Port Clock Gating Control. This bit controls the clock gating for GPIO_D Port module.
2	*GPIOC*	R/W	0	GPIO_C Port Clock Gating Control. This bit controls the clock gating for GPIO_C Port module.
1	*GPIOB*	R/W	0	GPIO_B Port Clock Gating Control. This bit controls the clock gating for GPIO_B Port module.
0	*GPIOA*	R/W	0	GPIO_A Port Clock Gating Control. This bit controls the clock gating for GPIO_A Port module.
6	*I2C*	R/W	0	I2C0 Clock Gating Control This bit controls the clock gating for I2C module 0.
5	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*SPI*	R/W	0	SPI Clock Gating Control This bit controls the clock gating for SPI.

Bits	Name	Type	Reset	Description
3:2	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*UART1*	R/W	0	UART1 Clock Gating Control This bit controls the clock gating for UART module 1.
0	*UART0*	R/W	0	UART0 Clock Gating Control This bit controls the clock gating for UART module 0.

4.4.2.10 SC_RST_APB - APB PERIPHERAL RESET REQUEST REGISTER

This register allows software to reset the APB peripherals. Writing a '1' to assert the resets and writing a 0 to de-asserts

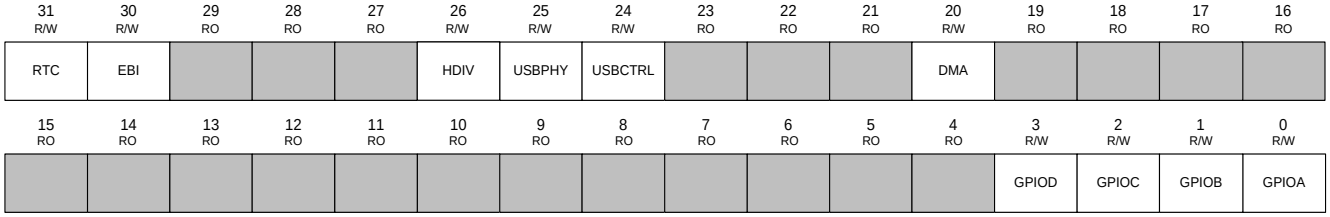


Offset: 0x0028

Bits	Name	Type	Reset	Description
31:28	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
27	*DAC*	R/W	0	DAC Reset Request 0: Manually clear after being set 1: Assert a reset signal to DAC
26	*COMP*	R/W	0	Comparator Reset Request 0: Manually clear after being set 1: Assert a reset signal to Analog Comparator
25	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
24	*ADC*	R/W	0	ADC Reset Request 0: Manually clear after being set 1: Assert a reset signal to ADC
23:22	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
21	*TM5*	R/W	0	Timer 5 Reset Request 0: Manually clear after being set 1: Assert a reset signal to TM5
20	*TM4*	R/W	0	Timer 4 Reset Request
19	*TM3*	R/W	0	Timer 3 Reset Request
18	*TM2*	R/W	0	Timer 2 Reset Request
17	*TM1*	R/W	0	Timer 1 Reset Request
16	*TM0*	R/W	0	Timer 0 Reset Request
15:12	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

11	*WDT*	R/W	0	WDT Reset Request. 0: Manually clear after being set 1: Assert a reset signal to WDT
10	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9	*I2S1*	R/W	0	I2S1 Reset Request. 0: Manually clear after being set 1: Assert a reset signal to I2S1
8	*I2S0*	R/W	0	I2S0 Reset Request. 0: Manually clear after being set 1: Assert a reset signal to I2S0
7	*I2C1*	R/W	0	I2C1 Reset Request. 0: Manually clear after being set 1: Assert a reset signal to I2C1
6	*I2C0*	R/W	0	I2C0 Reset Request. 0: Manually clear after being set 1: Assert a reset signal to I2C0
5	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*SPI*	R/W	0	SPI Reset Request. 0: Manually clear after being set 1: Assert a reset signal to SPI
3	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*UART2*	R/W	0	UART2 Reset Request. 0: Manually clear after being set 1: Assert a reset signal to UART2
1	*UART1*	R/W	0	UART1 Reset Request. 0: Manually clear after being set 1: Assert a reset signal to UART1
0	*UART0*	R/W	0	UART0 Reset Request 0: Manually clear after being set 1: Assert a reset signal to UART0.

4.4.2.11 SC_RST_AHB - AHB PERIPHERAL RESET REQUEST REGISTER

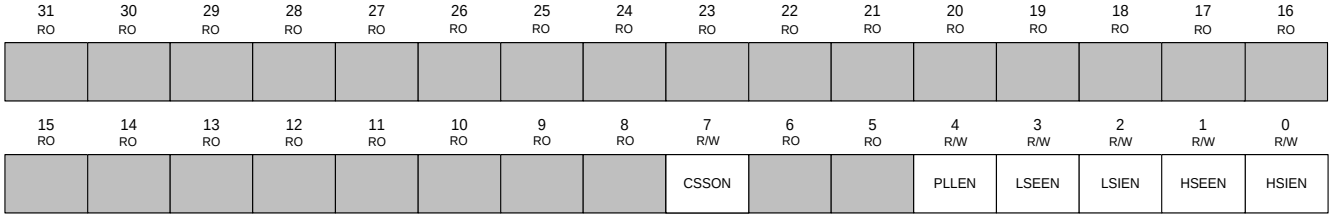


Offset: 0x002C

Bits	Name	Type	Reset	Description
31	*RTC*	R/W	0	RTC Request 0: Manually clear after being set 1: Assert a reset signal to RTC
30	*EBI*	R/W	0	EBI Request 0: Manually clear after being set 1: Assert a reset signal to EBI
29:27	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26	*HDIV*	R/W	0	
25	*USBPHY*	R/W	0	USB PHY Request 0: Manually clear after being set 1: Assert a reset signal to USBPHY
24	*USBCTRL*	R/W	0	USB Reset Request 0: Manually clear after being set 1: Assert a reset signal to USB
23:21	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
20	*DMA*	R/W	0	DMA Reset Request 0: Manually clear after being set 1: Assert a reset signal to DMA
19:4	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*GPIOD*	R/W	0	GPIO Port D Reset Request 0: Manually clear after being set 1: Assert a reset signal to GPIOD
2	*GPIOC*	R/W	0	GPIO Port C Reset Request. 0: Manually clear after being set 1: Assert a reset signal to GPIOC.
1	*GPIOB*	R/W	0	GPIO Port B Reset Request 0: Manually clear after being set 1: Assert a reset signal to GPIOB.
0	*GPIOA*	R/W	0	GPIO Port A Reset Request 0: Manually clear after being set 1: Assert a reset signal to GPIOA
6	*I2C*	R/W	0	I2C Reset Request. 0: Manually clear after being set 1: Assert a reset signal to I2C0

Bits	Name	Type	Reset	Description
5	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*SPI*	R/W	0	SPI Reset Request. 0: Manually clear after being set 1: Assert a reset signal to SPI
3:2	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*UART1*	R/W	0	UART1 Reset Request. 0: Manually clear after being set 1: Assert a reset signal to UART1
0	*UART0*	R/W	0	UART0 Reset Request 0: Manually clear after being set 1: Assert a reset signal to UART0.

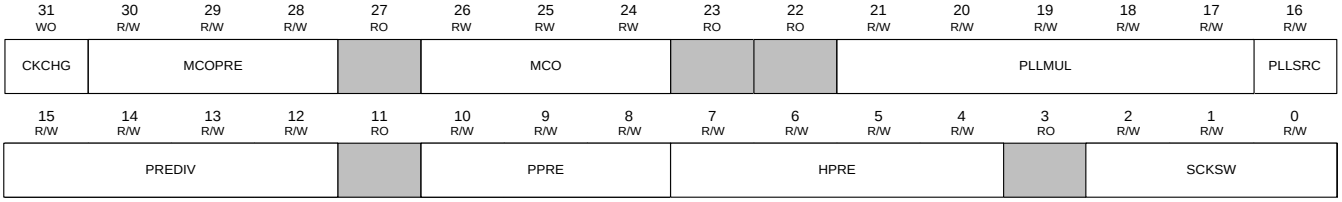
4.4.2.12 SC_CK_CTRL - CLOCK CONTROL REGISTER



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	*CSSON*	R/W	0	External crystal oscillator clock security sensor (CSS) Enable 0: CSS OFF 1: CSS ON
6:5	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*PLLEN*	R/W	0	PLL Enable 0: PLL OFF 1: PLL ON
3	*LSEEN*	R/W	0	LSE Clock Enable 0: LSE OFF 1: LSE ON
2	*LSIEN*	R/W	0	LSI Clock Enable 0: LSI OFF 1: LSI ON
1	*HSEEN*	R/W	0	HSE Clock Enable 0: HSE OFF 1: HSE ON
0	*HSIEN*	R/W	1	HSI Clock Enable 0: HSI OFF 1: HSI ON

4.4.2.13 SC_CK_CONF - CLOCK CONFIGURATION REGISTER

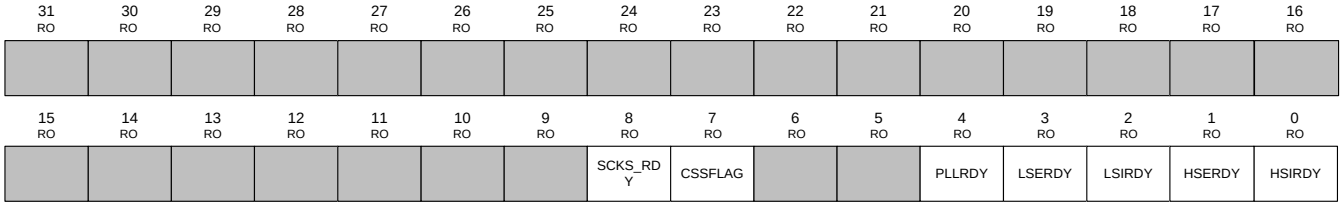


Offset: 0x0034

Bits	Name	Type	Reset	Description
31	*CKCHG*	WO	0	Change Clock Configuration
30:28	*MCOPRE*	R/W	0x0	Microcontroller Clock Output Divider (1~8) Output Frequency = MCO/(MCOPRE+1)
27	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26:24	*MCO*	R/W	0x0	Microcontroller Clock Output 0x0 : MCO output disabled, no clock on MCO 0x1 : Internal low speed (LSI) oscillator clock selected 0x2 : External low speed (LSE) oscillator clock selected 0x3 : Internal RC 4 MHz (HSI) oscillator clock selected 0x4 : External 4-36 MHz (HSE) oscillator clock selected 0x5 : System clock selected 0x6 : AHB clock selected 0x7 : APB clock selected
23:22	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
21:17	*PLLMUL*	R/W	0x0	PLL Multiplication Factor These bits can be written only when PLL is disabled. PLL clock output frequency is PLLSRC x PLLMUL[4:0] MHz Caution: The PLL output frequency must not exceed 48 MHz. 0 : External /Internal oscillator 4M (PLLSRC) 1 : PLL Input clock source x 1 2 : PLL Input clock source x 2 3 : PLL Input clock source x 3… 12 : PLL Input clock source x 12
16	*PLLSRC*	R/W	0	PLL Input Clock Source These bits can be written only when PLL is disabled. 0: HSI is selected as PLL input clock 1: HSE/PREDIV is selected as PLL input clock
15:12	*PREDIV*	R/W	0x0	HSE clock PREDIV division factor for PLL Reference Clock 4MHz These bits are set and cleared by software to select PREDIV division factor. PLL_reference_clock (4MHz) = HSE / (PREDIV+1) Example : HSE(20MHz) / PREDIV(4)+1 = 4MHz
11	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

Bits	Name	Type	Reset	Description
10:8	*PPRE*	R/W	0x0	PCLK Prescaler These bits control the division factor of the APB clock (PCLK). 0xX : HCLK is not divided 0x4 : HCLK is divided by 2 0x5 : HCLK is divided by 4 0x6 : HCLK is divided by 8 0x7 : HCLK is divided by 16
7:4	*HPRE*	R/W	0x0	HCLK Prescaler These bits control the division factor of the AHB clock. <0x8 : SYSCLK is not divided 0x8 : SYSCLK is divided by 2 0x9 : SYSCLK is divided by 4 0xA : SYSCLK is divided by 8 0xB : SYSCLK is divided by 16 >0xB : Reserved
3	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*SCKSW*	R/W	0x0	System clock switch(SYSCLK) 0x0 : HSI selected as system clock 0x1 : HSE selected as system clock 0x2 : PLL selected as system clock 0x3 : LSI selected as system clock 0x4 : LSE selected as system clock

4.4.2.14 SC_CK_STAT - CLOCK SOURCE STATUS REGISTER

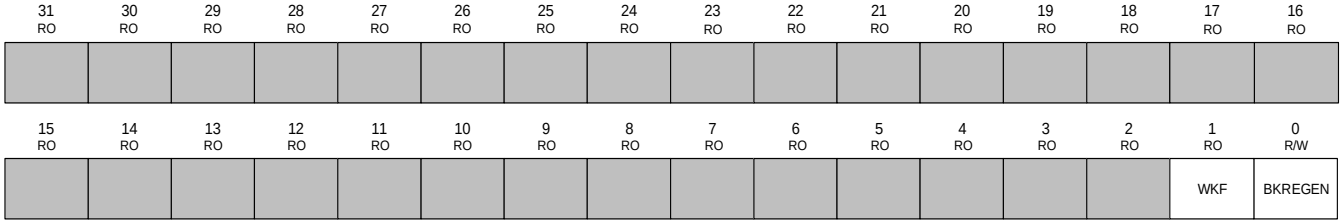


Offset: 0x003C

Bits	Name	Type	Reset	Description
31:9	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
8	*SCKS_RDY*	RO	0	System Clock Switch Ready Flag 0: System Clock Switch is not yet ready 1: System Clock Switch is ready
7	*CSSFLAG*	RO	0	External crystal oscillator clock security sensor (CSS) Ready Flag 0: CSS is not ready 1: CSS is ready
6:5	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*PLLRDY*	RO	0	PLL Clock Ready Flag 0: PLL Clock is not ready 1: PLL Clock is ready
3	*LSERDY*	RO	0	LSE Clock Ready Flag 0: LSE Clock is not ready 1: LSE Clock is ready
2	*LSIRDY*	RO	0	LSI Clock Ready Flag 0: LSI Clock is not ready 1: LSI Clock is ready
1	*HSERDY*	RO	0	HSE Clock Ready Flag 0: HSE Clock is not ready 1: HSE Clock is ready
0	*HSIRDY*	RO	0	HSI Clock Ready Flag 0: HSI Clock is not ready 1: HSI Clock is ready
11	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10:8	*PPRE*	R/W	0x0	PCLK Prescaler These bits control the division factor of the APB clock (PCLK). 0xX : HCLK is not divided 0x4 : HCLK is divided by 2 0x5 : HCLK is divided by 4 0x6 : HCLK is divided by 8 0x7 : HCLK is divided by 16
7:4	*HPRE*	R/W	0x0	HCLK Prescaler These bits control the division factor of the AHB clock. <0x8 : SYSCLK is not divided

Bits	Name	Type	Reset	Description
				0x8 : SYSCLK is divided by 2 0x9 : SYSCLK is divided by 4 0xA : SYSCLK is divided by 8 0xB : SYSCLK is divided by 16 >0xB : Reserved
3:2	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1:0	*SCKSW*	R/W	0x0	System Clock Switch 0x0 : HSI is selected as system clock 0x1 : HSE/PREDIV is selected as system clock 0x2 : PLL is selected as system clock 0x3 : Reserved

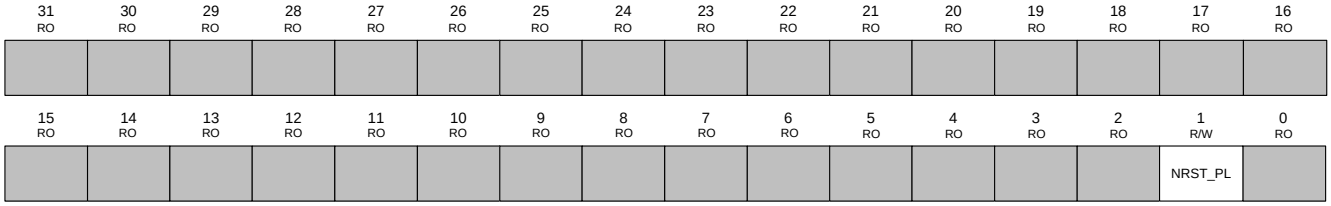
4.4.2.15 SC_BK_CTRL - BACKUP CONTROL REGISTER



Offset: 0x0040

Bits	Name	Type	Reset	Description
31:2	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*WKF*	RO	0	Low Power WakeUp Flag
0	*BKREGEN*	R/W	0	Access Backup Register Enable. After reset, access to the Backup register is disabled and the Backup domain is protected. To enable access to backup register, this bit is needed to be set. 0: Disable access for backup register 1: Enable access for backup register

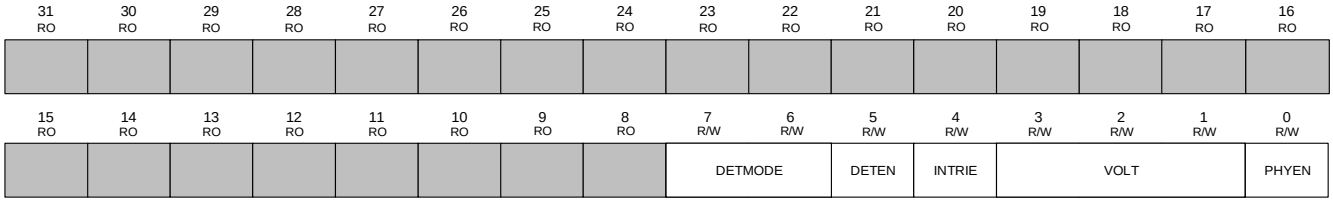
4.4.2.16 SC_PWR_CTRL - POWER CONTROL REGISTER



Offset: 0x0048

Bits	Name	Type	Reset	Description
31:2	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*NRST_PL*	R/W	0	NRST Pull Low, active high 0: No function 1: Pull Low
0	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

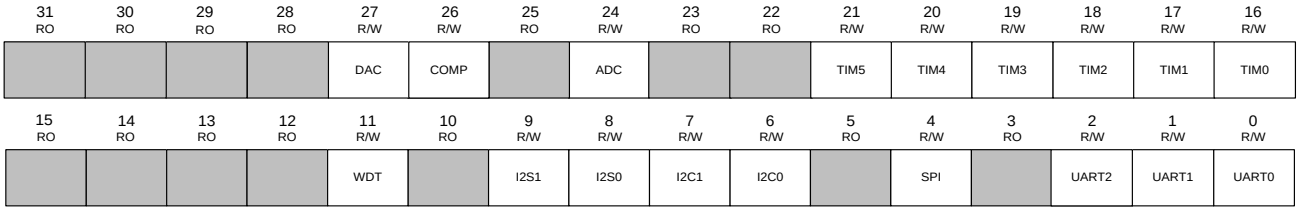
4.4.2.17 SC_PVD_DET - PVD DETECTION REGISTER



Offset: 0x004C

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	*DETMODE*	R/W	0x0	PVD Detection mode setting, 0x0: Interrupt 0x1: Gated Clock 0x2: Reset Request
5	*DETEN*	R/W	0	PVD Detection Enable. 0: PVD Detection is disabled 1: PVD Detection is enabled
4	*INTRIE*	R/W	0	PVD Detection Interrupt Enable 0: PVD Detection interrupt is disabled 1: PVD Detection interrupt is enabled
3:1	*VOLT*	R/W	0x7	PVD voltage select 0x0 : Rising - 2.2V, Falling - 2.1V 0x1 : Rising - 2.3V, Falling - 2.2V 0x2 : Rising - 2.4V, Falling - 2.3V 0x3 : Rising - 2.5V, Falling - 2.4V 0x4 : Rising - 2.6V, Falling - 2.5V 0x5 : Rising - 2.7V, Falling - 2.6V 0x6 : Rising - 2.8V, Falling - 2.7V 0x7 : Rising - 2.9V, Falling - 2.8V
0	*PHYEN*	R/W	0	PVD Detector Circuit Enable, active high

4.4.2.18 SC_SLP_APB - INTO SLEEP APB PERIPHERAL CLOCK GATING ENABLE REGISTER

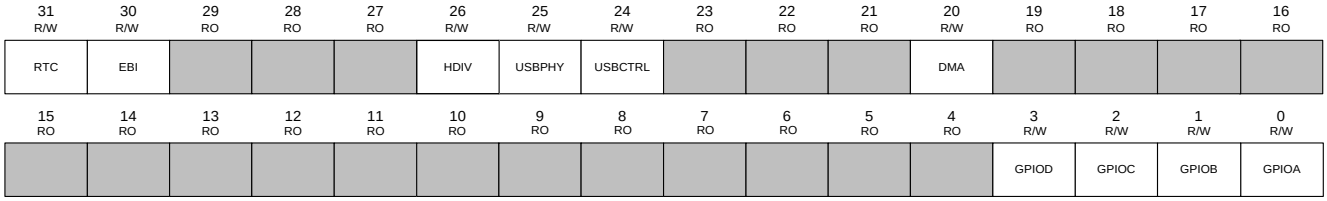


Offset: 0x0050

Bits	Name	Type	Reset	Description
31:28	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
27	*DAC*	R/W	0	Sleep Mode DAC Clock Gating Control. This bit controls the clock gating for DAC module in the sleep mode.
26	*COMP*	R/W	0	Sleep Mode Comparator Clock Gating Control. This bit controls the clock gating for Comparator module in the sleep mode.
25	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
24	*ADC*	R/W	0	Sleep Mode ADC Clock Gating Control. This bit controls the clock gating for ADC module in the sleep mode.
23:22	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
21	*TM5*	R/W	0	Sleep Mode Timer 5 Clock Gating Control. This bit controls the clock gating for Advence Timer module 5 in the sleep mode.
20	*TM4*	R/W	0	Sleep Mode Timer 4 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 4 in the sleep mode.
19	*TM3*	R/W	0	Sleep Mode Timer 3 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 3 in the sleep mode.
18	*TM2*	R/W	0	Sleep Mode Timer 2 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 2 in the sleep mode.
17	*TM1*	R/W	0	Sleep Mode Timer 1 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 1 in the sleep mode.
16	*TM0*	R/W	0	Sleep Mode Timer 0 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 0 in the sleep mode.
15:12	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	*WDT*	R/W	0	Sleep Mode WDT Clock Gating Control. This bit controls the clock gating for WDT module in the sleep mode.

10	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9	*I2S1*	R/W	0	Sleep Mode I2S1 Clock Gating Control. This bit controls the clock gating for I2S module 1 in the sleep mode.
8	*I2S0*	R/W	0	Sleep Mode I2S0 Clock Gating Control. This bit controls the clock gating for I2S module 0 in the sleep mode.
7	*I2C1*	R/W	0	Sleep Mode I2C1 Clock Gating Control This bit controls the clock gating for I2C module 1 in the sleep mode.
6	*I2C0*	R/W	0	Sleep Mode I2C0 Clock Gating Control This bit controls the clock gating for I2C module 0 in the sleep mode.
5	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*SPI*	R/W	0	Sleep Mode SPI Clock Gating Control This bit controls the clock gating for SPI module in the sleep mode.
3	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*UART2*	R/W	0	Sleep Mode UART2 Clock Gating Control This bit controls the clock gating for UART module 2 in the sleep mode.
1	*UART1*	R/W	0	Sleep Mode UART1 Clock Gating Control This bit controls the clock gating for UART module 1 in the sleep mode.
0	*UART0*	R/W	0	Sleep Mode UART0 Clock Gating Control This bit controls the clock gating for UART module 0 in the sleep mode.

4.4.2.19 SC_SLP_AHB - INTO SLEEP AHB PERIPHERAL CLOCK GATING ENABLE REGISTER

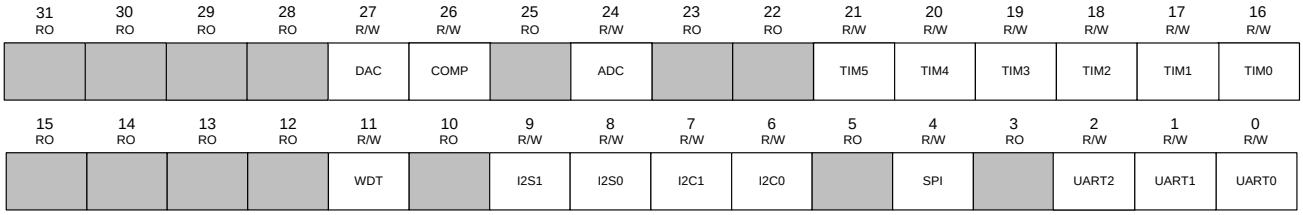


Offset: 0x0054

Bits	Name	Type	Reset	Description
31	*RTC*	R/W	0	Sleep Mode RTC Gating Control This bit controls the clock gating for RTC module in the sleep mode.
30	*EBI*	R/W	0	Sleep Mode EBI Gating Control This bit controls the clock gating for EBI module in the sleep mode.
29:27	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26	*HDIV*	R/W	0	Sleep Mode HDIV Clock Gating Control. This bit controls the clock gating for HDIV module in the sleep mode.
25	*USBPHY*	R/W	0	Sleep Mode USB PHY Gating Control This bit controls the clock gating for USB PHY module in the sleep mode.
24	*USBCTRL*	R/W	0	Sleep Mode USB Gating Control This bit controls the clock gating for USB module in the sleep mode.
23:21	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
20	*DMA*	R/W	0	Sleep Mode DMA Gating Control This bit controls the clock gating for DMA module in the sleep mode.
19:4	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*GPIOD*	R/W	0	Sleep Mode GPIO_D Port Clock Gating Control. This bit controls the clock gating for GPIO_D Port module in the sleep mode.
2	*GPIOC*	R/W	0	Sleep Mode GPIO_C Port Clock Gating Control. This bit controls the clock gating for GPIO_C Port module in the sleep mode.
1	*GPIOB*	R/W	0	Sleep Mode GPIO_B Port Clock Gating Control. This bit controls the clock gating for GPIO_B Port module in the sleep mode.
0	*GPIOA*	R/W	0	Sleep Mode GPIO_A Port Clock Gating Control. This bit controls the clock gating for GPIO_A Port module in the sleep mode.
6	*I2C*	R/W	0	Sleep Mode I2C0 Clock Gating Control This bit controls the clock gating for I2C module 0 in the sleep mode.
5	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

Bits	Name	Type	Reset	Description
4	*SPI*	R/W	0	Sleep Mode SPI Clock Gating Control This bit controls the clock gating for SPI module 0 in the sleep mode.
3:2	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*UART1*	R/W	0	Sleep Mode UART1 Clock Gating Control This bit controls the clock gating for UART module 1 in the sleep mode.
0	*UART0*	R/W	0	Sleep Mode UART0 Clock Gating Control This bit controls the clock gating for UART module 0 in the sleep mode.

4.4.2.20 SC_DSLP_APB - INTO SLEEP DEEP APB PERIPHERAL CLOCK GATING ENABLE REGISTER

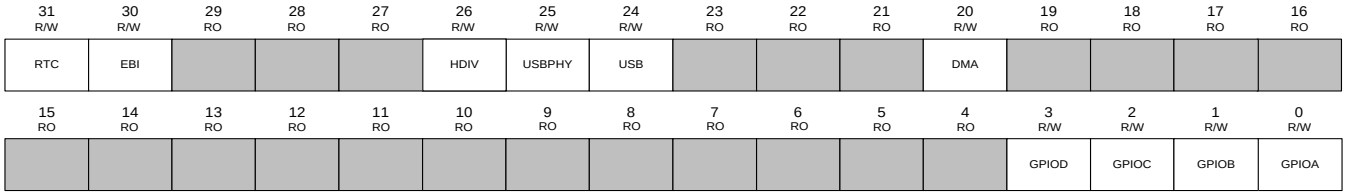


Offset: 0x0058

Bits	Name	Type	Reset	Description
31:28	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
27	*DAC*	R/W	0	Sleep Mode DAC Clock Gating Control. This bit controls the clock gating for DAC module in the deep sleep mode.
26	*COMP*	R/W	0	Deep Sleep Mode Comparator Clock Gating Control. This bit controls the clock gating for Comparator module in the deep sleep mode.
25	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
24	*ADC*	R/W	0	Deep Sleep Mode ADC Clock Gating Control. This bit controls the clock gating for ADC module in the deep sleep mode.
23:22	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
21	*TM5*	R/W	0	Deep Sleep Mode Timer 5 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 5 in the deep sleep mode.
20	*TM4*	R/W	0	Deep Sleep Mode Timer 4 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 4 in the deep sleep mode.
19	*TM3*	R/W	0	Deep Sleep Mode Timer 3 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 3 in the deep sleep mode.
18	*TM2*	R/W	0	Deep Sleep Mode Timer 2 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 2 in the deep sleep mode.
17	*TM1*	R/W	0	Deep Sleep Mode Timer 1 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 1 in the deep sleep mode.
16	*TM0*	R/W	0	Deep Sleep Mode Timer 0 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 0 in the deep sleep mode.
15:12	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

Bits	Name	Type	Reset	Description
11	*WDT*	R/W	0	Deep Sleep Mode WDT Clock Gating Control. This bit controls the clock gating for WDT module in the deep sleep mode.
10	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9	*I2S1*	R/W	0	Deep Sleep Mode I2S1 Clock Gating Control This bit controls the clock gating for I2S module 1 in the deep sleep mode.
8	*I2S0*	R/W	0	Deep Sleep Mode I2S0 Clock Gating Control This bit controls the clock gating for I2S module 0 in the deep sleep mode.
7	*I2C1*	R/W	0	Deep Sleep Mode I2C1 Clock Gating Control This bit controls the clock gating for I2C module 1 in the deep sleep mode.
6	*I2C0*	R/W	0	Deep Sleep Mode I2C0 Clock Gating Control This bit controls the clock gating for I2C module 0 in the deep sleep mode.
5	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*SPI*	R/W	0	Deep Sleep Mode SPI Clock Gating Control This bit controls the clock gating for SPI module in the deep sleep mode.
3	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*UART2*	R/W	0	Deep Sleep Mode UART2 Clock Gating Control This bit controls the clock gating for UART module 2 in the deep sleep mode.
1	*UART1*	R/W	0	Deep Sleep Mode UART1 Clock Gating Control This bit controls the clock gating for UART module 1 in the deep sleep mode.
0	*UART0*	R/W	0	Deep Sleep Mode UART0 Clock Gating Control This bit controls the clock gating for UART module 0 in the deep sleep mode.

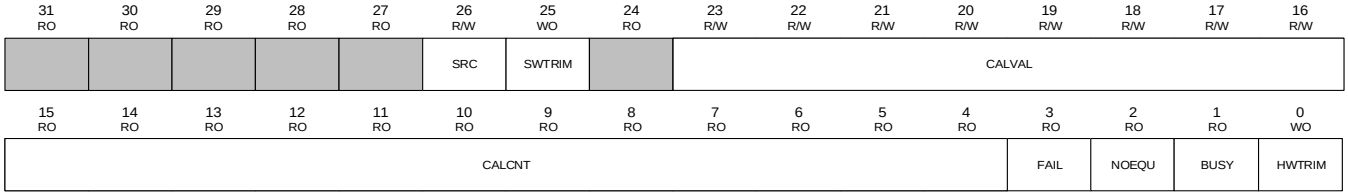
4.4.2.21 SC_DSLP_AHB - INTO DEEP SLEEP AHB PERIPHERAL CLOCK GATING ENABLE REGISTER



Offset: 0x005C

Bits	Name	Type	Reset	Description
31	*RTC*	R/W	0	Deep Sleep Mode RTC Gating Control This bit controls the clock gating for RTC module in the deep sleep mode.
30	*EBI*	R/W	0	Deep Sleep Mode EBI Gating Control This bit controls the clock gating for EBI module in the deep sleep mode.
29:27	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26	*HDIV*	R/W	0	Deep Sleep Mode HDIV Clock Gating Control. This bit controls the clock gating for HDIV module in the deep sleep mode.
25	*USBPHY*	R/W	0	Deep Sleep Mode USB PHY Gating Control This bit controls the clock gating for USB PHY module in the deep sleep mode.
24	*USB*	R/W	0	Deep Sleep Mode USB Gating Control This bit controls the clock gating for USB module in the deep sleep mode.
23:21	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
20	*DMA*	R/W	0	Deep Sleep Mode DMA Gating Control This bit controls the clock gating for DMA module in the deep sleep mode.
19:4	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*GPIOD*	R/W	0	Deep Sleep Mode GPIO_D Port Clock Gating Control. This bit controls the clock gating for GPIO_D Port module in the deep sleep mode.
2	*GPIOC*	R/W	0	Deep Sleep Mode GPIO_C Port Clock Gating Control. This bit controls the clock gating for GPIO_C Port module in the deep sleep mode.
1	*GPIOB*	R/W	0	Deep Sleep Mode GPIO_B Port Clock Gating Control. This bit controls the clock gating for GPIO_B Port module in the deep sleep mode.
0	*GPIOA*	R/W	0	Deep Sleep Mode GPIO_A Port Clock Gating Control. This bit controls the clock gating for GPIO_A Port module in the deep sleep mode.

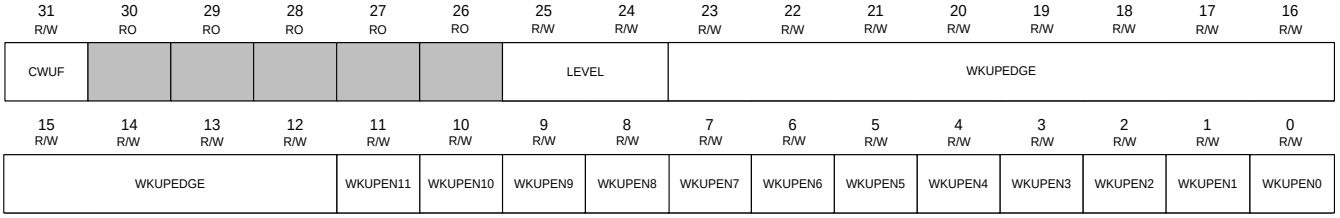
4.4.2.22 SC_OSC_TRIM - RING OSC HARDWARE TRIMMING REGISTER



Offset: 0x006C

Bits	Name	Type	Reset	Description
31:27	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26	*SRC*	R/W	0	Internal Oscillator 4M Clock Trimming Source 0: IO Port (PB02: 32.768KHz) 1: External crystal oscillator 32.768K Hz
25	*SWTRIM*	WO	0	HSI Manual Trimming Flow Control These bits provide an additional user-programmable trimming value to adjust to variations in voltage and temperature that influence the frequency of the HSI.
24	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:16	*CALVAL*	R/W	0x80	Calibration Result Value Note that this calibration result value would not be stored when system enters standby mode (level 2). User can store this result to *CALVAL* in Backup System Control Register (SC_BK_SCTRL) to avoid losing this calibrated result after system is waken up from standby mode.
15:4	*CALCNT*	RO	0x0	Trimming Counter Feedback Value
3	*FAIL*	RO	0	Flag For Indicating Trimming Fail 1: Circuit cannot detect any reference trimming source from IO port (PB02)
2	*NOEQU*	RO	0	Not Equal Flag 1: Circuit has detected a reference trimming source but the trimming failed as the trimming result was not equal to 4 MHz clock.
1	*BUSY*	RO	0	Busy Status 0: HSI finished trimming. 1: HSI is trimming. When this bit is 0, check *FAIL* and *NOEQU* if the trimming successes.
0	*HWTRIM*	WO	0	Trimming Start 1: When *SRC* = 0. Internal oscillator 4M starts auto trimming.

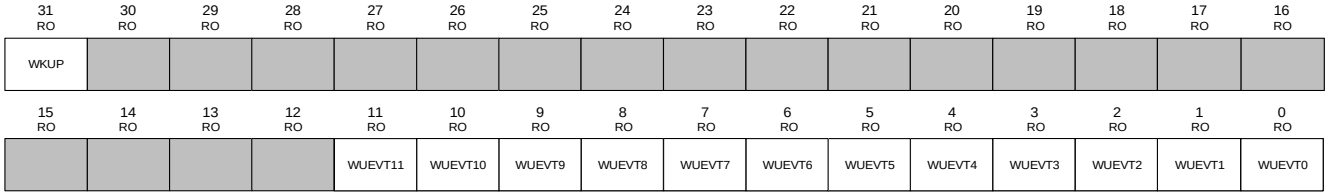
4.4.2.23 SC_BK_LPCTRL - BACKUP LOW POWER SLEEP CONTROLLER REGISTER



Offset: 0x0080

Bits	Name	Type	Reset	Description
31	*CLEAR*	WO	0	Clear Wake Flag, when system wakeup from Low Power Flow
30:26	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
25:24	*LEVEL*	R/W	0x0	Low Power Level Select 0 : Level 0, Sleep Mode (Normal) 1 : Level 1, Stop Mode (LDO into Low Power) 2 : Level 2, Standby Mode (LDO off) 3[A3] : Level 3, Reserved
23:12	*WKUPEDGE*	R/W	0xC00	Each of the wake up events edge sensitive type select 0 : Falling Edge-sensitive (default) 1 : Rising Edge-sensitive
11:0	*WKUPEN*	R/W	0x000	Enable the wake up events to wake up system Bit 0: PB02 Pin Bit 1: PB09 Pin Bit 2: PD04 Pin Bit 3: PD05 Pin Bit 4: PD06 Pin Bit 5: PD07 Pin Bit 6: PD08 Pin Bit 7: PD09 Pin Bit 8: PD10_WKUP Pin Bit 9: PD11_NRST Pin (only Falling) Bit 10: RTC Wakeup Counter (only Rising) Bit 11: Internal PVD Flag (only Rising)
15:4	*CALCNT*	RO	0x0	Trimming Counter Feedback Value
3	*FAIL*	RO	0	Flag For Indicating Trimming Fail 1: Circuit cannot detect any reference trimming source from IO port (PB02)
2	*NOEQU*	RO	0	Not Equal Flag 1: Circuit has detected a reference trimming source but the trimming failed as the trimming result was not equal to 4 MHz clock.
1	*BUSY*	RO	0	Busy Status 0: HSI finished trimming. 1: HSI is trimming. When this bit is 0, check *FAIL* and *NOEQU* if the trimming successes.
0	*HWTRIM*	WO	0	Trimming Start 1: When *SRC* = 0. Internal oscillator 4M starts auto trimming.

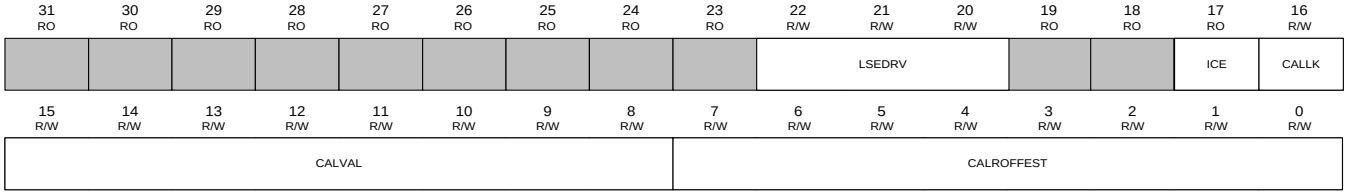
4.4.2.24 SC_BK_STATUS - BACKUP STATUS REGISTER



Offset: 0x0084

Bits	Name	Type	Reset	Description
31	*WKUP*	RO	0	Wakeup Flag This Flag indicates the system is waken up from RTC or a wake-up event.
30:12	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11:0	*WUEVTx*	RO	0x0	Wakeup Pin x Event Status Bit 0: PB02 pin Bit 1: PB09 pin Bit 2: PD04 pin Bit 3: PD05 pin Bit 4: PD06 pin Bit 5: PD07 pin Bit 6: PD08 pin Bit 7: PD09 pin Bit 8: PD10 pin Bit 9: PD11_NRST pin Bit 10: RTC wakeup pin Bit 11: Internal PVD flag

4.4.2.25 SC_BK_SCTRL - BACKUP SYSTEM CONTROL REGISTER

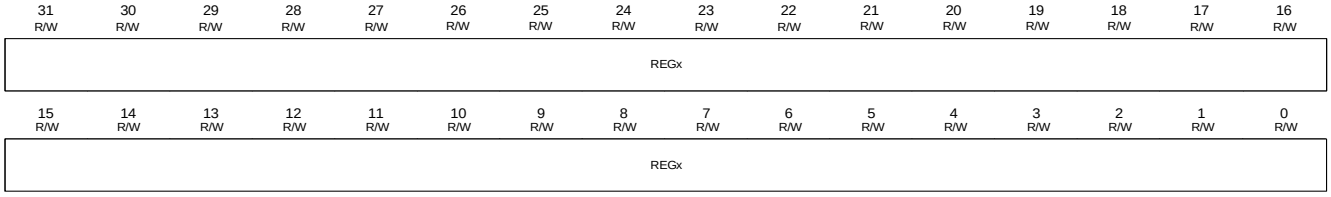


Offset: 0x0088

Bits	Name	Type	Reset	Description
31:23	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
22:20	*LSEDRV*	R/W	0x1	LSE oscillator drive capability 00: Lower driving capability 01: Medium low driving capability 10: Medium high driving capability 11: Higher driving capability
19:18	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
17	*ICE*	R/W	0	ICE Protection 0: Disable ICE protect 1: Enable ICE protect (ICE is temporarily disabled)
16	*CALLK*	R/W	0	Calibration Lock 1: To lock the value written in the "*CALRVAL*" field.
15:8	*CALVAL*	R/W	0x80	LSI clock calibration result Write the calibrated result obtained from *CALVAL* in SC_OSC_TRIM to this field can preserve the result from losing in standby mode. Use *CALLK* to lock this value and prevent it from being modified. System will use this value after it is waken up from standby mode.
7:0	*CALROFFEST*	R/W	0x00	LSI clock calibration offest
24	*ALRMMS*	R/W	0	Backup RTC sleep alarm counter set value for second or minute
23:0	*ALRMVAL*	WO	0x1FFFF	Backup RTC sleep alarm counter set value

4.4.2.26 SC_BK_REGX - BACKUP X REGISTER

A total of eight 32-bit backup registers locate from 0x0090 to 0X00AC. The value of x is from 0 to 7.



Offset: 0x0090 - 0x00AC

Bits	Name	Type	Reset	Description
31:0	*REGx*	R/W	0x0	User define field
24	*WKUPFLAG*	RO	0	Wakeup Flag This Flag indicates the system is waken up from RTC or a wake-up event. 0: Waken up from RTC 1: Waken up from Low-power flow
23:0	*SLCNT*	RO	0x0	Backup RTC sleep counter value

4.5 FLASH CONTROLLER (FC)

The PT32C302 embedded flash has 4kB Information Block and 64kB embedded flash memory that can be updated through ISP procedure. The In-System-Programming (ISP) function enables the user to update the program code while the chip is soldered on the PCB. After the chip is powered on, the Cortex™-M0 CPU fetches the program code from flash memory.

- The embedded flash can operate up to 24 MHz with zero wait state (and 48MHz with one wait state) for continuous read access in typical case.
- All embedded flash memory supports 512 bytes page erase.
- The embedded flash supports In-Application-Programming.
- 32-bit word programming

4.5.1 MEMORY ORGANIZATION

The Embedded Flash Controller is composed of two sections

- Control section (0x50010000 - 0x5001FFFF) is used to configure the controller such as clock frequency, read cycle delay, programming cycle information.
- Storage section (0x08000000 - 0x0800FFFF / 0x1FFFF000 - 0x1FFFFFFF), the address range depends on the remap register setting. This section is used to read data from flash macro, flash model activities such as reading, programming, page erasing and mass erasing of information block or main block.

4.5.2 FLASH CONTROLLER CLOCKING FREQUENCY

When the system clock is changed, the sequence of configuring the wait cycles of the embedded flash memory controller differs depending on whether the clock is changed from low to high speed or high to low speed. For low to high speed the wait cycle in **SC_NVMACR** register is first loaded before changing the PLL frequency. For high to low speed the sequence is reversed, that is, the PLL frequency is first modified followed by wait cycle setup.

Wait cycles are added whenever the operating frequency is faster than the flash memory read access time. For operating frequency that is under 24 MHz, no wait cycle need to be added; for operating frequency between 24 MHz and 48 MHz, then 1 wait cycle is added in order to reach the appropriate read access time. Set 1 to "*WCYCLE*" field of **SC_NVMACR** register.

4.5.3 MAIN FLASH MEMORY REMAP

Once the user select to start the system booting progress at Eflash, user can further by setting Memory remap by setting *EFBASE* bit in System control - software configuration register **(SC_REMAP)** to select each 4k Byte of flash main block as base address.

Once the Eflash boot mode is selected (refer to Booting Configuration), user can further modify the starting base address of main flash memory field (by writing different value in *EFBASE* field). The configuration of main flash memory remapping is as follows:

1. Write 0x2 to REMAP field in the SC_REMAP register.
2. Determine the accessible base address of main flash memory. In PT32C302, the accessible location of main flash
3. memory is based on the multiple of 4k byte.

After configuring the previous steps, then configurate CPU Reset.

For example, if the Main Flash is aliased in the boot memory space (0x0000 0000):

- Write 0x0 to *EFBASE*¸ then the accessing is starting on 0x0000 0000
- Write 0x1 to *EFBASE*, then the accessing is starting on 0x0000 1000
- Write 0x2 to *EFBASE*, then the accessing is starting on 0x0000 2000

And so on…For more detailed, refer to **SC_REMAP** register's *EFBASE* bit.

The booting configuration in in system control is for reading operation only, writing operation depends on the address that user writes to the **FC_PADDR** register.

4.5.4 FLASH PROGRAMMING/ERASING

All erase/program operations are handles via three registers: FC_CMD, FC_PDATA, and FC_PADDR. During a Flash memory operation (write, page erase, or mass erase) access to the Flash memory is inhibited. As a result, instruction and literal fetches are held off until the Flash memory operation is complete. If instruction execution is required during a Flash memory operation, the code that is executing must be placed in SRAM and executed from there while the flash operation is in progress.

Programming Procedure:

1. Write source data to the FC_PDATA register.
2. Write the target address to the FC_PADDR register.
3. Write the PGCMD bit to the FC_CMD register.
4. Poll the FC_CMD register until the PGCMD bit is automatically cleared.

Page Erasing Procedure:

1. Write the page address to the FC_PADDR register.
2. Write the PGERASE bit to the FC_CMD register.
3. Poll the FC_CMD register until the PGERASE bit is automatically cleared.

Mass Erasing Procedure:

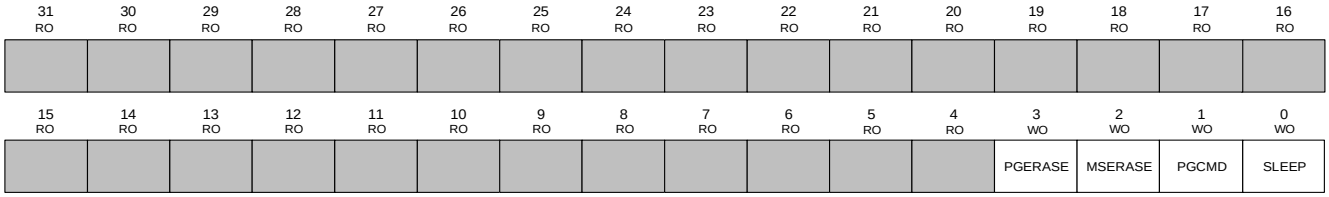
1. Write the MASERASE bit to the FC_CMD register.
2. Poll the FC_CMD register until the MASERASE bit is automatically cleared

4.5.5 FLASH CONTROLLER REGISTER MAP

Base Address: 0x5001_0000

Offset	Symbol	Type	Reset Value	Description	See page
reserved	0x0000	N/A	-	Reserved	-
0x0004	FC_CMD	WO	0x0000_0000	Flash Command Register	85
0x0008	FC_PDATA	R/W	0x0000_0000	Flash Program Data Register	86
reserved	0x0000	N/A	-	Reserved	-
0x0010	FC_PADDR	R/W	0x0000_0000	Flash Program Address Register	86

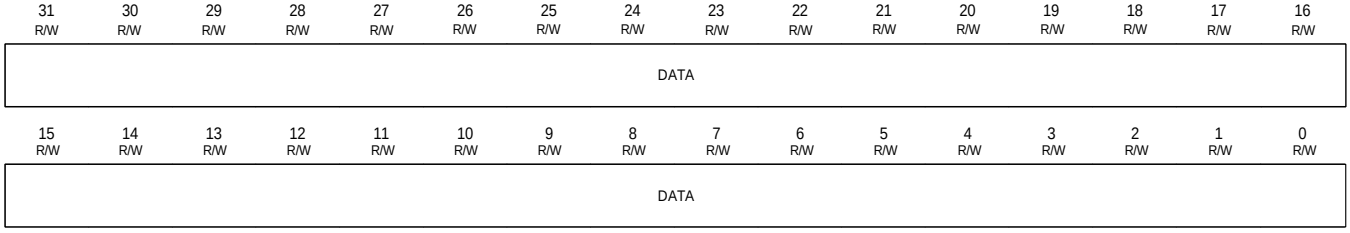
4.5.5.1 FC_CMD - EMBEDDED COMMAND REGISTER



Offset: 0x0004

Bits	Name	Type	Reset	Description
Bits	Name	Type	Reset	Description
31:4	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*PERCMD*	WO	0	Page Erase cycle This bit is used to erase a page of Flash main memory bit and it will be auto-cleared after the operation flow is done. The minimum time of erase time is 40ms. 1: Set this bit to erase the flash memory page specified by the FC_PADDR.
2	*MERCMD*	WO	0	Mass Erase cycle This bit is used to mass erase the Flash main memory and it will be auto-cleared after the operation flow is done. The minimum time of erase time is 40ms. 1: Set this bit to erase the Flash main memory.
1	*PGCMD*	WO	0	Program cycle. 1: The data stored in FC_PDATA is written into the location as specified by the contents of FC_PADDR. The takes 30-40μs.
0	*SLEEP*	R/W	0	Flash Memory Sleep Mode Enable 1: The embedded flash will be put into sleep mode when the device enters low power mode.

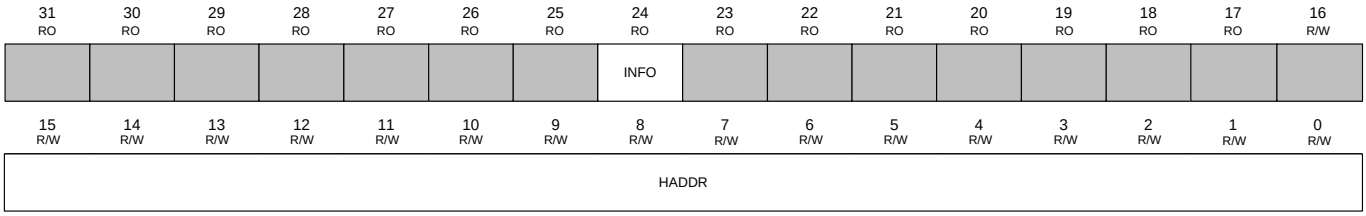
4.5.5.2 FC_PDATA - FLASH CONTROLLER PROGRAM DATA REGISTER



Offset: 0x0008

Bits	Name	Type	Reset	Description
31:0	*DATA*	R/W	0x0	Embedded Program 32bits Data Register

4.5.5.3 FC_PADDR - FLASH CONTROLLER PROGRAM ADDRESS REGISTER



Offset: 0x0010

Bits	Name	Type	Reset	Description
31:25	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
24	*INFO*	WO	0	Eflash Information Block control "Embedded Program/Erase Aaddress Register. (Page Size=512 Bytes) Mass Erase: HADDR[15:0] don't care. Page Erase : HADDR[8:0] don't care. HADDR[15:9] is page address.(128 page/64KBytes)"
23:0	*HADDR*	R/W	0x000000	Program: HADDR[1:0] don't care. HADDR[15:2] is double word address.

4.6 GENERAL PURPOSE I/O (GPIO)

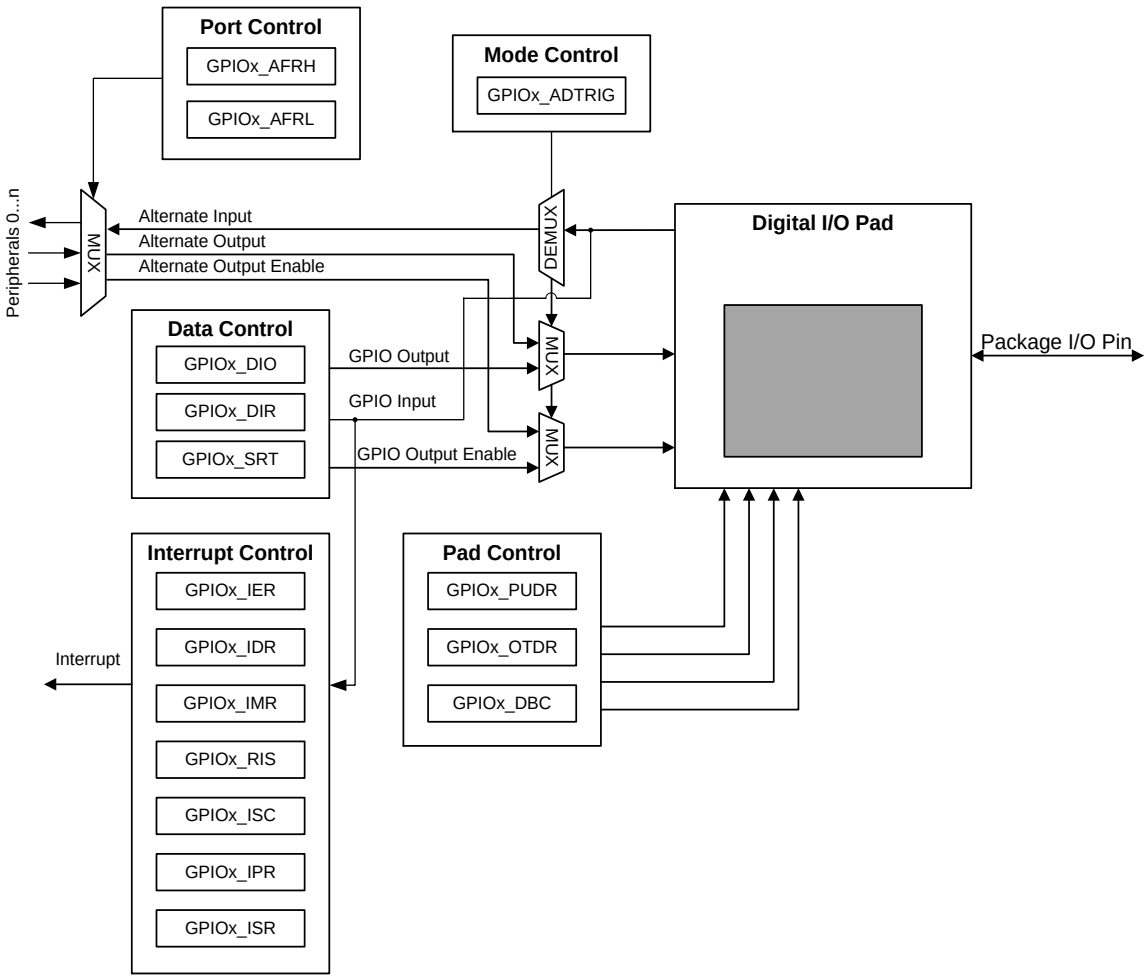
In PT32C302, the GPIO module has up to 56 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 56 pins are arranged in 4 ports named as GPIOA, GPIOB, GPIOC, and GPIOD. Each pin is independent and has the corresponding register bits to control the pin mode function and data. After reset, the I/O mode of all pins are floating.

- Input states : Floating, Pull-up/Pull-down, Analog
- Output states : Open drain or push-pull(normal)
- Pins configured as digital inputs are Schmitt-triggered
- Bit-level set and clear registers allow a single instruction set or clear of any number of bits in one port
- Programmable debounce time. (1-7) * (1-256) Clocks.
- Programmable control for GPIO interrupts

- Interrupt generation masking
- Edge-triggered on rising, falling
- Level-sensitive on High or Low value

4.6.1 BLOCK DIAGRAM

Figure 4.6-1: GPIO Block Diagram



4.6.2 FUNCTIONAL DESCRIPTION

4.6.2.1 DATA CONTROL

The data control process in GPIO is controlled by following registers:

- Data Input/ Output

GPIO (A/B/C/D) Input and Output Register (GPIOx_DIO) contains the output or input value of the corresponding I/O port. It is a read-only/write-only register depending on the direction setting on port.

- Data Direction Control

GPIO (A/B/C/D) Data Direction Register (GPIOx_DIR) configures each individual pin as an input or an output. Each pin is set to be in Input state by default. Writing a '0' to bit is set the pin to work in output state and the corresponding data register bit will be driven out on the GPIO port.

- Data Set and Clear

GPIO (A/B/C/D) Set/Reset Register (GPIOx_SRT) controls the modification of individual bit without affecting other bits. By writing a '1' to the bit in this register, it will set or clear the corresponding bit that is stored in GPIOx_DIO.

4.6.2.2 INTERRUPT CONTROL

The interrupt in GPIO are controlled by a set of seven registers.

- Interrupt Control (IER, IDR, IMR)

By default, the generation of interrupts of each pin is disabled, so user can configure the respective pin as interrupt. Interrupts are disabled on the corresponding bits of Port (X) if the corresponding data direction register is set to Output or if Port (X) mode is set to Hardware**.** Interrupt enable register **(GPIOx_IER)** enables the interrupt request lines by writing a '1'. Similarly, Interrupt disable register **(GPIOx_IDR)** disables the interrupt request lines by writing a '1'. IER and IDR are write only registers which control the masking of interrupts. The overall result of these two registers can be shown by Interrupt Mask Register **(GPIOx_IMR)**. IMR is a read-only register using '1' or '0' to indicate if the interrupt request line is enabled/ or disabled.

- Interrupt Status Read (RIS)

Raw Interrupt Status **(GPIOx_RIS)** is a read-only register to read all interrupt status of the module.

- Interrupt Clear (ISC)

Interrupt Status & Interrupt Clear Register **(GPIOx_ISC)** is used to indicate the non-masked interrupt status of the module, since only now-masked interrupts are asserted to processor. Writing a '1' to the bit in this register can clear the corresponding interrupt.

- Sense and Polarity Define

Interrupt Sense Register **(GPIOx_ISR)** configures the corresponding pins to detect level or edges. Interrupt Polarity Register **(GPIOx_IPR)** selects the polarity of the interrupt/event signals.

4.6.2.3 ANALOG CONFIGURATION

When I/O port is programmed as analog configuration by register Trigger ADC Register **(GPIOx_ADTRIG)**, the output buffer is disabled. The Schmitt trigger input is deactived and its output is a constant '0'.

4.6.2.4 MODE CONTROL

Each pin is 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. By default, each I/O pin are connected to Alternate Function 0 (AF0). Pins can work under alternate function mode, i.e. AF0-AF8 by controlling two alternate function registers: **Alternate Function Low Register (GPIOx_AFRL)** and **Alternate Function High Register (AFGH)**

4.6.2.5 PAD CONTROL

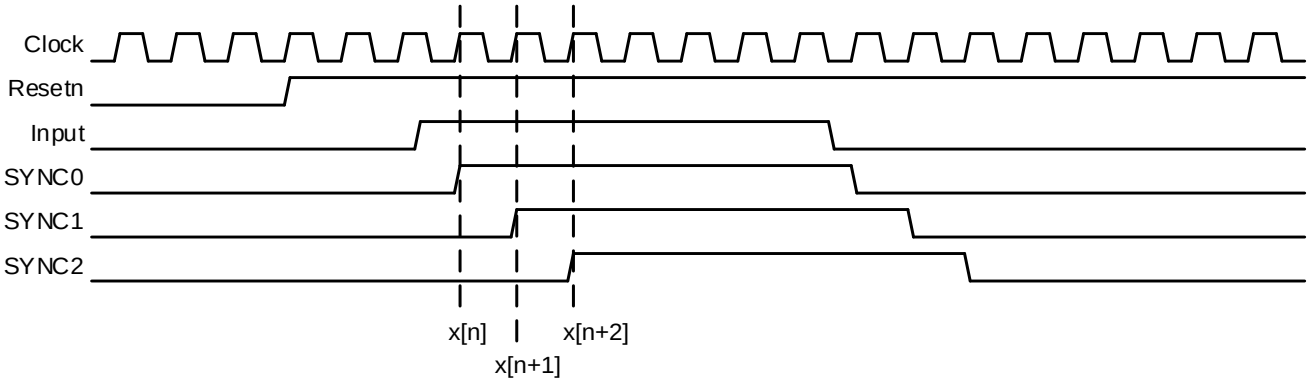
The pad control registers in GPIO include the GPIO (A/B/C/D) Port Pull-Up/Pull-Down Register (**GPIOx_PUDR**) and GPIO (A/B/C/D) Port Output Type and Drive Register **(GPIOx_OTDR)**. These register control the pull-up and pull-down resistor, output type and drive strength.

4.6.2.6 DEBOUNCE FUNCTION DESCRIPTION

Each GPIO pin has the debounce function. The timing of the signal from IO into the core, it could be expressed as follows:

When signal into the GPIO, it will through the sample circuit, the waveform as follows:

Figure 4.6-2: Debounce Waveform



In the above figure, SYNC0 means the Input signal is sampling once by system clock; SYNC1 means the SYNC0 is sampling once by system clock; SYNC2 means the SYNC1 is sampling once by system clock. The SYNC0, SYNC1 and SYNC2 can be expressed as $x[n]\times T_s$, $x[n+1]\times T_s$ and $x[n+2]\times T_s$. The T_s is system clock period, n is the sample time. If close the debounce module, the time can be express as $x[n+1]\times T_s$.

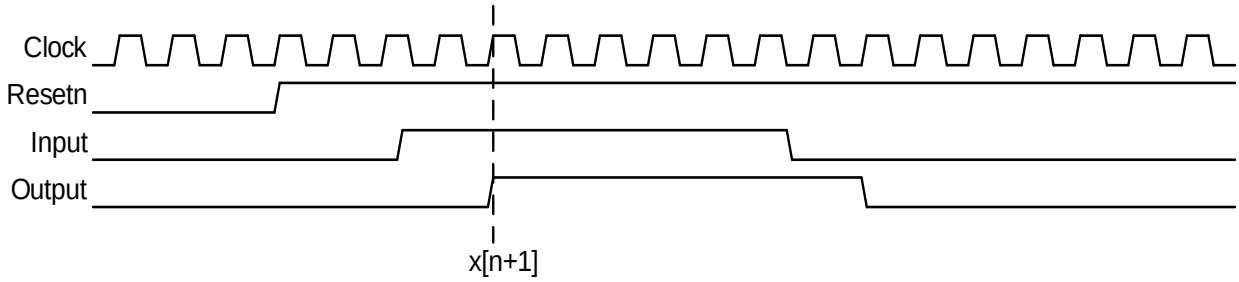
If open the debounce module, SYNC2 signal will go into the debounce circuit, then it will be sampled and counted times (sample frequency and count times value are setting by user). The time can be express as $[(SPT+1)\times(FILTCNT+1)]\times T_s$. The SPT is sample frequency value, FILTCNT is the count times.

When SYNC1 and SYNC2 are not the same, the count times value will be cleared to 0. If count times value meet FILTCNT register, the debounce module will output the signal.

Example 1

If system frequency is 48MHz, and close debounce.

Figure 4.6-3: Debounce Waveform

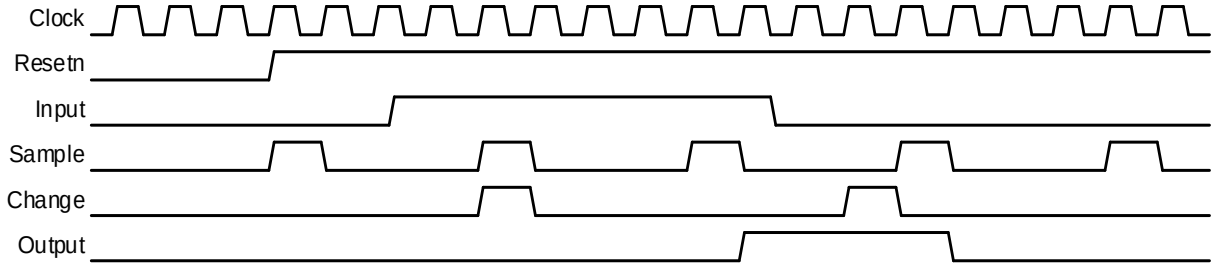


In the above diagram, the debounce function was be disabled, so the output will happen **$x[n+1]\times T_s$**.

Example 2

If system frequency is 48MHz, and sample frequency (**SPT**) is set to 0x3, count times (**FILTCNT**) is 0x0.

Figure 4.6-4: Debounce Waveform



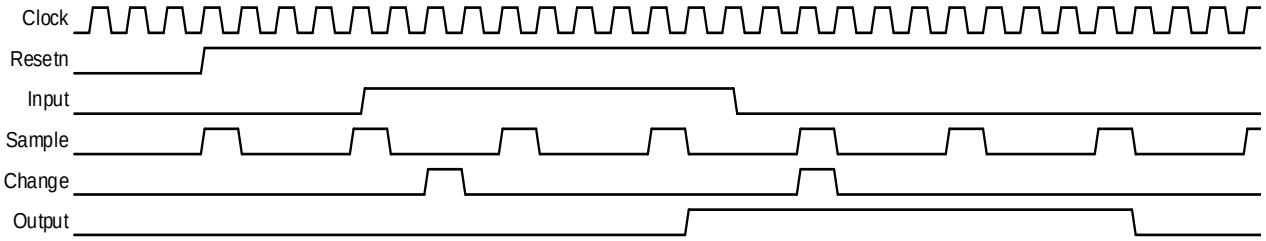
In the above diagram, debounce module is enabled, “Sample” is the sample frequency, “Change” is represented for SYNC1 and SYNC2 are not the same. When “Change” is high, it means count times value will be cleared to 0, so the “Output” will be set after “Change” was happened and SYNC2 signal is sampling once by “Sample”.

After debounce circuit sampling, it have the maximum error is $4\times T_s$, so the input time is: $[(3+1)\times(0+1)]\times T_s = 4\times T_s$, and then add sample circuit $3\times T_s$, the result is $7\times T_s$. Compare with the diagram, the input time is $6\times T_s$, the lost $1\times T_s$ is the sampling error.

Example 3

If system frequency is 48MHz, sample frequency (**SPT**) is 0x3, count times (**FILTCNT**) is 0x1.

Figure 4.6-5: Debounce Waveform



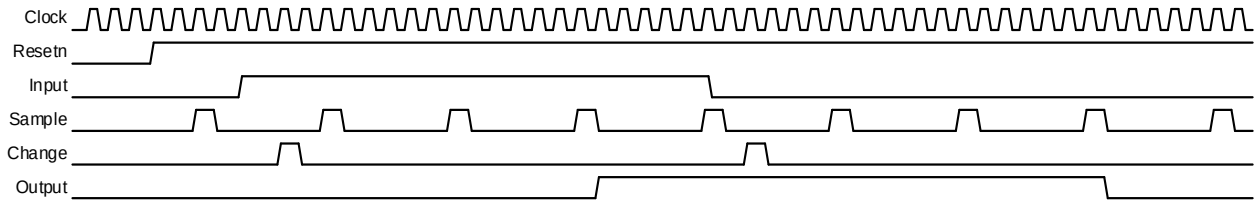
In the above diagram, debounce module is enabled, “**Sample**” is the sample frequency, “**Change**” is represented for **SYNC1** and **SYNC2** are not the same. When “**Change**” is high, it means count times value will be cleared to 0, so the “**Output**” will be set after “**Change**” was happened and **SYNC2** signal is sampling twice by “**Sample**”.

After debounce circuit sampling, it have the maximum error is $4\times T_s$, so the input time is: $[(3+1)\times(1+1)]\times T_s = 8\times T_s$, and then add sample circuit $3\times T_s$, the result is $11\times T_s$. Compare with the diagram, the input time is $8\times T_s$, the lost $3\times T_s$ is the sampling error.

Example 4

If system frequency is 48MHz, sample frequency (**SPT**) is 0x5, count times (**FILTCNT**) is 0x2.

Figure 4.6-6: Debounce Waveform



In the above diagram, debounce module is enabled, "**Sample**" is the sample frequency, "**Change**" is represented for **SYNC1** and **SYNC2** are not the same. When "**Change**" is high, it means count times value will be cleared to 0, so the "**Output**" will be set after "**Change**" was happened and **SYNC2** signal is sampling third by "**Sample**".

After debounce circuit sampling, it have the maximum error is $6\times T_s$, so the input time is: $[(5+1)\times(2+1)]\times T_s = 18\times T_s$, and then add sample circuit $3\times T_s$, the result is $21\times T_s$. Compare with the diagram, the input time is $16\times T_s$, the lost $5\times T_s$ is the sampling error.

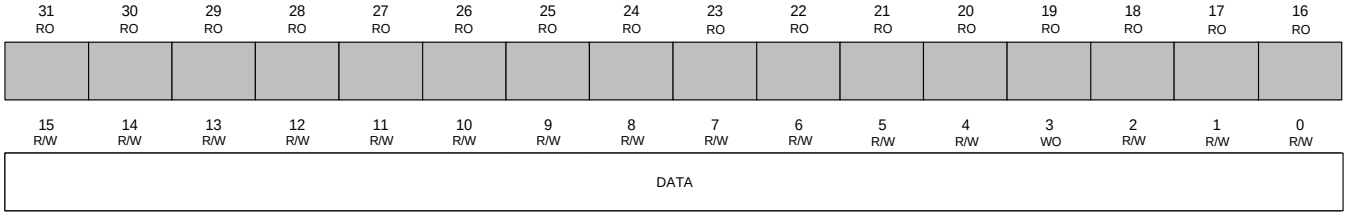
4.6.3 GENERAL-PURPOSE INPUT/OUTPUT REGISTER MAP

GPIO port's base address:

- GPIO Port A: 0x5002_0000
- GPIO Port B: 0x5002_0100
- GPIO Port C: 0x5002_0200
- GPIO Port D: 0x5002_0300

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	GPIOx_DIO	R/W	0x0000_FFFF	GPIOx Data Input and Output Register	93
0x0004	GPIOx_DIR	R/W	0x0000_FFFF	GPIOx Data Direction Register	93
0x0008	GPIOx_SRT	WO	0x0000_0000	GPIOx Set/Rest Register	94
0x000C	GPIOx_DBC	R/W	0x0000_0000	GPIOx Debounce Count Register	94
0x0010	GPIOx_IPR	R/W	0x0000_0000	GPIOx Interrupt Polarity Register	95
0x0014	GPIOx_ISR	R/W	0x0000_0000	GPIOx Interrupt Sense Register	95
0x0018	GPIOx_IER	WO	0x0000_0000	GPIOx Interrupt Enable Register	96
0x001C	GPIOx_IDR	WO	0x0000_0000	GPIOx Interrupt Disable Register	96
0x0020	GPIOx_IMS	RO	0x0000_0000	GPIOx Interrupt Mask Status Register	97
0x0024	GPIOx_RIS	RO	0x0000_0000	GPIOx Raw Interrupt Status Register	97
0x0028	GPIOx_ISC	RW/1C	0x0000_0000	GPIOx Interrupt Masked Status & Interrupt Clear Register	98
0x002C	reserved	N/A	-	Reserved	-
0x0030	GPIOx_PUDR	R/W	0x0000_0000	GPIOx Port Pull-Up/Pull-Down Register	98
0x0034	GPIOx_OTSR	R/W	0x0000_0000	GPIOx Port Output Type and Drive Register	99
0x0038	GPIOx_AFRL	R/W	0x0000_0000	GPIOx Alternate Function Low Register	99
0x003C	GPIOx_AFRH	R/W	0x0000_0000	GPIOx Alternate Function High Register	100
0x0040	GPIOx_ADTRIG	R/W	0x0000_0000	GPIOx Trigger ADC Register	100

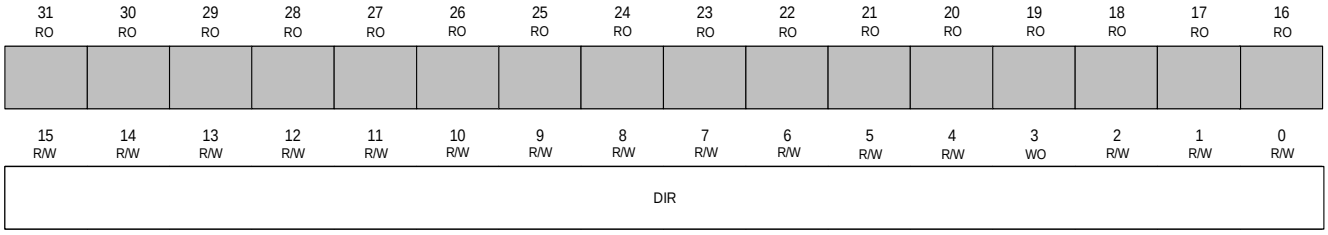
4.6.3.1 GPIOX_DIO - GPIOX INPUT AND OUTPUT REGISTER



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*DOUT*	W	0xFFFF	GPIOx output data, when GPIO direction was set to the output. (Port A, Port B, Port C : 0x0000_03FF / Port D: 0x0000_3800)
15:0	*DIN*	R	0x03FF	GPIOx input data, when GPIO direction was set to the input. (Port A, Port B, Port C : 0x0000_03FF / Port D: 0x0000_3800)

4.6.3.2 GPIOX_DIR - GPIOX DATA DIRECTION REGISTER



Offset: 0x0004

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*DIR*	R/W	0xFFFF	GPIO direction select Values written to this register independently control the direction of the corresponding data bit in Port. 0: Pins are output 1: Pins are input as default

4.6.3.3 GPIOX_SRT - GPIOX SET/RESET REGISTER



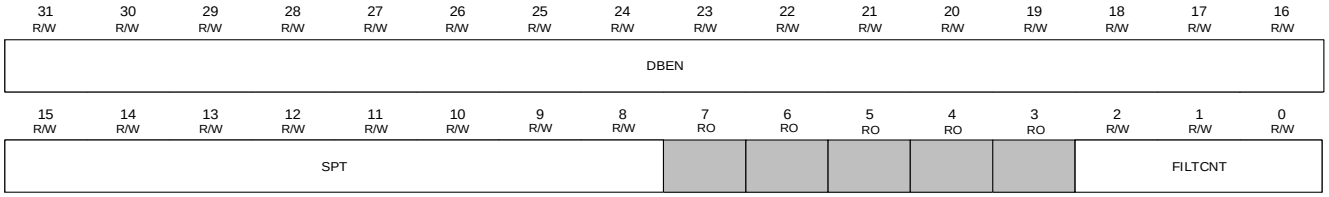
Offset: 0x0008

Bits	Name	Type	Reset	Description
31:16	*RESET*	WO	0x0	Port x reset bit 0: No action on the corresponding pin's output data 1: Reset the corresponding pin's output data, it is auto-clear in the next clock cycle. If Set/Reset both write 1 at same time, it takes as force reset.
15:0	*SET*	WO	0x0	Port x set bit 0: No action on the corresponding pin's output data. 1: Set the corresponding pin's output data, it is auto-clear in the next clock cycle.

4.6.3.4 GPIOX_DBC - GPIOX DEBOUNCE COUNT REGISTER

The DBEN[x] bit is used to enable debounce function of each corresponding GPIO input. If the input pulse width cannot be sampled by continuous "SPT" cycles, the input signal transition is seen as signal bounce, and will not trigger interrupt and signal state change. The debounce sample period is control by "SPT". For example: If enable debounce, and to set SPT = 5, FILTCNT = 1. So it means every 6 system clock to sampling once, and if sampling twice value are the same, it will output the value.

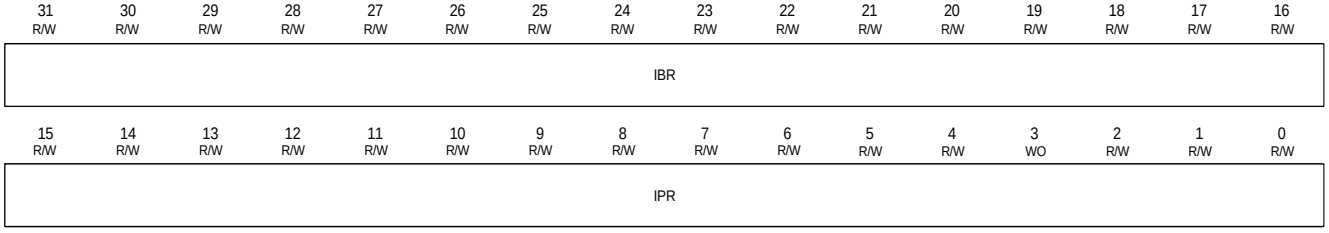
Derivation of the formula (system frequency / SPT / FILTCNT) = maximum data transfer effective frequency. For example: If system clock is 48 MHz, the debounce output of frequency is below 4 MHz (48/6/2 = 4 MHz); If the SPT and FILTCNT value are maximum, the valid frequency of input data is 23.437 kHz (48/256/8 = 23.437 kHz).



Offset: 0x000C

Bits	Name	Type	Reset	Description
31:16	*DBEN*	R/W	0x0	GPIO Debounce Enable 1: Enable debounce Debouncing on each pin can be controlled individually
15:8	*SPT*	R/W	0x0	GPIO Debounce Period Control Must be set to some value greater than zero, if the debounce function is enabled
7:3	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*FILTCNT*	R/W	0x0	GPIO Debounce Count Times

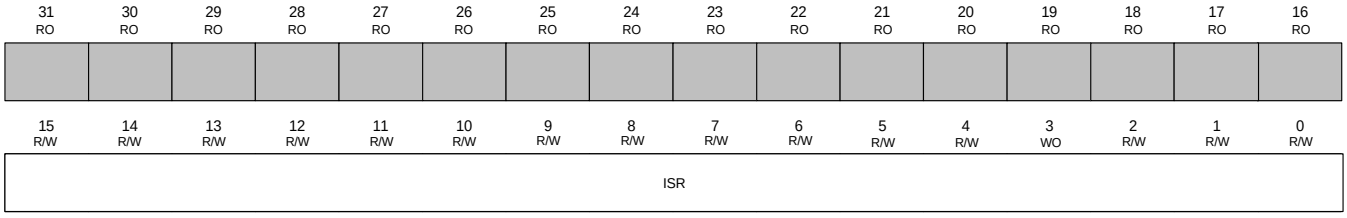
4.6.3.5 GPIOX_IPR - GPIOX INTERRUPT POLARITY REGISTER



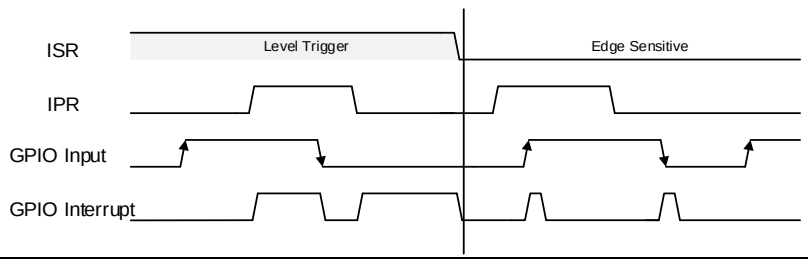
Offset: 0x0010

Bits	Name	Type	Reset	Description
31:16	*IBR*	R/W	0x000	0: According to the IPR bits to generate interrupt trigger. 1: Detect both edge to generate interrupt trigger.
15:0	*IPR*	R/W	0x000	Interrupt Polarity GPIOx interrupt is triggered when input data is level high or low. Controls the polarity of edge or level sensitivity that can occur on input of Port (X). Whenever a 0 is written to a bit of this register, it configures the interrupt type to falling-edge or active-low sensitive; otherwise, it is rising-edge or active-high sensitive. 0 : Active-low (default) 1 : Active-high

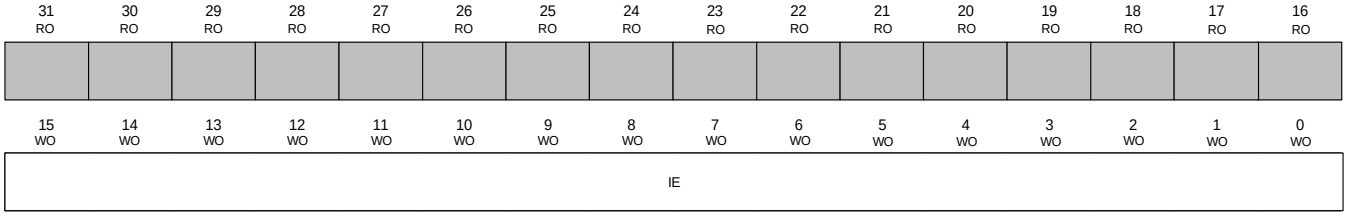
4.6.3.6 GPIOX_ISR - GPIOX INTERRUPT SENSE TYPE REGISTER



Offset: 0x0014

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*ISR*	R/W	0x0	GPIOx interrupt sense type select Controls the type of interrupt that can occur on Port (X).Whenever a 1 is written to a bit of this register, it configures the interrupt type to be level-sensitive; otherwise, and it is edge-sensitive. 0 : Edge-sensitive (default) 1 : Level-sensitive 

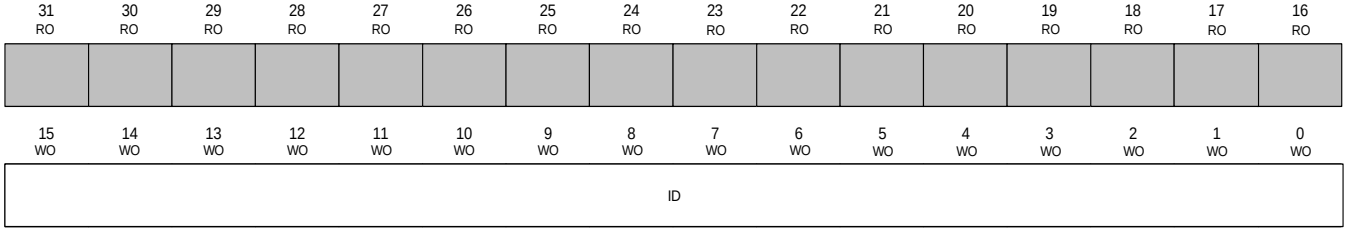
4.6.3.7 GPIOX_IER - GPIOX INTERRUPT ENABLE REGISTER



Offset: 0x0018

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*IE*	WO	0x0	GPIOx Interrupt Enable 0 : Configure Port (X) bit as normal GPIO signal (default) 1 : Configure Port (X) bit as interrupt

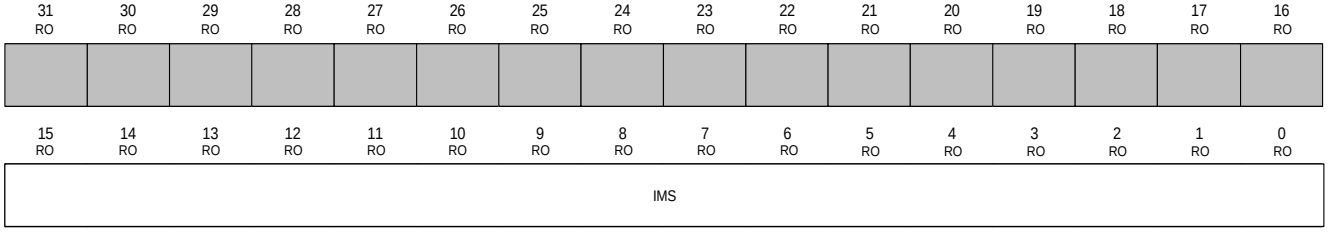
4.6.3.8 GPIOX_IDR - GPIOX INTERRUPT DISABLE REGISTER



Offset: 0x001C

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*ID*	WO	0x0	GPIOx Interrupt Disable 0 : Configure Port (X) bit as normal GPIO signal (default) 1 : Disable Port (X) bit as interrupt

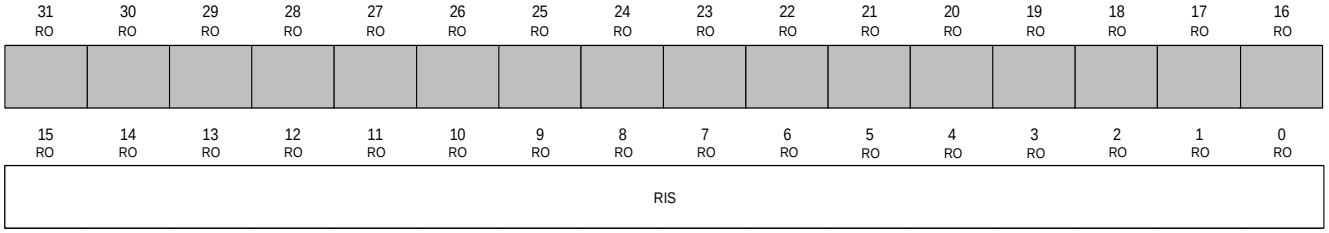
4.6.3.9 GPIOX_IMR - GPIOX INTERRUPT MASK STATUS REGISTER



Offset: 0x0020

Bits	Name	Type	Reset	Description
31:16	reserved	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	IMR	RO	0x0000	GPIOx Interrupt Mask Status 0 : Corresponding pin interrupt is masked 1 : Corresponding pin interrupt is not masked

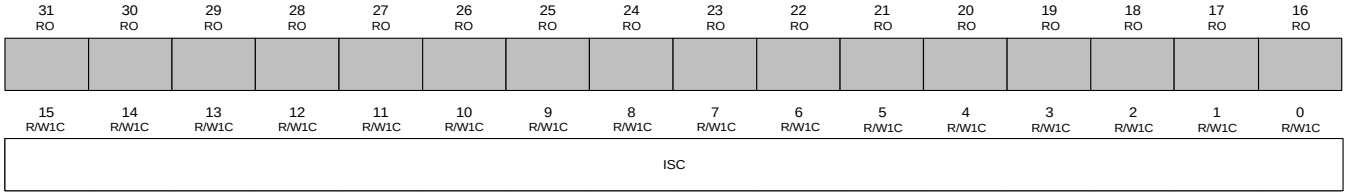
4.6.3.10 GPIOX_RIS - GPIOX RAW INTERRUPT STATUS REGISTER



Offset: 0x0024

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*RIS*	RO	0x0	GPIOx Interrupt Raw Status 0 : Corresponding pin interrupt requirements not met 1 : Corresponding pin interrupt has met requirements

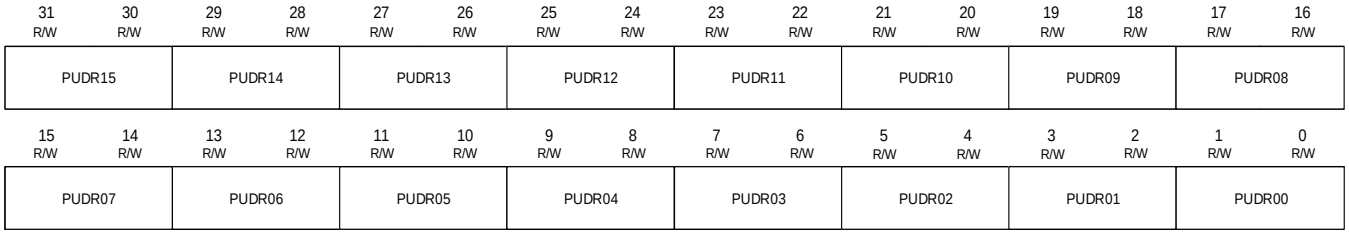
4.6.3.11 GPIOX_ISC - GPIOX INTERRUPT STATUS & INTERRUPT CLEAR REGISTER



Offset: 0x0028

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*ISC*	R/W1C	0x0	GPIOx Interrupt Status and Clear 0 : Corresponding pin interrupt not active 1 : Corresponding pin interrupt asserting Write '1' to clear this interrupt

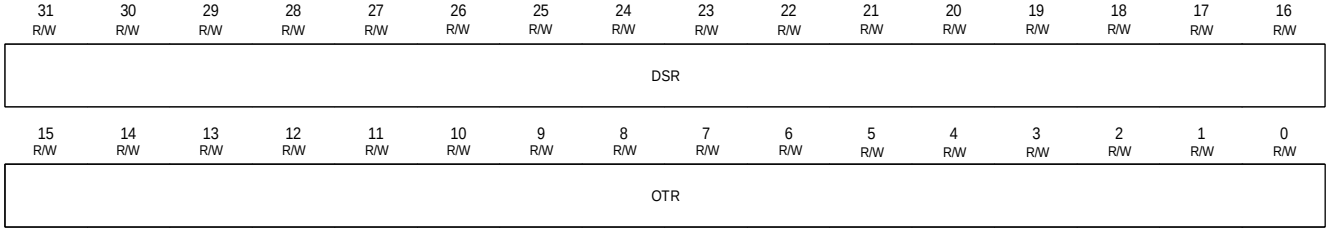
4.6.3.12 GPIOX_PUDR - GPIOX PORT PULL-UP/PULL-DOWN REGISTER



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:30	*PUDR15*	R/W	0x0	GPIOx Port Pull-Up/Pull-Down Register 00 : Floating 01 : Pull-Up 10 : Pull-Down 11 : Reserved Floating
29:28	*PUDR14*	R/W	0x0	
27:26	*PUDR13*	R/W	0x0	
25:24	*PUDR12*	R/W	0x0	
23:22	*PUDR11*	R/W	0x0	
21:20	*PUDR10*	R/W	0x0	
19:18	*PUDR09*	R/W	0x0	
17:16	*PUDR08*	R/W	0x0	
15:14	*PUDR07*	R/W	0x0	
13:12	*PUDR06*	R/W	0x0	
11:10	*PUDR05*	R/W	0x0	
9:8	*PUDR04*	R/W	0x0	
7:6	*PUDR03*	R/W	0x0	
5:4	*PUDR02*	R/W	0x0	
3:2	*PUDR01*	R/W	0x0	
1:0	*PUDR00*	R/W	0x0	

4.6.3.13 GPIOX_OTDR - GPIOX PORT OUTPUT TYPE AND DRIVE REGISTER

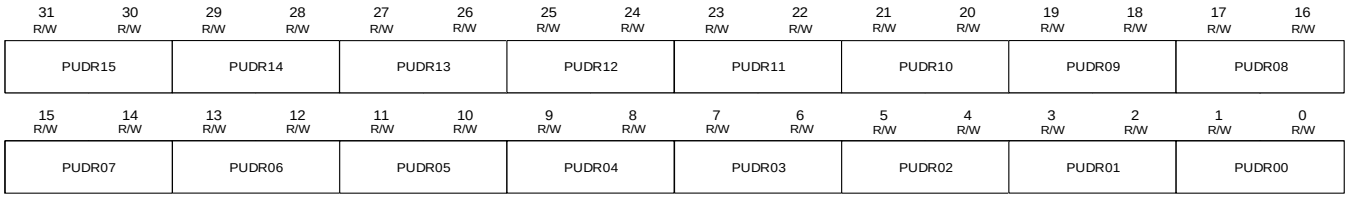


Offset: 0x0034

Bits	Name	Type	Reset	Description
31:16	*DSR*	R/W	0x0	GPIOx Output Drive Select Register 0 : 2mA 1 : 4mA
15:0	*OTR*	R/W	0x0	GPIOx Output Type Select Register 0 : Normal 1 : Open Drain

4.6.3.14 GPIOX_AFRL - ALTERNATE FUNCTION LOW REGISTER

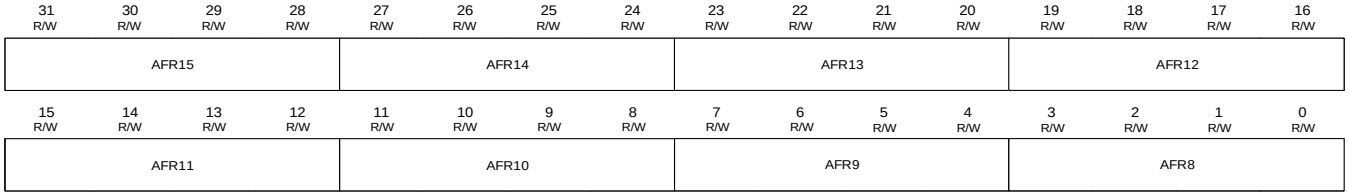
GPIO provides multiple functions of IO options, select other functions through **GPIOx_AFRL** and **GPIOx_AFRH**, classified functions. For example, if the pin PA00 whose default function is GPIO (PA00), and to configure its functions to AD0_CAIP0 (ADCIN0 & Comparator 0 Positive input), you must setting the register **GPIOA_AFR0** is 8. For further description about the alternate functions in PT32C302, refer to Signal Description and Pin Definition and Multiplexing.



Offset: 0x0038

Bits	Name	Type	Reset	Description
31:28	*AFR7*	R/W	0x0	Alternate function selection for port x pin y (y = 0...7) 0: AF0 1: AF1 2: AF2 3: AF3 4: AF4 ….. 8: AF_ANA(Analog)
27:24	*AFR6*	R/W	0x0	
23:20	*AFR5*	R/W	0x0	
19:16	*AFR4*	R/W	0x0	
15:12	*AFR3*	R/W	0x0	
11:8	*AFR2*	R/W	0x0	
7:4	*AFR1*	R/W	0x0	
3:0	*AFR0*	R/W	0x0	

4.6.3.15 GPIOX_AFRH - ALTERNATE FUNCTION HIGH REGISTER

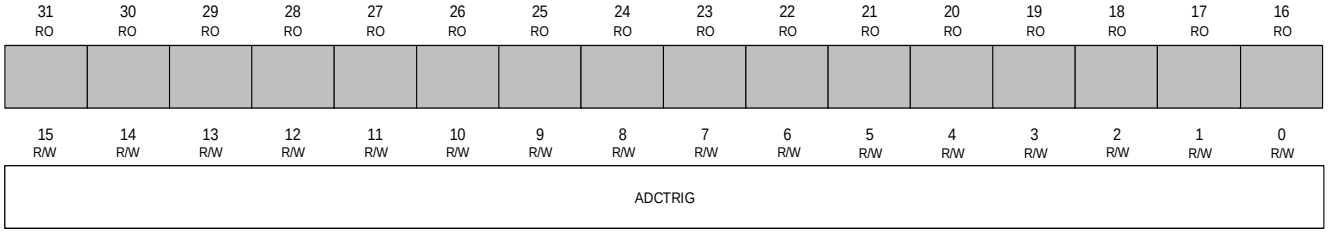


Offset: 0x003C

Bits	Name	Type	Reset	Description
31:28	*AFR15*	R/W	0x0	Alternate function selection for port x pin y (y = 8...15) 0: AF0 1: AF1 2: AF2 3: AF3 4: AF4 ….. 8: AF_ANA(Analog)
27:24	*AFR14*	R/W	0x0	
23:20	*AFR13*	R/W	0x0	
19:16	*AFR12*	R/W	0x0	
15:12	*AFR11*	R/W	0x0	
11:8	*AFR10*	R/W	0x0	
7:4	*AFR9*	R/W	0x0	
3:0	*AFR8*	R/W	0x0	

4.6.3.16 GPIOX_ADTRIG - GPIOX TRIGGER ADC REGISTER

Provide users choose to open GPIOx Port Interrupt trigger sampling ADC module.



Offset: 0x0040

Bits	Name	Type	Reset	Description
31:16	*reserved*	R/W	0000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*ADCTRIG*	R/W	0x0	GPIOx ADC Trigger Source Enable 0 : Disable Port (X) bit as trigger (default) 1 : Enable Port (X) bit for ADC trigger source

4.7 UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART)

The UART Controller provides two channels of Universal Asynchronous Receiver/Transmitters (UART). UART supports the flow control function. UART performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the CPU. The UART controller also supports IrDA SIR, modem operation and RS-485 mode functions.

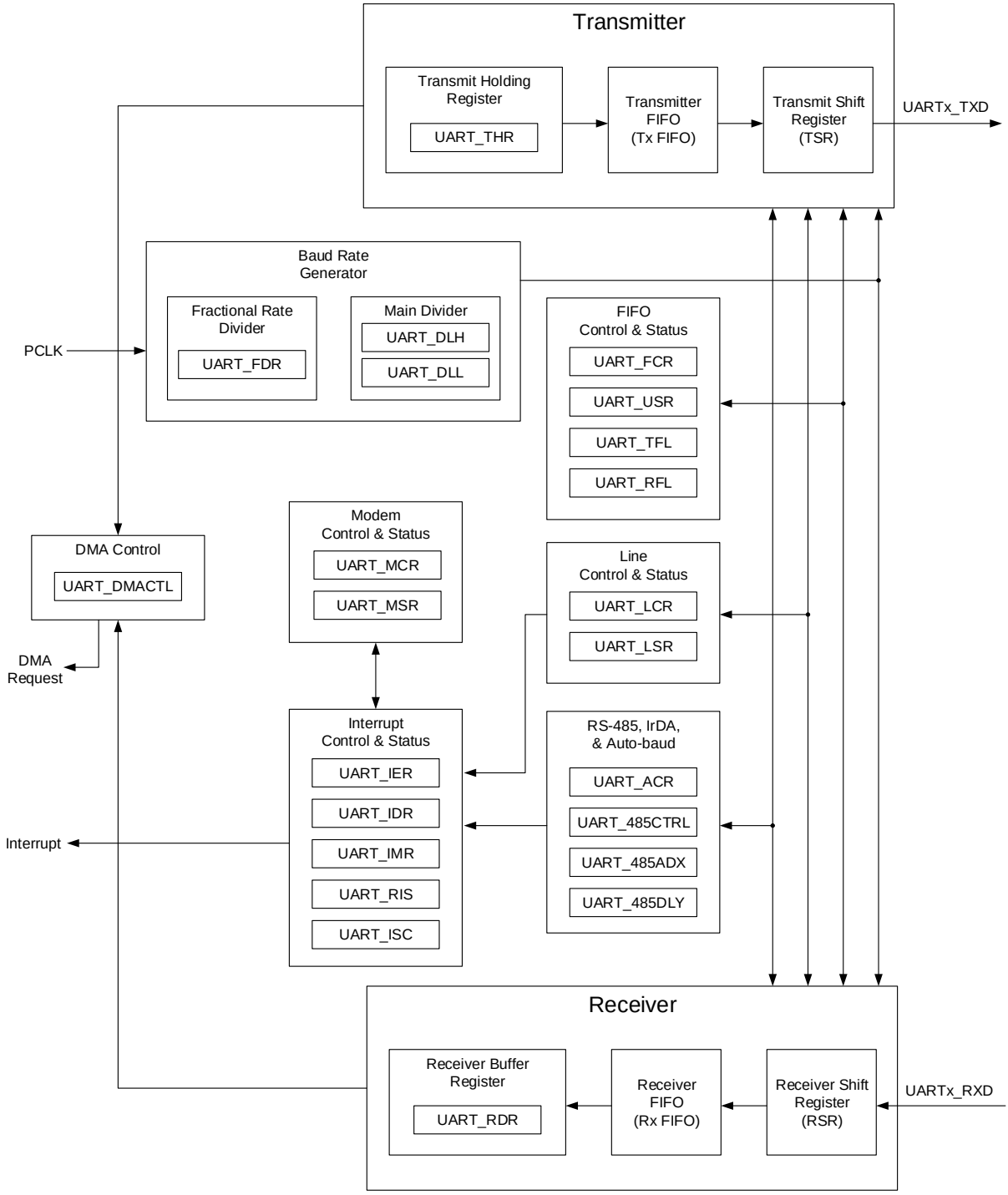
- Full Duplex, Asynchronous Communication.
- 16-byte Receive and Transmit FIFOs.
- Compatible with 16C550 standard.
- Programmable receiver buffer trigger level.
- Programmable baud-rate generator for each channel individually.
- Supports auto baud rate detect function.
- Eight interrupt sources
- Built-in fractional baud rate generator covering wide range of baud rates without a need for external crystals of particular values
- Hardware auto flow control/flow control function (CTSn, RTSn) and programmable RTSn flow control trigger level
- Supports CTSn wake-up function
- Support IrDA SIR Mode
  - Supports 3/16 bit period modulation.
- RS-485 support
  - 9-bit mode
- Fully programmable serial-interface characteristics
  - Programmable number of data bit, 5-, 6-, 7-, 8- bit character
  - Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
  - Programmable stop bit, 1, 1.5, or 2 stop bit generation

4.7.1 BLOCK DIAGRAM

The Receiver block monitors the serial input line for valid input. The Receiver Shift Register (RSR) accepts valid characters via UARTx_RXD. After a valid character is assembled in the RSR, it is passed to the UART Rx Buffer Register to await access by the CPU or host.

The Transmit block accepts data written by the CPU or host and buffers the data in the UART Transmit Holding Register. The Transmit Shift Register reads the data stored in the **UART_THR** and assembles the data to transmit via the UARTx_TXD.

Figure 4.7-1: UART Block Diagram



4.7.2 FUNCTIONAL DESCRIPTION

4.7.2.1 TRANSMIT/RECEIVE LOGIC

The transmit logic performs parallel-to-serial conversion on the data read from the transmit FIFO. The control logic outputs the serial bit stream beginning with a start bit and followed by the data bits(LSB first), parity bit, and the stop bits according to the programmed configuration in the line control register. The receive logic performs serial-parallel conversion on the received by stream after a valid start pulse has been detected.

4.7.2.2 AUTO-BAUD RATE

The UART auto-baud rate detection can be used to measure the incoming baud rate based on the "AT" protocol (Hayes command). If the auto-baud feature is enabled, controller will measure the bit time of the receive data stream and set the divisor latch registers **UART_DLL** and **UART_DLH** accordingly.

Auto-baud rate detection is started by setting the **UART_ACR** - *START* bit and stopped by clearing this bit. The *START* bit will be cleared once auto-baud has finished and reading the bit will return the status of auto-baud (pending/finished). Self-restart of auto-baud rate detection is supported if the **UART_ACR** - *ATSTART* bit is set. The controller will automatically detect the baud rate one more time if the first detection is failed.

Two auto-baud rate measuring modes which can be selected by the **UART_ACR** - *MODE* bit are available.

- In Mode 0 - Baud rate is measured on two subsequent falling edges of the UART Rx pin (the falling edge of the start bit and the falling edge of the least-significant bit (LSB)).
- In Mode 1 - Baud rate is measured between the falling edge and the subsequent rising edge of the UART Rx pin (the length of the start bit).

The **UART_ACR** - *ATSTART* bit can be used to automatically restart baud rate measurement if a time-out occurs (the rate measurement counter overflows). If this bit is set, the rate measurement will restart at the next falling edge of the UART Rx pin.

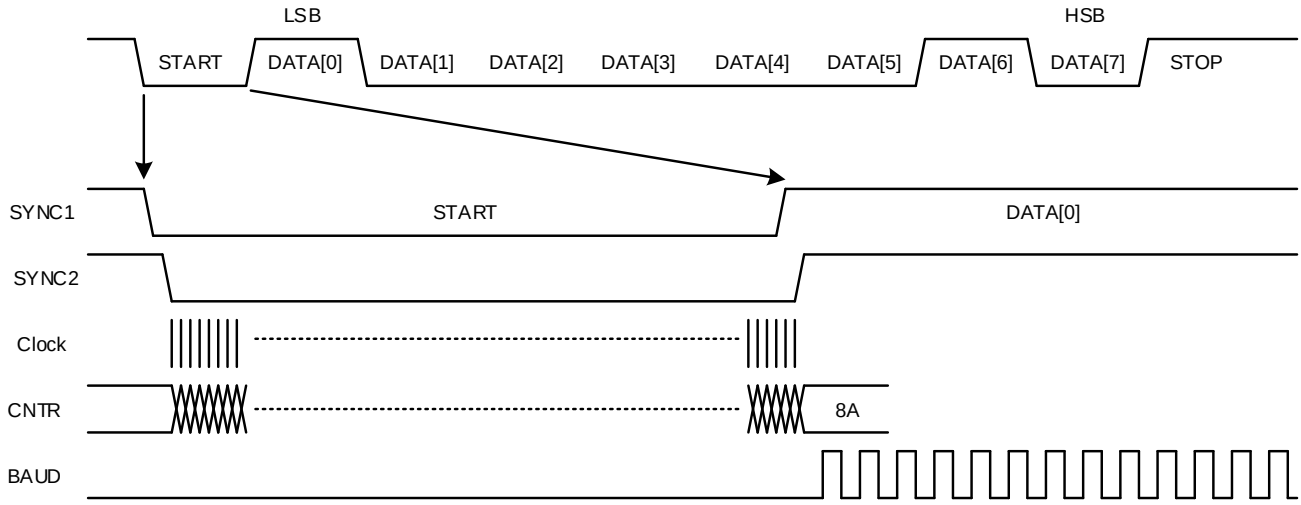
The auto-baud function can generate two interrupts.

- The auto-baud rate detection timeout interrupt (denoted as *ABTOIE* in **UART_IER**) will get set if the auto-baud rate measurement counter overflows).
- The auto-baud rate detection end interrupt (denoted as *ABEIE* in **UART_IER**) will get set if the auto-baud has completed successfully.

When the software is expecting an "AT" command, it configures the UART with the expected character format and sets the **UART_ACR** - *START* bit. The initial values in the divisor latches **UART_DLH** and **UART_DLL** don't care. Because of the "A" or "a" ASCII coding ("A" = 0x41,"a" = 0x61), the USART0 Rx pin sensed start bit and the LSB of the expected character are delimited by two falling edges. When the ACR Start bit is set, the auto-baud protocol will execute the following phases:

1. While UART_ACR-Start bit is set, the baud rate measurement counter is reset and the UART_RBR is reset.
2. When UART Rx pin detect falling edge, which is the START bit, the baud rate measurement counter is start counting the baud rate cycle using the UART (0/1) PCLK as time base which is controlled by the register SC_GCLK_APB.
3. If mode 0 is used, the baud rate measurement counter is stopped by the next falling edge (End Edge).
   If mode 1 is used, the baud rate measurement counter is stopped by the rising edge (End Edge).
4. When the end edge is detected, the baud rate measurement counter is load count value to UART_DLH or **UART_DLL** register, and the baud rate measurement counter is switch to normal mode. After loading value to **UART_DLH** or **UART_DLL** register, if *ABEIE* bit interrupt is asserted, then UART will start receiving data bit.

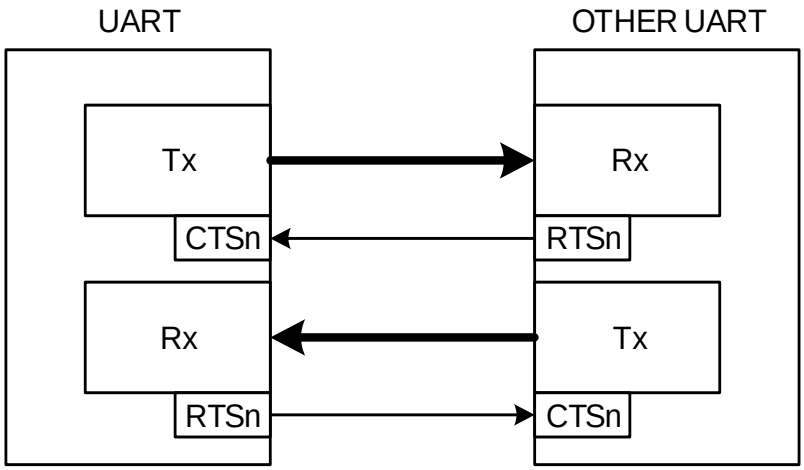
Figure 4.7-2: Only START bit to use for Auto-Baud-Rate



4.7.2.3 AUTO-FLOW CONTROL

If Auto-Flow Control is enabled, UART's receive (Rx) and transmit (Tx) are controlled by receive FIFO and transmit FIFO by using UART_RTSn and UART_CTSn pin.

Figure 4.7-3: The Auto-Flow-Control connect graphic



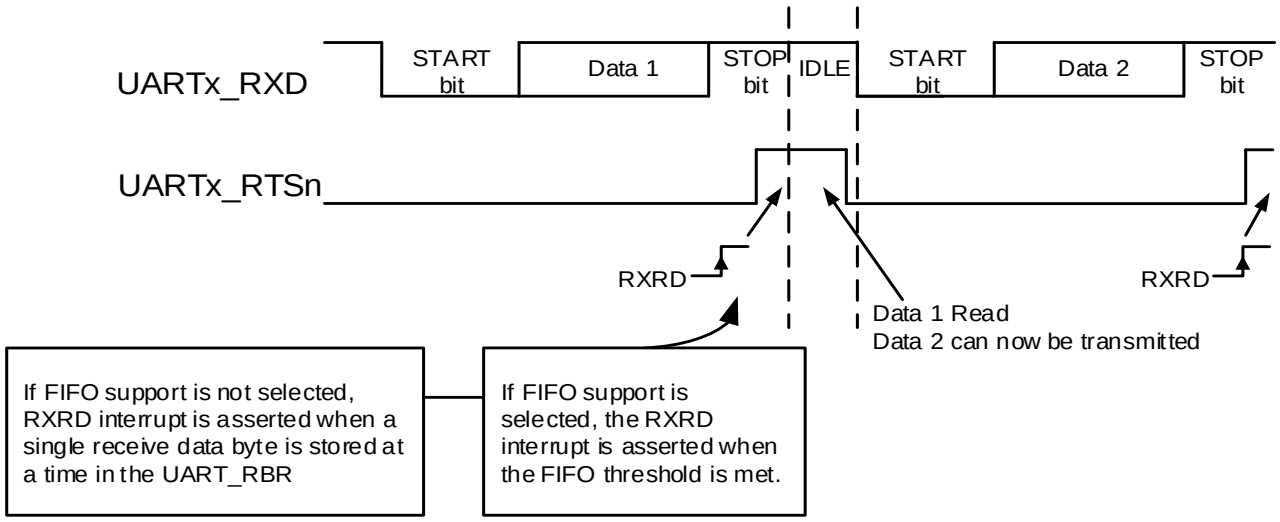
AUTO-RTS

When Auto RTS is enabled, the UART_RTSn output is high when the receiver FIFO level reaches the threshold set by **UART_FCR** register. When UART_RTSn is connected to the UART_CTSn input of another UART device, the other UART stops sending serial data until the receiver FIFO is completely empty.

The selectable receiver FIFO threshold values are: 1, $^1/_4$, $^1/_2$or “2 less than full”. Since one additional character may be transmitted to the UART after UART_RTSn has become inactive (due to data has already entered the transmitter block of another UART), setting the threshold to “2 less than full” allows maximum use of the FIFO with a safety zone of one character.

Once the receiver FIFO becomes completely empty by reading the Receive Buffer Register UART_RBR, UART_RTSn become low, signaling the other UART to continue sending data.

Figure 4.7-4: Auto-RTS Flow Control

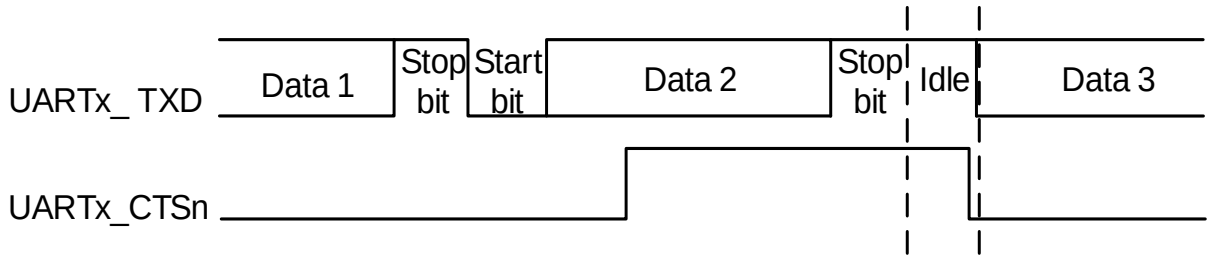


AUTO-CTS

When Auto CTS is enabled (active), the UART transmitter is disabled whenever the UART_CTSn input becomes high. This prevents overflowing the FIFO of the receiving UART.

If the UART_CTSn input is not inactivated before the middle of the last stop bit, another character is transmitted before the transmitter is disabled. While the transmitter is disabled, the transmitter FIFO can still be written to, and even overflowed.

Figure 4.7-5: Auto-CTS Function



The Auto-Flow Control can reduce the interrupt to the system. When Auto-Flow Control is enable, the CTSn status is not trigger the interrupt to the system since the device automatically controls its transmitter. When not using AutoFlow Control, the transmitter will send any information stored in the Tx FIFO, causing the receiver overflow.

4.7.2.4 INTERRUPT CONTROL

The UART can generate interrupts when the following conditions are met:

- Auto-baud rate detection timeout – Auto-baud rate detection Timeout occurs
- Auto-baud rate detection end – Auto-baud rate detection completes
- UART busy – Master has tried to write to the Line Control while the UART is busy
- Character timeout - No characters in or out of the RCVR FIFO during the last 4 character times and there is at least 1 character in it during this time
- Modem status - Clear to send or data set ready on ring indicator or data carrier detect
- Rx line status - Overrun/parity/ framing errors or break interrupt
- Tx empty - Transmitter holding register empty or XMIT FIFO at or below threshold
- Receive data available - Receiver data available or RCVR FIFO trigger level reached

The interrupt in UART are controlled by a set of five registers.

- Interrupt Control (**UART_IER**, **UART_IDR**, **UART_IMR**)

  UART Interrupt enable register **(UART_IER)** enables the interrupt request lines by writing a '1'. Similarly, UART Interrupt disable register **(UART_IDR)** disables the interrupt request lines by writing a '1'. **UART_IER** and **UART_IDR** are write only registers which control the masking of interrupts. The overall result of these two registers can be shown by **UART Interrupt Mask Register (UART_IMR)**. **UART_IMR** is a read-only register using '1' or '0' to indicate if the interrupt request line is enabled/ or disabled.

- Interrupt Status Read (RIS)

  UART Raw Interrupt Status (UART_RIS) is a read-only register to read all raw interrupt status of the module.

- Interrupt Clear (ISC)

  UART Interrupt Status & Interrupt Clear Register (UART_ISC) is used to indicate the non-masked interrupt status of the module, since only now-masked interrupts are asserted to processor. Writing a '1' to the bit in this register can clear the corresponding raw interrupt.

4.7.2.5 FIFO OPERATION

The UART can be configured to implement two 16-byte FIFOs to buffer transmit and receive data. If the FIFO support is not selected, then no FIFOs are implemented and only a single receive data byte and transmit data byte can be stored at a time in **UART_RBR** and **UART_THR**.

The trigger points at which the FIFOs generate interrupts is controlled via the UART FIFO Control Register (**UART_FCR**). Both FIFOs can be individually configured to trigger interrupt at different levels. Available configuration include 1, 1/4, 1/2or "2 less than full". For example, if the 1/4 is selected for the transmit FIFO, the UART generate a transmit interrupt after 4 data bytes are transmitted.

4.7.2.6 IRDA SIR

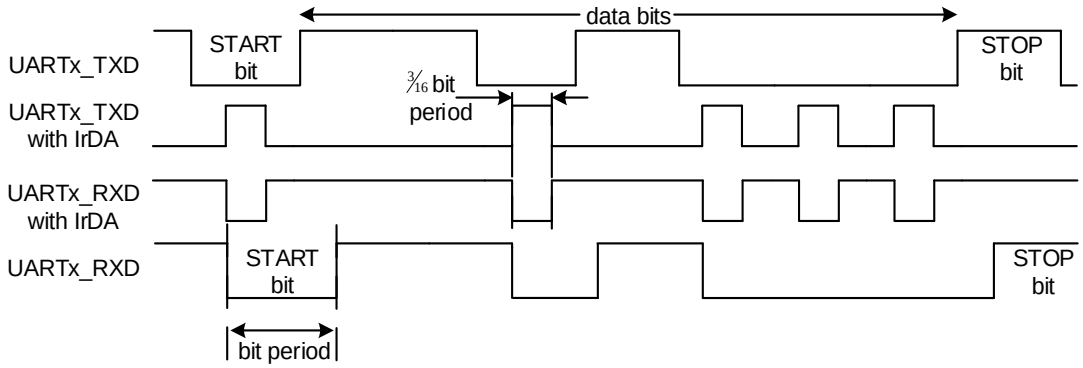
IrDA SIR mode supports bi-directional data communications with remote devices using infrared radiation as the transmission medium. Its data format is similar to the standard serial data format. Each data character is sent serially, beginning with a start bit, followed by 8 data bits, and ending with at least one stop bit. Thus, the number of data bits that can been sent is fixed. No parity information can be supplied and only one stop bit is used while in this mode.

Trying to adjust the number of sending data bits or enable parity with the UART Line Control Register **(UART_LCR)** has no effect. When the UART is configured to support IrDA 1.0 SIR it can be enabled with UART Mode Control Register **(UART_MCR)** *SIRE* bit. When the UART is not configured to support IrDA SIR Mode, none of the logic is implemented and the mode cannot be activated, reducing total gate counts. When SIR mode is enabled and active, the SIR block uses the UARTx_TXD and UARTx_RXD pins for SIR protocol. These signal should be connected to an infrared transceiver to implement IrDA SIR physical layer link.

Transmitting a single infrared pulse signals a logic zero, while a logic one is represented by not sending a pulse. A zero logic level is transmitted as a high pulse of 3/16th duration of the selected baud rate bit period on the output pin, while logic one levels are transmitted as a static LOW signal. These levels control the driver of an infrared transmitter, sending a pulse of light for each zero. On the reception side, the incoming light pulses energize the photo transistor base of the receiver, pulling its output LOW and driving the UART input pin LOW.

The timing diagram for the IrDA SIR data format in comparison to the standard serial format as follows.

Figure 4.7-6: IrDA SIR Data Format



IrDA SIR physical layer specifies a minimum of 10ms delay between transmission and reception, therefore when the IrDA SIR mode is enabled, the data transfers can only occur in half-duplex fashion. This delay must be generated by software.

4.7.2.7 RS-485/EIA-485 SUPPORT

In addition to standard transmit and receive data lines, UART also support for RS-485/9-bit mode function.

AUTO-9 BIT MODE

UART provides a 9-bit RS-485 mode. This feature is useful in a multi-drop configuration of the UART where a single master connected to multiple slaves. The addressable slave is one of multiple slaves controlled by a single master. In this mode of operation, a 'master' station transmits an address character followed by data characters for the addressed 'slave' stations. The slave stations examine the received data and interrupt the controller if the received character is an address character (9^{th} = parity bit = 1).

Each UART slave receiver can be assigned a unique address. The slave can be programmed to either manually or automatically reject data following an address which is not assigned to. With RS-485/EIA-485 support, each UART can be set to the following modes:

NORMAL MULTIDROP MODE (NMM)

Set the **UART_485CTRL**.*NMMEN* bit to enable NMM mode. In this mode, an address is detected when a received byte causes the UART to set the parity error and generate an interrupt. If the receiver is disabled (i.e. **UART_485CTRL**.*RXDIS* bit= '1'), any received data bytes will be ignored and will not be stored in the RXFIFO. When an address byte is detected (parity bit = '1') it will be placed onto the RXFIFO and an Rx Data Ready Interrupt will be generated. The processor can then read the address byte and decide whether or not to enable the receiver to accept the following data. While the receiver is enabled (**UART_485CTRL**.*RXDIS* bit ='0'), all received bytes will be accepted and stored in the RXFIFO regardless of whether they are data or address. When an address character is received a parity error interrupt will be generated and the processor can decide whether or not to disable the receiver.

AUTO ADDRESS DETECTION (AAD) MODE

When both **UART_485CTRL** register *NMMEN* bit (9-bit mode enable) and *AADEN* bit (AAD mode enable) are set, the UART is in auto address detect mode. In this mode, the receiver will compare any address byte received (parity = '1') to the 8-bitvalue programmed into the **UART_485ADX** register. If the receiver is disabled (**UART_485CTRL**.*RXDIS* bit = '1'), any received byte will be discarded if itis either a data byte OR an address byte which fails to match the **UART_485ADX** value. When a matching address character is detected it will be pushed onto the Rx FIFO along with the parity bit, and an Rx Data Ready Interrupt will be generated. The processor can then read the address byte and decide whether or not to enable the receiver to accept the following data. While the receiver is enabled (**UART_485CTRL**.*RXDIS* bit = '0'), all bytes received will be accepted and stored in the Rx FIFO. Until an address byte which does not match the **UART_RS485ADX** value is received. When this occurs, any received byte will be discarded. The received non-matching address character will not be stored in the Rx FIFO.

RS-485/EIA-485 Auto Direction Control

RS485 / EIA-485 mode includes automatic control to select whether to send the RTSn/CTSn pin state allows a direction control output signal. This feature is enabled by setting **UART_485CTRL**.*DIRCTRL* bit. If direction control is enabled, when **UART_485CTRL**.*OINV* = 0, then use the RTSn pin. When RS485CTRL.*OINV* = 1, then use the CTSn pin. When the automatic direction control is enabled, the selected pin in the CPU write data to TXFIFO will be pulled down (driven low). Once the last data bit sent, the selected pin will be pulled (driven high).

4.7.3 UART REGISTER MAP

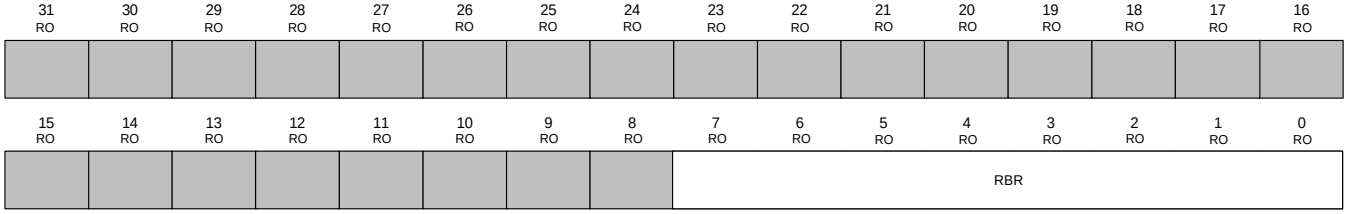
UART base address:

- UART 0: 0x4800_0000
- UART 1: 0x4800_0100
- UART 2: 0x4800_0200

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	UART_RBR	RO	0x0000_0000	UART Receive Buffer Register	110
0x0000	UART_THR	WO	0x0000_0000	UART Transmit Holding Register	110
0x0000	UART_DLL	R/W	0x0000_0000	UART Divisor Latch (Low)	111
0x0004	UART_DLH	R/W	0x0000_0000	UART Divisor Latch (High)	111
0x0004	UART_FCR	R/W	0x0000_0000	UART FIFO Control Register	112
0x0008	UART_ACR	R/W	0x0000_0000	UART Auto Baud Rate Control Register	113
0x000C	UART_FDR	R/W	0x0000_0000	UART Fractional divider register	113
0x0010	UART_LCR	R/W	0x0000_0000	UART Line Control Register	114
0x0014	UART_MCR	R/W	0x0000_0000	UART Modem Control Register	115
0x0018	UART_LSR	RO	0x0000_0060	UART Line Status Register	116
0x001C	UART_MSR	RO	0x0000_0000	UART Modem Status Register	118
0x0020	UART_IER	WO	0x0000_0000	UART Interrupt Enable Register	119
0x0024	UART_IDR	WO	0x0000_0000	UART Interrupt Disable Register	120
0x0028	UART_IMR	RO	0x0000_0000	UART Interrupt Mask Register	121
0x002C	UART_RIS	RO	0x0000_0002	UART Raw Interrupt Status Register	122
0x0030	UART_ISC	R/W1C	0x0000_0000	UART Interrupt Status and Clear Register	123
0x0034	UART_485CTRL	R/W	0x0000_0000	UART RS485 Control Register	124
0x0038	UART_485ADX	R/W	0x0000_0000	UART RS485 Address Match Register	125
0x003C	UART_485DLY	R/W	0x0000_0000	UART RS485 Direction Delay Register	125
0x007C	UART_USR	RO	0x0000_0006	UART Status Register	126
0x0080	UART_TFL	RO	0x0000_0000	UART Transmit FIFO Level Register	127
0x0084	UART_RFL	RO	0x0000_0000	UART Receive FIFO Level Register	127
0x00A8	UART_DMACTL	R/W	0x0000_0000	UART DMA Controller	128

4.7.3.1 UART_RBR - UART RECEIVE BUFFER REGISTER

This register is the data register used in receiving data. If the FIFO is enabled, this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data arrives, then the data already in the FIFO is preserved, but any incoming data are lost. If the FIFO is disabled, the data in this register must be read before the next data arrives, otherwise it is overwritten.

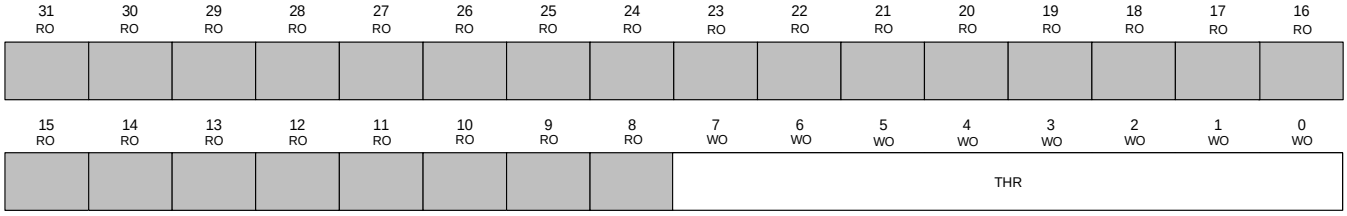


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*RBR*	RO	0x0	Receiver Buffer Register By reading this register, the UART will return an 8-bit data receiver from RX pin (LSB first).

4.7.3.2 UART_THR - UART TRANSMIT HOLDING REGISTER

This register is the data register used in transmitting data. If the FIFO is enabled and *THRE* bit in **UART_LSR** register is set, x number of characters of data may be written to this register before the FIFO is full.

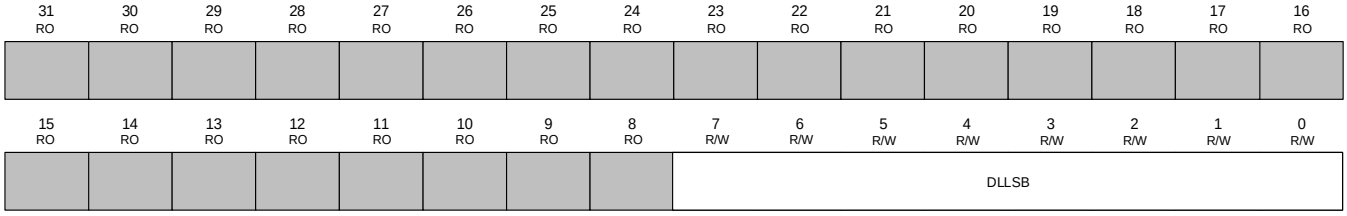


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*THR*	WO	0x0	Receiver Buffer Register By writing this register, the UART will send out an 8-bit data receiver from TX pin (LSB first).

4.7.3.3 UART_DLL - UART DIVISOR LATCH DLL

UART_DLL and **UART_DLH** registers together store the 16-bit divisor for generation of the baud clock in the baud rate generator. **UART_DLL** stores the least significant part of the divisor. Note that **UART_DLL** and **UART_DLH** may only be accessed when the **UART_LCR**.*DLAB* is set and the UART is not busy **UART_USR**.*BUSY* =0.

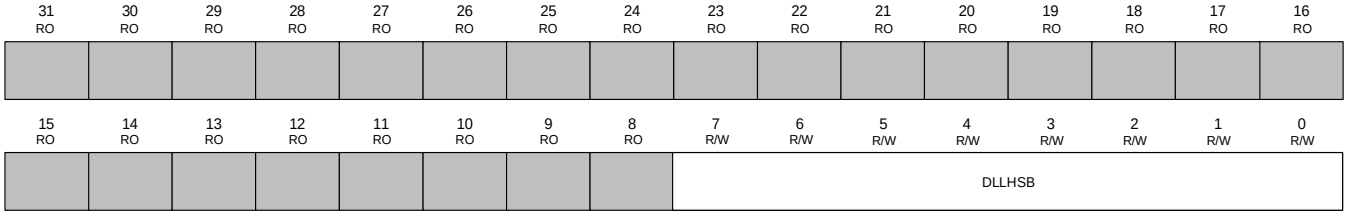


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*DLLSB*	R/W	0x0	UART Divisor Latch LSB Associated with UART_DLH register to determine the UART baud rate.

4.7.3.4 UART_DLH - UART DIVISOR LATCH DLH

UART_DLH stores the most significant part of the divisor

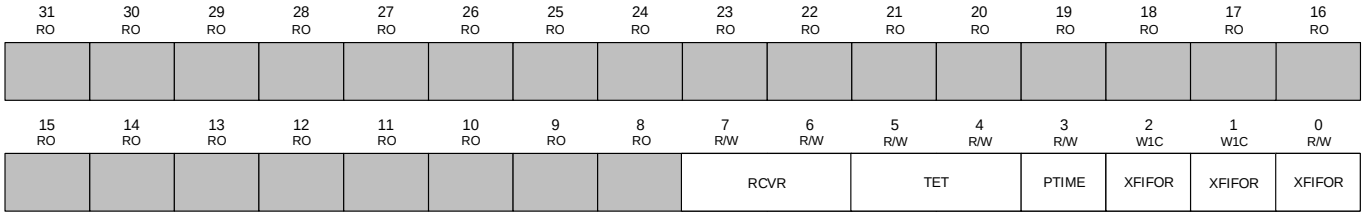


Offset: 0x0004

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*DLHSB*	R/W	0x0	UART Divisor Latch HSB Associated with UART_DLL register to determine the UART baud rate.

4.7.3.5 UART_FCR - UART FIFO CONTROL REGISTER

This register is used to enable the FIFOs, clear the FIFOs, and set the FIFO trigger levels.

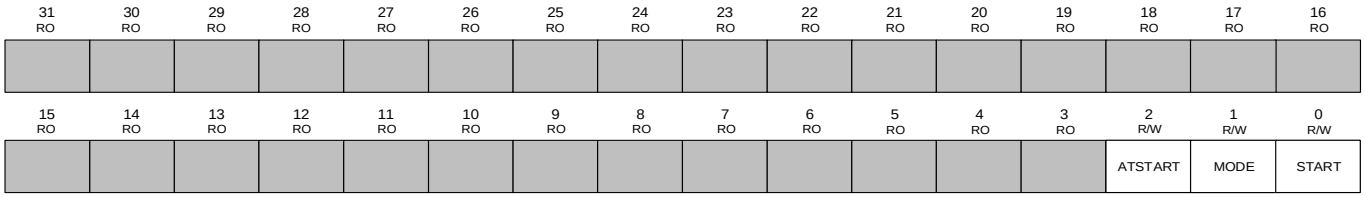


Offset: 0x0004

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:6	*RCVR*	R/W	0x0	Receive trigger level in 16-byte FIFO mode This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt (RXRDIE) is generated. 00 : 1 character in the FIFO 01 : FIFO ¼ full 10 : FIFO ½ full 11 : FIFO 2 less than full
5:4	*TET*	R/W	0x0	Transmit trigger level in 16-byte FIFO mode This is used to select the empty threshold level at which the THRE Interrupts are generated when the mode is active. 00 : FIFO empty 01 : 2 characters in the FIFO 10 : FIFO ¼ full 11 : FIFO ½ full
3	*TETES*	R/W	0	TET Effective Switch This bit determines whether the Tx empty interrupt (TXEMPIE) will be triggered by TET bit. 0: TXEMPIE will only be triggered when the FIFO is empty regardless of the value in TET 1: TXEMPIE will be set in responds to the set value in TET
2	*XFIFOR*	W1C	0	XMIT FIFO Reset When XFIFOR is set, all the byte in the transmit FIFO are cleared and treats the FIFO as empty. Note that this bit will return to 0 in the next clock cycle. 0: No FIFO transmit reset 1: Reset the Tx pointers
1	*RFIFOR*	W1C	0	RCVR FIFO Reset When RFIFOR is set, all the byte in the receiver FIFO are cleared and treats the FIFO as empty. Note that this bit will return to 0 in the next clock cycle. 0: No FIFO receive reset 1: Reset the Rx pointers.
0	*FIFOE*	R/W	0	FIFO Enable 0: UART FIFO Disable. 1: UART FIFO Enable.

4.7.3.6 UART_ACR - UART AUTO BAUD RATE CONTROL REGISTER

During the serial transmission rate measuring user input clock / data rate, the entire measurement process is controlled by this register.

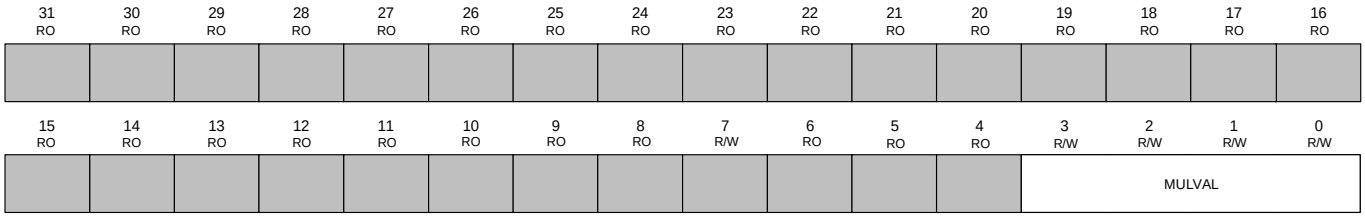


Offset: 0x0008

Bits	Name	Type	Reset	Description
31:3	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*ATSTART*	R/W	0	Auto-Baud rate detection restart (Auto) This bit is set to enable baud rate detection restart one more time after the first failure of auto-baud rate detection. 0 : No re-start detection 1 : Restart baud rate detection after first timeout (Counter is restarted by next falling edge)
1	*MODE*	R/W	0	Auto-Baud Mode Select (refer to Auto-Baud Rate) 0 : Mode 0 1 : Mode 1
0	*START*	R/W	0	Auto-Baud Rate Detection Start After the Auto-Baud feature, this bit is automatically cleared 0 : Disable Auto-Baud rate detection 1 : Enable Auto-Baud rate detection

4.7.3.7 UART_FDR - UART FRACTIONAL DIVIDER REGISTER

UART Fractional Divider Register is used to generate baud rate clock prescale, and the user can freely read and write to the register. The prescale using APB clock and according to the requirements specified in decimal generates an output clock.

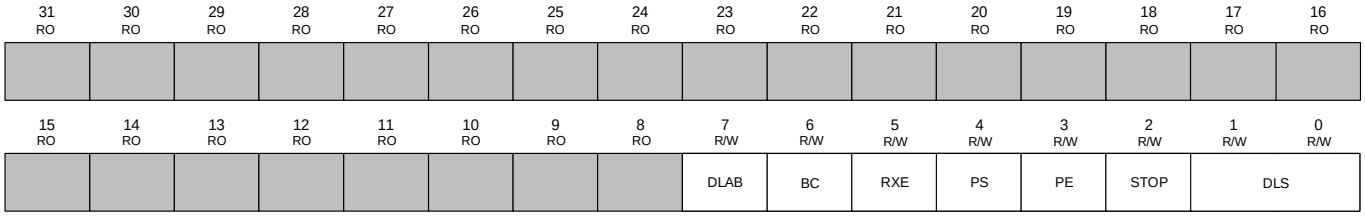


Offset: 0x000C

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:0	*MULVAL*	R/W	0x0	Prescale Multiplier Value Associated value with UART_DLL and UART_DLH to determine the decimal part of the UART baud rate, only can be written at the UART_LCR.*DLAB* = '1'.

4.7.3.8 UART_LCR - UART LINE CONTROL REGISTER

This register is used to specify the asynchronous data communication format.

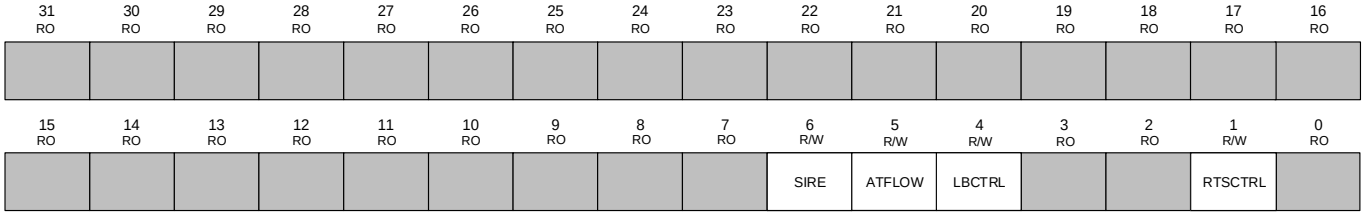


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	*DLAB*	R/W	0	Divisor Latch Access Bit This bit is writeable only when UART is not busy and is used to enable reading and writing of the Divisor Latch register (UART_DLL and UART_DLH). 0 : Disable access to the divisor latches 1 : Enable access to the divisor latches
6	*BC*	R/W	0	Break Control Bit. This is used to cause a break condition to be transmitted to the receiving device. 0 : No TX break condition 1 : The serial data output (Tx) is forced to the Spacing State (logic 0).
5	*RXE*	R/W	0	UART Receiver Enable This bit is used to enable UART receiver after user is done with the setting of UART. 0 : Disable UART receiver 1 : Enable UART receiver
4	*PS*	R/W	0	Parity Select 0 : Odd Parity check 1 : Even Parity check
3	*PE*	R/W	0	Parity Enable 0 : Disable Parity 1 : Enable Parity
2	*STOP*	R/W	0	Number of stop bits This is used to select the number of stop bits per character that the peripheral transmits and receives. 0 : 1 Stop bit 1 : 2 Stop bit(if LCR[1:0]=00, it will be 1.5 Stop bit)
1:0	*DLS*	R/W	0x0	Data Length Select 00 : 5-bit data format 01 : 6-bit data format 10 : 7-bit data format 11 : 8-bit data format

4.7.3.9 UART_MCR - UART MODEM CONTROL REGISTER

MCR enabled Modem loopback mode and controls the Modem output signal.

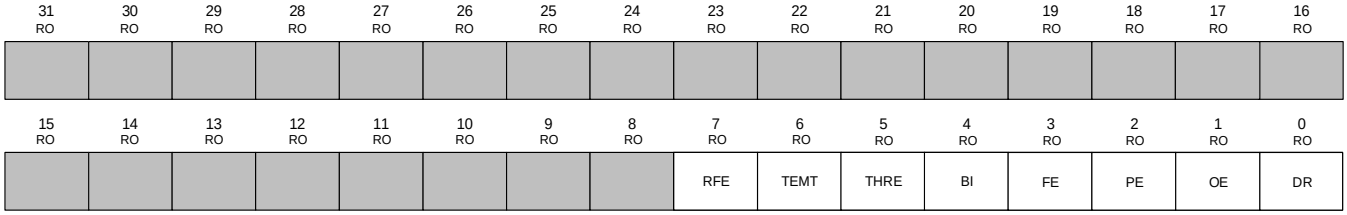


Offset: 0x0014

Bits	Name	Type	Reset	Description
31:7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6	*SIRE*	R/W	0	SIR Mode Enable 0: Disable IrDA SIR Mode. 1: Enable IrDA SIR Mode. While in this mode, the UARTx_TXD and UARTx_RXD are routed to the infrared encoder/decoder.
5	*ATFLOW*	R/W	0	Auto Flow Control Enable When FIFOs are enabled and this bit is set, Auto Flow Control features are enabled as described in Auto-Flow Control. 0: Disable Auto-Flow-Control 1: Enable Auto-Flow-Control
4	*LBCTRL*	R/W	0	LoopBack Enable This is used to put the UART into a diagnostic mode for test purpose. If operating in UART mode, data on the UARTx_TXD line is held high, while serial data output is looped back to the UARTx_RXD line, internally. In this mode, all the interrupts are fully functional. If operating in infrared mode, data on the UARTx_TXD with IrDA line is held low, while serial data output is inverted and looped back to the UARTx_RXD line. 0: Disable loopback Mode 1: Enable loopback Mode
3:2	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*RTSCTRL*	R/W	0	RTSn Control If ATFLOW = 1, then the RTSn will be controlled by Rx threshold. If ATFLOW = 0, user can control RTSn by this bit.
0	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.7.3.10 UART_LSR - UART LINE STATUS REGISTER

This register provides the status of data transfer between UART and CPU.



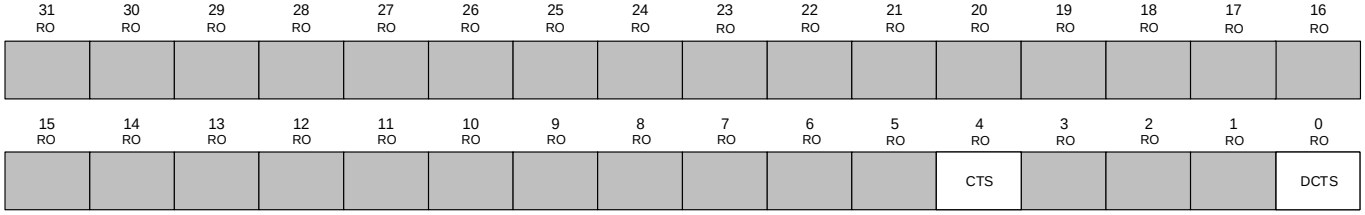
Offset: 0x0018

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	*RFE*	RO	0	Receiver FIFO Data Error This is used to indicate if there is at least one parity error, framing error, or break indication in the FIFO. 0 : No data error in RX FIFO 1 : Data error in RX FIFO
6	*TEMT*	RO	1	Transmitter Empty UART_THR and TSR empty. 0 : Either the THR or TSR contains a data character. 1 : UART_THR and TSR are both empty. In FIFO mode, this bit is set to 1 whenever the transmitter FIFO and TSR are both empty. Refer to UART Block Diagram.
5	*THRE*	RO	1	Transmit Holding Register Empty UART_THR empty. This bit indicates that the UART is ready to accept new character for transmission. This bit will trigger an interrupt to processor when the THR interrupt is set enabled. 0 : UART_THR load data from CPU 1 : A character is transferred from the UART_THR into the TSR. In FIFO mode, this bit is set to 1 when the transmitter FIFO is empty; it is cleared when at least 1 byte is written to the transmitter FIFO.
4	*BI*	RO	0	Break Interrupt This bit is used to indicate the detection of a break sequence on the serial input data. Reading UART_LSR will clears this bit. 0 : No break condition 1 : The receiver received a break signal. (UARTx_RXD was a logic 0 for one character frame time). In the FIFO mode, only one break character is loaded into the FIFO.
3	*FE*	RO	0	Framing Error When the received characters stop bit is a logic 0(i.e. the receiver did not have a valid stop bit), a framing error occurs. Reading this register will clears the bit. Framing error detection time depends on the UART_FCR.*FIFOE*. When detecting a frame error, RX will attempt to resynchronize with the data by assuming the error to be the start bit of the next character and continue receiving the other bit. In FIFO mode, this error is associated with the character at the top of the FIFO. 0 : No framing error 1 : Framing error

Bits	Name	Type	Reset	Description
2	*PE*	RO	0	Parity Error When the receive character does not have correct parity information and is suspect, a parity error occurs. Reading UART_LSR will clears this bit. In FIFO mode, this error is associated with the character at the top of the FIFO. 0: No parity error. 1: Parity error.
1	*OE*	RO	0	Overrun Error This error occurs if a new data character was received before the previous data was read. Reading UART_LSR will clear this bit. In FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. The data in the FIFO is retained and the data in the UART_RBR is lost. In the non-FIFO mode, the *OE* bit is set when a new character arrives in the receiver before the previous character was read from the UART_RBR. When this happens, the data in the UART_RBR is overwritten. 0: No overrun error 1: Overrun error
0	*DR*	RO	0	Data Ready When UART_RBR contains unread characters, this bit will be set; When the UART_RBR FIFO is empty, this bit will be cleared. 0: No data in UART_RBR or receiver FIFO 1: Data has been received and is save in the UART_RBR or receiver FIFO.

4.7.3.11 UART_MSR - UART MODEM STATUS REGISTER

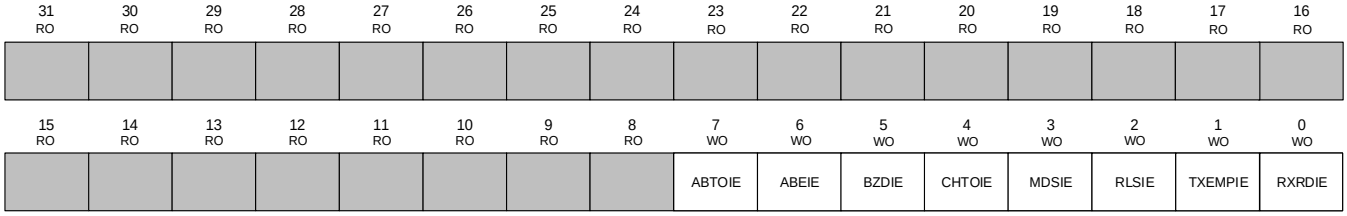
This is a read-only register that provides Modem input signals status information. Read this register clears the **UART_MSR.**DCTS. Note that modem signal does not have a direct impact on the UART operation, it only helps software to achieve Modem signal operation.



Offset: 0x001C

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*CTS*	RO	0x0	Clear To Send. This bit is the complement of CTSn. When CTSn is asserted, it is an indication that the modem or data set is ready to exchange data with UART. 0 : CTSn input is de-asserted 1 : CTSn input is asserted
3:1	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*DCTS*	RO	0x0	Delta Clear To Send This is used to indicate that the modem control line CTSn has changed since the last time the MSR was read. 0 : Modem status change is not detected on the input CTS 1 : Modem status change is detected on the input CTS

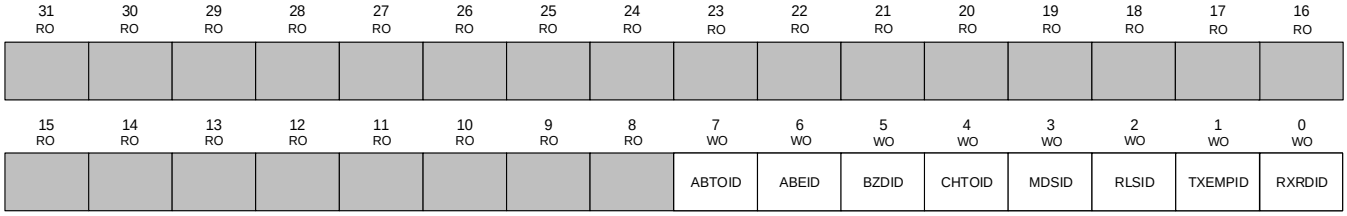
4.7.3.12 UART_IER - UART INTERRUPT ENABLE REGISTER



Offset: 0x0020

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	*ABTOIE*	WO	0	Auto-Baud rate detection Timeout Interrupt Enable 1: Interrupt is enabled for auto-baud rate detection occurs.
6	*ABEIE*	WO	0	UART Auto-Baud rate detection End Interrupt Enable 1: Interrupt is enabled for auto-baud rate detection is end.
5	*BZDIE*	WO	0	UART Busy Interrupt Enable 1: Interrupt is enabled for UART is busy.
4	*CHTOIE*	WO	0	UART Character Timeout Interrupt Enable 1: Interrupt is enabled for character timeout occurs.
3	*MDSIE*	WO	0	UART Modem Status Interrupt Enable 1: Interrupt is enabled for modem status
2	*RLSIE*	WO	0	UART Rx Line Status Interrupt enable. 1: Interrupt is enable for receiver line status
1	*TXEMPIE*	WO	0	UART Tx Empty Interrupt Enable 1: Interrupt is enabled for Transmit Holding Register is empty
0	*RXRDIE*	WO	0	UART Receive Data available Interrupt Enable. 1: Interrupt is enabled for receive data is available.

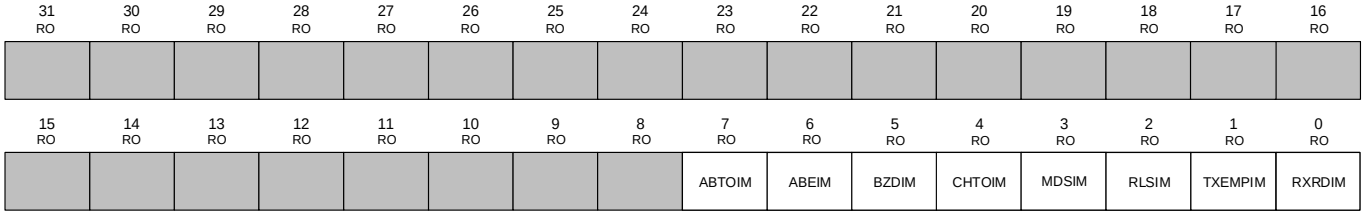
4.7.3.13 UART_IDR - UART INTERRUPT DISABLE REGISTER



Offset: 0x0024

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	*ABTOID*	WO	0	Auto-Baud rate detection Timeout Interrupt Disable 1: Interrupt is disabled for auto-baud rate detection occurs.
6	*ABEID*	WO	0	UART Auto-Baud rate detection End Interrupt Disable 1: Interrupt is disabled for auto-baud rate detection is end.
5	*BZDID*	WO	0	UART Busy Interrupt Disable 1: Interrupt is disabled for UART is busy.
4	*CHTOID*	WO	0	UART Character Timeout Interrupt Disable 1: Interrupt is disabled for character timeout occurs.
3	*MDSID*	WO	0	UART Modem Status Interrupt Disable 1: Interrupt is disabled for modem status
2	*RLSID*	WO	0	UART Rx line status Interrupt Disable. 1: Interrupt is disabled for receiver line status
1	*TXEMPID*	WO	0	UART Tx Empty Interrupt Disable 1: Interrupt is disabled for Transmit Holding Register is empty
0	*RXRDID*	WO	0	UART Receive Data available Interrupt Disable. 1: Interrupt is disabled for receive data is available.

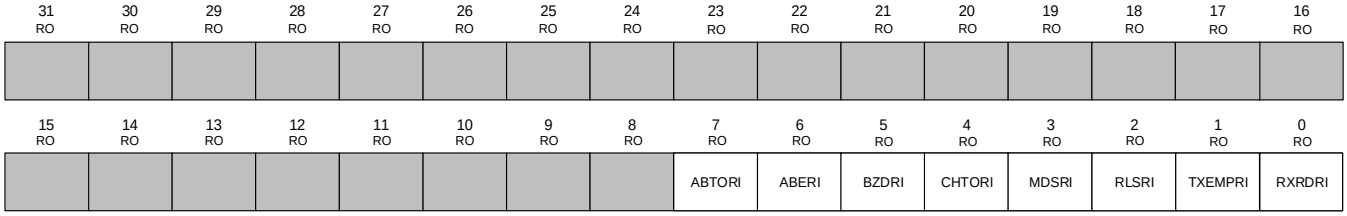
4.7.3.14 UART_IMR - UART INTERRUPT MASK STATUS REGISTER



Offset: 0x0028

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	*ABTOIM*	RO	0	Auto-Baud rate detection Timeout Interrupt Mask status 0: Auto-Baud rate detection timeout interrupt will be masked. 1: Auto-Baud rate detection timeout interrupt is enabled.
6	*ABEIM*	RO	0	UART Auto-Baud rate detection End Interrupt Mask status 0: Auto-Baud rate detection end interrupt will be masked. 1: Auto-baud rate detection interrupt is enabled.
5	*BZDIM*	RO	0	UART Busy Interrupt Mask status 0: UART busy interrupt will be masked. 1: UART busy interrupt is enabled.
4	*CHTOIM*	RO	0	UART Character Timeout Interrupt Mask status 0: UART character timeout interrupt will be masked. 1: UART character timeout interrupt is enabled.
3	*MDSIM*	RO	0	UART Modem Status Interrupt Mask status 0: Modem status interrupt will be masked. 1: Modem status interrupt is enabled.
2	*RLSIM*	RO	0	UART Rx line status Interrupt Mask status. 0: Rx line status interrupt will be masked. 1: Rx line status is enabled.
1	*TXEMPIM*	RO	0	UART Tx Empty Interrupt Mask status 0: Tx empty interrupt will be masked. 1: Tx empty interrupt is enabled.
0	*RXRDIM*	RO	0	UART Receive Data available Interrupt Mask status. 0: UART receive data available interrupt will be masked. 1: UART receive data available interrupt is enabled.

4.7.3.15 UART_RIS - UART RAW INTERRUPT STATUS REGISTER

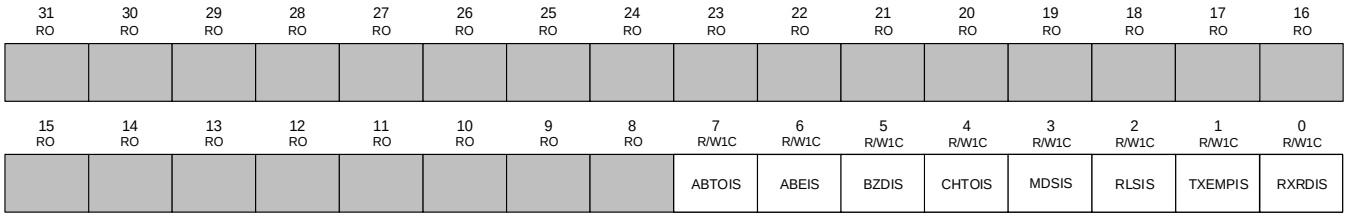


Offset: 0x002C

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	*ABTORI*	RO	0	Auto-Baud rate detection Timeout Raw Interrupt status 0: No interrupt 1: Auto-Baud rate detection timeout interrupt has occurred
6	*ABERI*	RO	0	UART Auto-Baud rate detection End Raw Interrupt status 0: No interrupt. 1: Auto-baud rate detection interrupt has occurred
5	*BZDRI*	RO	0	UART Busy Raw Interrupt status 0: No interrupt. 1: UART busy interrupt has occurred.
4	*CHTORI*	RO	0	UART Character Timeout Raw Interrupt status 0: No interrupt. 1: UART character timeout interrupt has occurred.
3	*MDSRI*	RO	0	UART Modem Status Raw Interrupt status 0: No interrupt. 1: Modem status interrupt has occurred.
2	*RLSRI*	RO	0	UART Rx line status Raw Interrupt status 0: No interrupt. 1: Rx line status has occurred.
1	*TXEMPRI*	RO	1	UART Tx Empty Raw Interrupt status 0: No interrupt. 1: Tx empty interrupt has occurred.
0	*RXRDRI*	RO	0	UART Receive Data available Raw Interrupt status 0: No interrupt. 1: UART receive data available interrupt has occurred.

4.7.3.16 UART_ISC - UART INTERRUPT STATUS AND CLEAR REGISTER

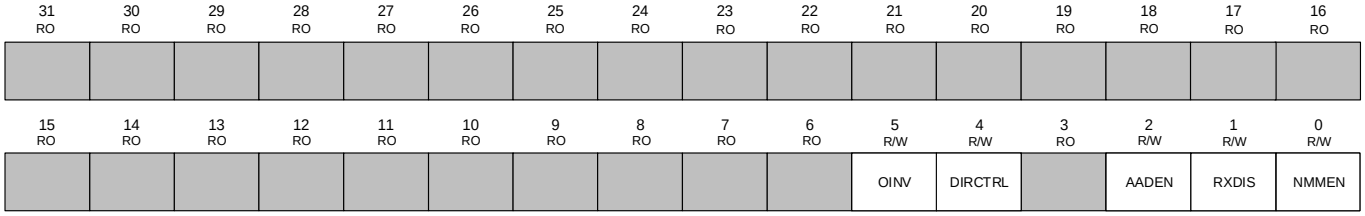
Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	*ABTOIS*	R/W1C	0	Auto-Baud rate detection Timeout Interrupt Status and clear 0: Auto-Baud rate detection has no timeout or the interrupt is masked. 1: Auto-Baud rate detection timeout Interrupt has been signaled.
6	*ABEIS*	R/W1C	0	UART Auto-Baud rate detection End Interrupt Status and clear 0: Auto-Baud rate detection has not end or the interrupt is masked. 1: Auto-baud rate detection interrupt has been signaled.
5	*BZDIS*	R/W1C	0	UART Busy Interrupt Status and clear 0: UART is not busy or the interrupt is masked. 1: UART busy interrupt has been signaled.
4	*CHTOIS*	R/W1C	0	UART Character Timeout Interrupt Status and clear 0: UART character has no timeout or the interrupt is masked. 1: UART character timeout interrupt has been signaled.
3	*MDSIS*	R/W1C	0	UART Modem Status Interrupt Status and clear 0: UART Modem status has no interrupt or the interrupt is masked. 1: Modem status interrupt has been signaled.
2	*RLSIS*	R/W1C	0	UART Rx line status Interrupt Status and clear 0: UART Rx line status has no interrupt or the interrupt is masked. 1: Rx line status has been signaled.
1	*TXEMPIS*	R/W1C	0	UART Tx Empty Interrupt Status and clear 0: UART Tx is not empty or the interrupt is masked. 1: Tx empty interrupt has been signaled.
0	*RXRDIS*	R/W1C	0	UART Receive Data available Interrupt Status and clear 0: UART receive data is not available or the interrupt is masked. 1: UART receive data available interrupt has been signaled.

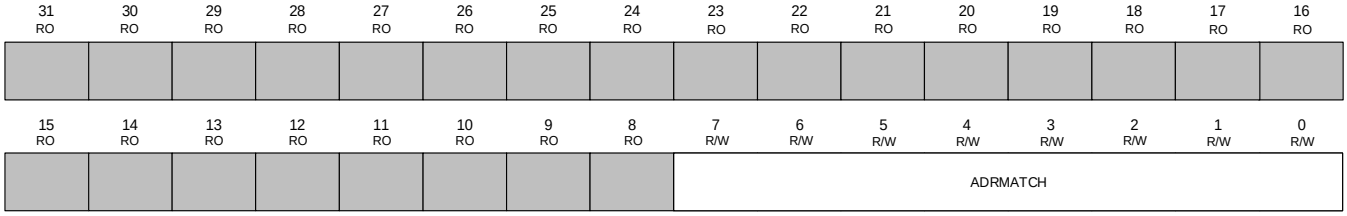
4.7.3.17 UART_485CTRL - UART RS485 CONTROL REGISTER



Offset: 0x0034

Bits	Name	Type	Reset	Description
31:6	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*OINV*	R/W	0	This bit retains the RTSn (or CTSn) polarity direction control signal on pin. 0: When the transmitter has data to send, direction control pin will be driven to a logic "0". After the last data bit is sent, this bit will be driven to a logic "1". 1: When the transmitter has data to send, direction control pin will be driven to a logic "1". After the last data bit is sent, this bit will be driven to a logic "0".
4	*DIRCTRL*	R/W	0	Direction Control enable 0: Disable automatic direction control. 1: Enable automatic direction control.
3	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*AADEN*	R/W	0	Automatic Address Detection (AAD) Enable Note: It can't be active with RS-485 NMM operation 0: Disable AAD 1: Enable AAD
1	*RXDIS*	R/W	0	Receiver Disable 0: Enable receiver 1: Disable receiver
0	*NMMEN*	R/W	0	RS-485 / EIA-485 Normal Multi-Mode (NMM) Enable. Note that NMM can't be active with RS-485 AAD operation mode. 0 : Disable NMM 1 : Enable NMM

4.7.3.18 UART_485ADX - UART RS485 ADDRESS MATCH REGISTER

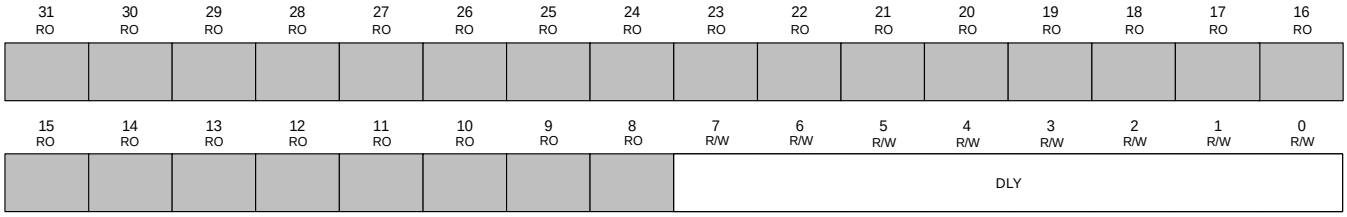


Offset: 0x0038

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*ADRMATCH*	R/W	0x0	Contains the Address Match value

4.7.3.19 UART_RS485DLY - UART RS485 DIRECTION DELAY REGISTER

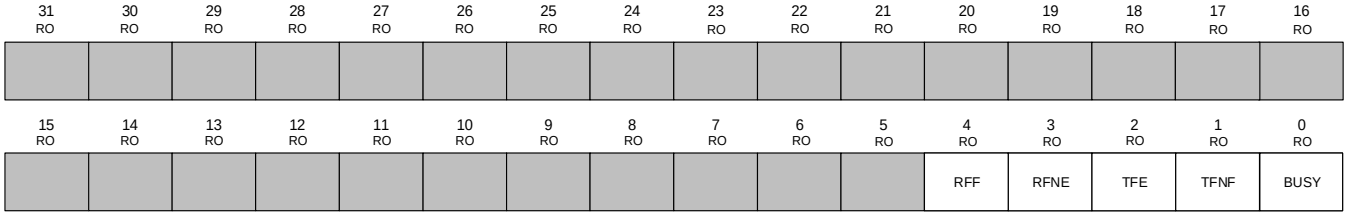
The user may program the 8-bit **UART_485DLY** register with a delay between the last stop bit leaving the TxFIFO and the de-assertion of RTSn (or CTSn). This delay time is in periods of the baud clock. Any delay time from 0 to 255 bit times may be programmed



Offset: 0x003C

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*DLY*	R/W	0x0	Contains a direction control (RTSn or CTSn) Delay value. This register works in conjunction with an 8-bit counter.

4.7.3.20 UART_USR - UART STATUS REGISTER

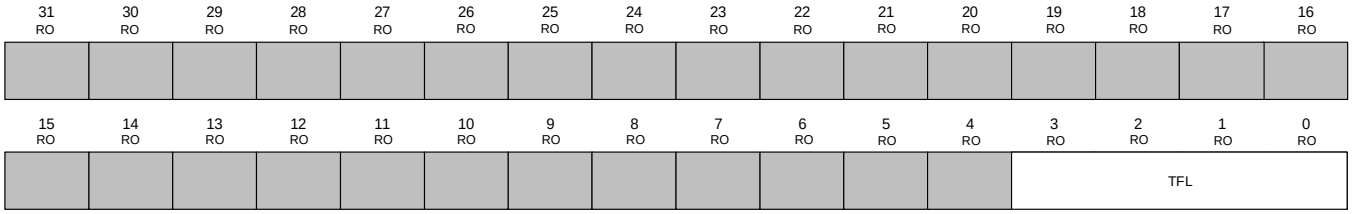


Offset: 0x007C

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*RFF*	RO	0	Receive FIFO Full This bit is cleared when the Rx FIFO is no longer full. 0: Receive FIFO not full 1: Receive FIFO full
3	*RFNE*	RO	0	Receive FIFO Not Empty This bit is cleared when the Rx FIFO is empty. 0: Receive FIFO is empty 1: Receive FIFO is not empty
2	*TFE*	RO	1	Transmit FIFO Empty This bit is cleared when the Tx FIFO is no longer empty 0: Transmit FIFO is not empty 1: Transmit FIFO is empty
1	*TFNF*	RO	1	Transmit FIFO Not Empty This bit is cleared when the Tx FIFO is full. 0: Transmit FIFO is full 1: Transmit FIFO is not full
0	*BUSY*	RO	0	UART Busy 0: UART is IDLE or inactive. 1: UART is busy(A serial transfer is in progress)

4.7.3.21 UART_TFL - UART TRANSMIT FIFO LEVEL REGISTER

Transmit FIFO level register contains a number of data, which transfer in the FIFO to be transmitted.

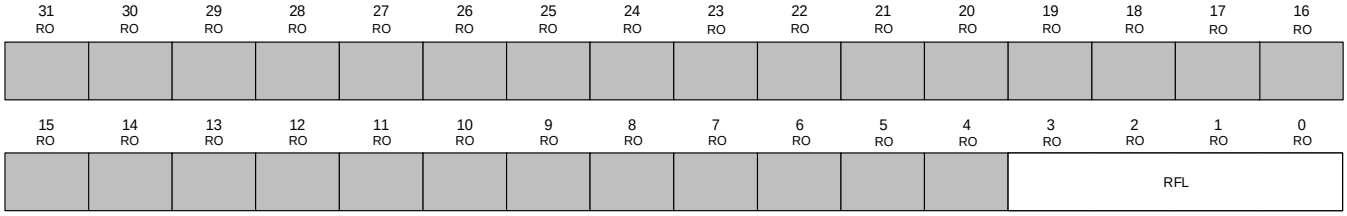


Offset: 0x0080

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:0	*TFL*	RO	0x0	Transmit FIFO Level This is used to indicate the number of data entries in the transmit FIFO.

4.7.3.22 UART_RFL - UART RECEIVE FIFO LEVEL REGISTER

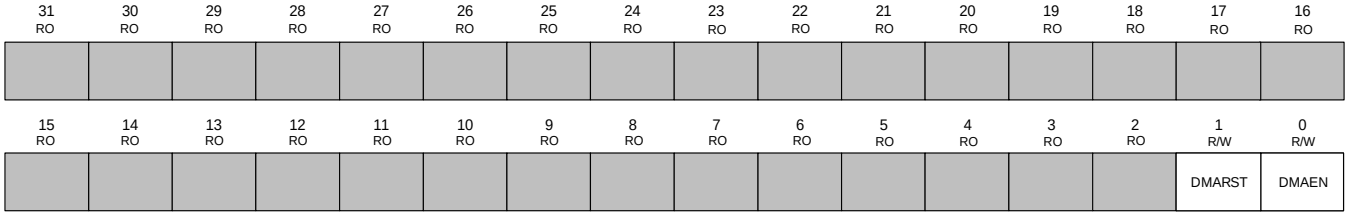
Receive FIFO level register contains the number of valid data in the receive FIFO.



Offset: 0x0084

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:0	*RFL*	RO	0x0	Receive FIFO Level This is used to indicate the number of data entries in the Receive FIFO.

4.7.3.23 UART__DMACTL - UART DMA CONTROL REGISTER



Offset: 0x0084

Bits	Name	Type	Reset	Description
31:2	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*DMARST*	R/W	0	DMA Reset 1: DMA Request cannot transmit to DMA controller
0	*DMAEN*	R/W	0	DMA Enable 1: Enable DMA

4.8 ANALOG TO DIGITAL CONVERTER (ADC)

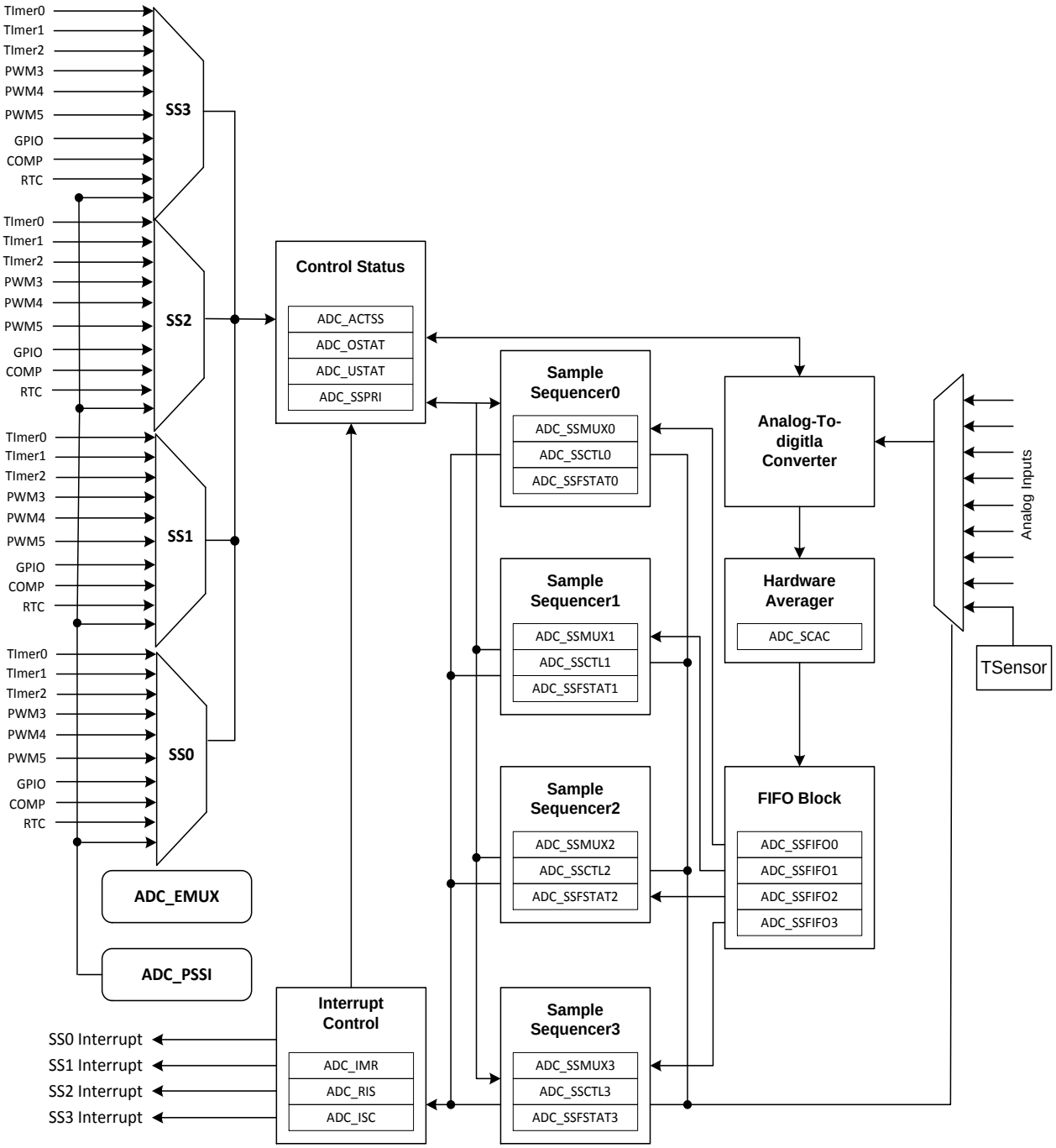
An analog-to-digital converter (ADC) is a peripheral that converts a continuous analog voltage to a discrete digital number. The PT32C302 contains one 12-bit successive approximation analog-to-digital converter (SAR A/D converter) with 8 external input channels, plus an internal temperature sensor. Each ADC module supports four programmable sequencers which allow for the sampling of multiple analog input sources without controller intervention. A / D converter supports three operating modes: single-mode, single-cycle scan mode and continuous scan mode. Each sample sequence provides flexible programming with fully configurable input source, trigger events, and interrupt.

The ADC module provides the following features:

- Analog input voltage range: 0 to reference values (up to 3.3V).
- Up to 8 single-end analog input channels or 4 differential analog input channels.
- Support programmable sampling time (with ADC_CLK units)
- Support average mode.
- Support PGA control mode.
- On-chip internal temperature sensor, support temperature sensing control mode.
- Maximum ADC clock frequency is 48 MHz, the sampling time of each conversion time is 20 clock + input impedance of the decision.
- Flexible trigger control
  - Controller (software)
  - Timer 0/1/2/3/4/5
  - Comparator
  - PWM 0/1/2
  - GPIO
  - RTC
- A/D conversion can perform as
  - One time on a specified channel.
  - One cycle on all specified channels with the sequence from the lowest numbered channel to the highest numbered channel.
  - Single-cycle scan until software stops A/D conversion.
- Efficient transfers using Micro Direct Memory Access Controller

4.8.1 BLOCK DIAGRAM

Figure 4.8-1: ADC Block Diagram



4.8.2 FUNCTIONAL DESCRIPTION

The PT32C302 ADC collects sample data by using a programmable sequence-based approach instead of the traditional single or double-sampling approach. Each sample sequence is a fully programmed series of consecutive samples, allowing the ADC to collect data from multiple input sources without serviced by the microprocessor. The DMA can be used to more efficiently move data from the sample sequencers without CPU intervention.

4.8.2.1 SAMPLE SEQUENCER

The sampling control and data capture is handled by the sample sequencer. All of the sequencers are identical except for the number of samples that can be captured and the depth of the FIFO. Following table shows the maximum number of samples that each sequencer can support and its corresponding FIFO depth.

Table 4.8-1: Samples and FIFO Depth of Sequencers

Sequencer	Number of Samples	FIFO Depth
SS0	8	8
SS1	4	4
SS2	4	4
SS3	1	1

For a given sample sequence, each sample is defined by bit field in the ADC Sample Sequence Input Multiplexer Select Register (**ADC_SSMUXx)** and ADC Sample Sequence Control **(ADC_SSCTLx)** registers. The **ADC_SSMUXx** fields select the input pin, while **ADC_SSCTLx** fields contain the sample control bits corresponding to parameter such as temperatures sensor selection, interrupt enable, end of sequence, and differential input mode. Sample sequencer are enabled by setting the respective *SSxEN* bit in the ADC Active Sample Sequencer **(ADC_ACTSS)** register and should be configured before enabled Sampling is then initialed by setting the *SSxI* bit in the ADC Processor Sample Sequence Initiate **(ADC_PSSI)** register.

When configuring a sample sequence, multiple used of the same input pin within the same sequence are allowed. In the **ADC_SSCTLx** register, the *IEx* bits can be set for any combination of samples, allowing interrupt to be generated after every sample in the sequence if necessary. Also the *END* bit can be set at any point within sample sequence. For example, if Sequencer 0 is used, the *END* bit can be set in the nibble associated with the fifth sample, allowing Sequencer 0 to complete execution of the sample sequence after the fifth sample.

After a sample sequence completes execution, the result data can be retrieved from the ADC Sample Sequence Result FIFO **(ADC_SSFIFOx)** registers. The FIFOs are sample circular buffers that read a single address to “pop” result data. For software debug purposes, the positions of the FIFO head and tail pointers are visible in the ADC Sample Sequence FIFO Status **(ADC_SSFSTATx)** registers along with *FULL* and *EMPTY* status flags. If a write is attempted when the FIFO is full, the write does not occur and an overflow condition is indicated. Overflow and Underflow conditions are monitored using the **ADC_OSTAT** and **ADC_USTAT.**

4.8.2.2 MODULE CLOCKING

The module is clocked the same with AHB source. The maximum support sampling rate of internal ADC which convert input analog signal with 20 ADC cycle. The maximum sample rate of ADC is 2.4MHZ with 48MHZ clock source selection. The ADC can setup to adopt lower clock source by setting *CLKDIV* field of **ADC Initial Control Register** (**ADC_INI)**.

4.8.2.3 SAMPLE PRIORITIZATION

When sampling events (triggers) happens concurrently, they are prioritized for processing by the value in the **ADC Sample Sequencer Priority** Register **(ADC_SSPRI)**. Valid priority value are in the rage of 0-3, with 0 being the highest priority and 3 being the lowest. Multiple active sample sequencer units with the same priority do not provide consistent result, so user must ensure all active sample sequencer units have a unique value.

4.8.2.4 SAMPLING EVENTS/TRIGGERS

Events/Triggering for each sample sequencer is defined in the ADC Event Multiplexer Select (ADC_EMUX) register. Trigger sources include processor software trigger (default), analog comparators, and an external signal on a GPIO specified by the GPIO, a Timer continuous sampling. The processor triggers sampling by setting the SSx bits in the ADC Processor Sample Sequence Initiate (ADC_PSSI) register.

User must be taken care when using the continuous sampling trigger. If a sequencer's priority is too high, it is possible to starve other lower priority sequencers. Generally, a sample sequencer using continuous sampling should be set to the lowest priority. Continuous sampling can be used with a digital comparator to cause an interrupt when a particular voltage is seen on an input.

4.8.2.5 INTERRUPT CONTROL

The register configurations of the sample sequencers dictate which events generate raw interrupts, but do not have control over whether the interrupt is actually sent to the interrupt controller. In PT32C302, the interrupt in ADC are controlled by a set of five registers.

- Interrupt Control (IER, IDR, IMR)

  ADC Interrupt enable register **(ADC_IER)** enables the interrupt request lines by writing a '1'. Similarly, ADC Interrupt disable register **(ADC_IDR)** disables the interrupt request lines by writing a '1'. IER and IDR are write only registers which control the masking of interrupts. The overall result of these two registers can be shown by ADC Interrupt Mask Register **(ADC_IMR)**. IMR is a read-only register using '1' or '0' to indicate if the interrupt request line is enabled/ or disabled. This register controls whether the raw interrupt signals are promoted.
- Interrupt Status Read (RIS)

  ADC Raw Interrupt Status **(ADC_RIS)** is a read-only register to show all raw interrupt signal of the module.
- Interrupt Clear (ISC)

  ADC Interrupt Status & Interrupt Clear Register **(ADC_ISC)** is used to indicate the non-masked interrupt status of the module, since only now-masked interrupts are asserted to processor. Writing a '1' to the bit in this register can clear the corresponding interrupt status or disable the interrupt by writing 1 to IDR.

4.8.2.6 AVERAGING CIRCUIT

Higher precision results can be generated using the hardware averaging circuit, however, the improved results are at the cost of throughput. Up to 64 samples can be accumulated and averaged to form a single data entry in the sequencer FIFO. Throughput is decreased proportionally to the number of samples in the averaging calculation. For example, if the averaging circuit is configured to average 16 samples, the throughput is decreased by a factor of 16.

By default the averaging circuit is off, and all data from the converter passes through to the sequencer FIFO. The averaging circuit is controlled by the ADC Averaging Channel Control (ADC_AVGC) register.

4.8.3 INITIALIZATION AND CONFIGURATION

4.8.3.1 MODULE INITIALIZATION

In order for the ADC module to be used, the PLL must be enabled and programmed. Using unsupported frequencies can cause faulty operation in the ADC module

Initialization of the ADC module is a simple process with very few steps: enabling the clock to the ADC, muxing the IO with Analog input, and reconfiguring the sample sequencer priorities.

The initialization sequence for the ADC is as follows:

1. Enable the ADC clock using the in System Control Register: Enable the bit *ADC* in APB Peripheral Gated Clock Register **(SC_GCLK_APB).**
2. Set the GPIO's GPIOx_AFRH and GPIOx_AFRL register for the ADC input pins. Refer Multiplexing Pins Function Selection to find out which GPIO pins to enable.
3. If required by the application, reconfigure the sample sequencer priorities in the **ADC_SSPRI** register. The default configuration has Sample Sequencer 0 with the highest priority and Sample Sequencer 3 as the lowest priority

4.8.3.2 SAMPLE SEQUENCER CONFIGURATION

The configuration for each sample sequencer is as follows:

1. Ensure that the sample sequencer is disabled by clearing the corresponding *SSxEN* bit in the **ADC_ACTSS** register. Programming of the sample sequencers is allowed without having them enabled. Disabling the sequencer during programming prevents erroneous execution if a trigger event were to occur during the configuration process.
2. Configure the trigger event for the sample sequencer in the **ADC_EMUX** register.
3. For each sample in the sample sequence, configure the corresponding input source in the **ADC_SSMUXx** register.
4. For each sample in the sample sequence, configure the sample control bits in the corresponding nibble in the **ADC_SSCTLx** register. When programming the last nibble, ensure that the *END* bit is set. Failure to set the *END* bit causes unpredictable behavior.
5. If interrupts are to be used, set the corresponding MASK bit in the **ADC_IER/IDR** register.
6. Enable the sample sequencer logic by setting the corresponding *SSxEN* bit in the **ADC_ACTSS** register.

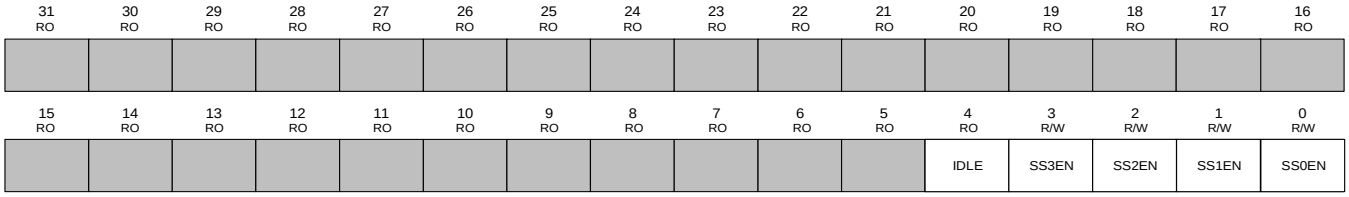
4.8.4 ADC REGISTER MAP

Base Address: 0x4001_8000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	ADC_ACTSS	R/W	0x0000_0010	ADC Active Sample Sequencer	135
0x0004	ADC_IER	WO	0x0000_0000	ADC Interrupt Enable	136
0x0008	ADC_IDR	WO	0x0000_0000	ADC Interrupt Disable	136
0x000C	ADC_IMR	RO	0x0000_0000	ADC Interrupt Mask	137
0x0010	ADC_RIS	RO	0x0000_0000	ADC Raw Interrupt Status	138
0x0014	ADC_ISC	R/W1C	0x0000_0000	ADC Interrupt Status and Clear	139
0x0018	ADC_OSTAT	R/W1C	0x0000_0000	ADC Overflow Status	140
0x001C	ADC_EMUX	R/W	0x0000_0000	ADC Event Multiplexer Select	141
0x0020	ADC_USTAT	R/W1C	0x0000_0000	ADC Underflow Status	142
0x0028	ADC_SSPRI	R/W	0x0000_3210	ADC Sample Sequencer Priority	143
0x002C	ADC_INI	R/W	0x0000_0000	ADC Initial Control	144
0x0030	ADC_PSSI	R/W	0x0000_0000	ADC Processor Sample Sequence Initiate	145
0x0034	ADC_AVGC	R/W	0x0000_0000	ADC Average Control	146
0x0038	ADC_PGAC	R/W	0x0000_0000	ADC PGA Gain Control	148
0x003C	ADC_DIFC	R/W	0x0000_0000	ADC PGA Mode Control	150
0x0040	ADC_TMPC	R/W	0x0000_0000	ADC Temperature Control	151
0x0044	ADC_FRF	W1C	0x0000_0000	ADC FIFO Refresh	151
0x0048	ADC_WAIT	R/W	0x0000_0000	ADC Wait Counter Register	152
0x004C	ADC_DMACTRL	R/W	0x0000_0000	ADC DMA Control Register	152
0x0050	ADC_SSMUX0	R/W	0x0000_0000	ADC Sample Sequence Input Multiplexer Select 0	153
0x0054	ADC_SSCTL0	R/W	0x2000_0000	ADC Sample Sequence Control 0	155
0x0058	ADC_SSFIFO0	RO	0x0000_0000	ADC Sample Sequence Result FIFO 0	157
0x005C	ADC_SSFSTAT0	RO	0x0000_0100	ADC Sample Sequence FIFO 0 Status	158
0x0070	ADC_SSMUX1	R/W	0x0000_0000	ADC Sample Sequence Input Multiplexer Select 1	159
0x0074	ADC_SSCTL1	R/W	0x0000_2000	ADC Sample Sequence Control 1	160
0x0078	ADC_SSFIFO1	RO	0x0000_0000	ADC Sample Sequence Result FIFO 1	157
0x007C	ADC_SSFSTAT1	RO	0x0000_0100	ADC Sample Sequence FIFO 1 Status	158
0x0090	ADC_SSMUX2	R/W	0x0000_0000	ADC Sample Sequence Input Multiplexer Select 2	159
0x0094	ADC_SSCTL2	R/W	0x0000_2000	ADC Sample Sequence Control 2	160
0x0098	ADC_SSFIFO2	RO	0x0000_0000	ADC Sample Sequence Result FIFO 2	157
0x009C	ADC_SSFSTAT2	RO	0x0000_0100	ADC Sample Sequence FIFO 2 Status	158
0x00B0	ADC_SSMUX3	R/W	0x0000_0000	ADC Sample Sequence Input Multiplexer Select 3	161
0x00B4	ADC_SSCTL3	R/W	0x0000_0002	ADC Sample Sequence Control 3	162
0x00B8	ADC_SSFIFO3	RO	0x0000_0000	ADC Sample Sequence Result FIFO 3	157
0x00BC	ADC_SSFSTAT3	RO	0x0000_0100	ADC Sample Sequence FIFO 3 Status	158

4.8.4.1 ADC_ACTSS - ADC ACTIVE SAMPLE SEQUENCER

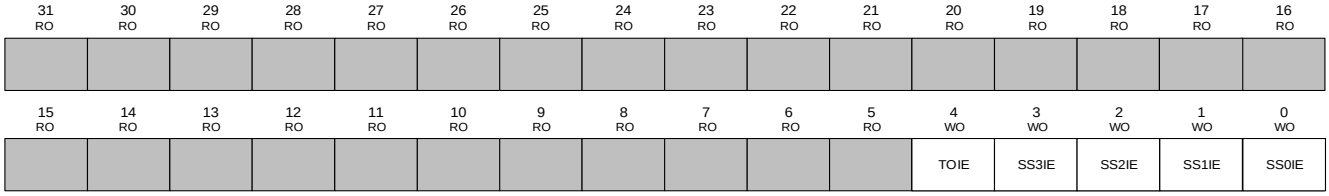
This register controls the activation of the sample sequencers. Each sample sequencer can be enabled or disabled independently.



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*IDLE*	RO	1	ADC Idle state 0: ADC is busy 1: ADC is idle
3	*SS3EN*	R/W	0	ADC Sample Sequencer 3 (SS3) Enable 0: SS3 is disabled. 1: SS3 is enabled
2	*SS2EN*	R/W	0	ADC Sample Sequencer 2 (SS2) Enable 0: SS2 is disabled. 1: SS2 is enabled
1	*SS1EN*	R/W	0	ADC Sample Sequencer 3 (SS1) Enable 0: SS1 is disabled. 1: SS1 is enabled
0	*SS0EN*	R/W	0	ADC Sample Sequencer 0 (SS0) Enable 0: SS0 is disabled. 1: SS0 is enabled

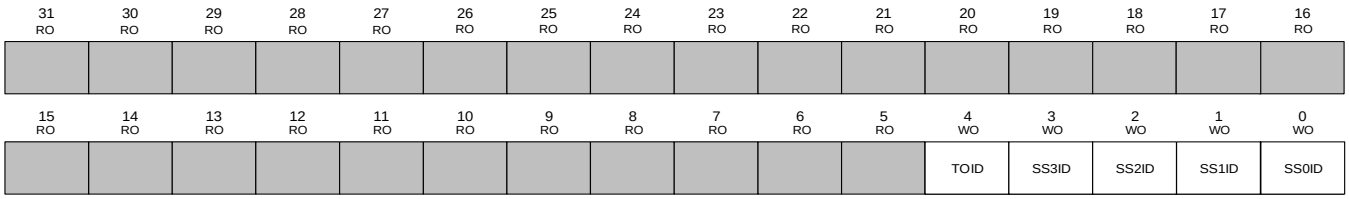
4.8.4.2 ADC_IER - ADC INTERRUPT ENABLE



Offset: 0x0004

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*TOIE*	WO	0	Timeout Interrupt Enable 1: Timeout interrupt is enabled.
3	*SS3IE*	WO	0	ADC Sample Sequencer 3 (SS3) Enable 1: SS3 interrupt is enabled
2	*SS2IE*	WO	0	ADC Sample Sequencer 2 (SS2) Enable 1: SS2 interrupt is enabled
1	*SS1IE*	WO	0	ADC Sample Sequencer 3 (SS1) Enable 1: SS1 interrupt is enabled
0	*SS0IE*	WO	0	ADC Sample Sequencer 0 (SS0) Enable 1: SS0 interrupt is enabled

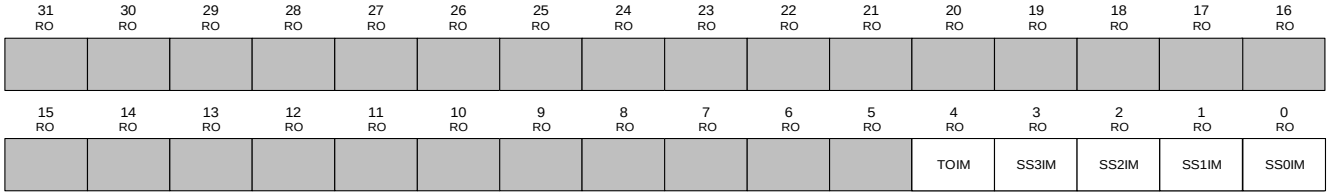
4.8.4.3 ADC_IDR - ADC INTERRUPT DISABLE



Offset: 0x0008

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*TOID*	RO	0	Timeout Interrupt Disable 1: Timeout interrupt is disabled.
3	*SS3ID*	WO	0	ADC Sample Sequencer 3 (SS3) Disable 1: SS3 interrupt is disabled
2	*SS2ID*	WO	0	ADC Sample Sequencer 2 (SS2) Disable 1: SS2 interrupt is disabled
1	*SS1ID*	WO	0	ADC Sample Sequencer 3 (SS1) Disable 1: SS1 interrupt is disabled
0	*SS0ID*	WO	0	ADC Sample Sequencer 0 (SS0) Disable 1: SS0 interrupt is disabled

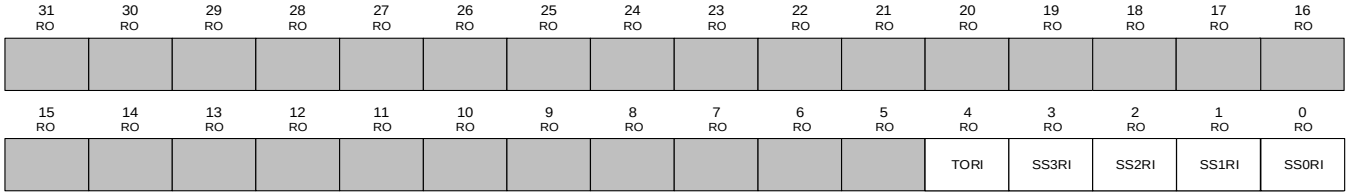
4.8.4.4 ADC_IMR - ADC INTERRUPT MASK STATUS



Offset: 0x000C

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*TOIM*	RO	0	Timeout Interrupt Mask status 0: Timeout interrupt will be masked. 1: Timeout interrupt is enabled.
3	*SS3IM*	RO	0	ADC Sample Sequencer 3 (SS3) Interrupt Mask status 0: SS3 interrupt will be masked. 1: SS3 interrupt is enabled.
2	*SS2IM*	RO	0	ADC Sample Sequencer 2 (SS2) Interrupt Mask status 0: SS2 interrupt will be masked. 1: SS2 interrupt is enabled.
1	*SS1IM*	RO	0	ADC Sample Sequencer 3 (SS1) Interrupt Mask status 0: SS1 interrupt will be masked. 1: SS1 interrupt is enabled.
0	*SS0IM*	RO	0	ADC Sample Sequencer 0 (SS0) Interrupt Mask status 0: SS0 interrupt will be masked. 1: SS0 interrupt is enabled.

4.8.4.5 ADC_RIS - ADC RAW INTERRUPT STATUS

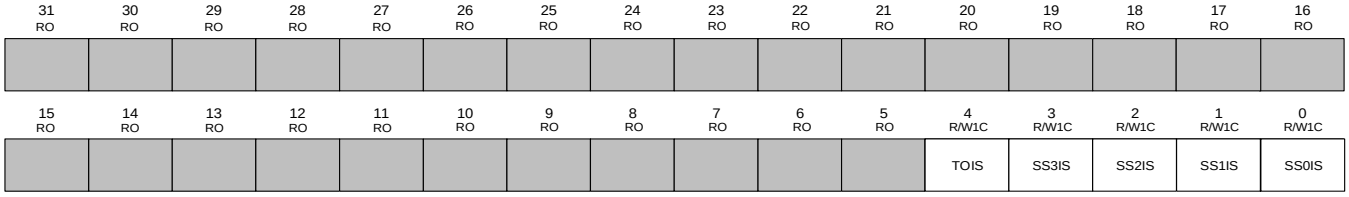


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*TORI*	RO	0	Timeout Raw Interrupt status 0: No interrupt. 1: Timeout interrupt is asserting.
3	*SS3RI*	RO	0	ADC Sample Sequencer 3 (SS3) Raw Interrupt status 0: No interrupt. 1: SS3 interrupt is asserting.
2	*SS2RI*	RO	0	ADC Sample Sequencer 2 (SS2) Raw Interrupt status 0: No interrupt. 1: SS2 interrupt is asserting.
1	*SS1RI*	RO	0	ADC Sample Sequencer 3 (SS1) Raw Interrupt status 0: No interrupt. 1: SS1 interrupt is asserting.
0	*SS0RI*	RO	0	ADC Sample Sequencer 0 (SS0) Raw Interrupt status 0: No interrupt. 1: SS0 interrupt is asserting.

4.8.4.6 ADC_ISC - ADC INTERRUPT STATUS AND CLEAR

Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt.

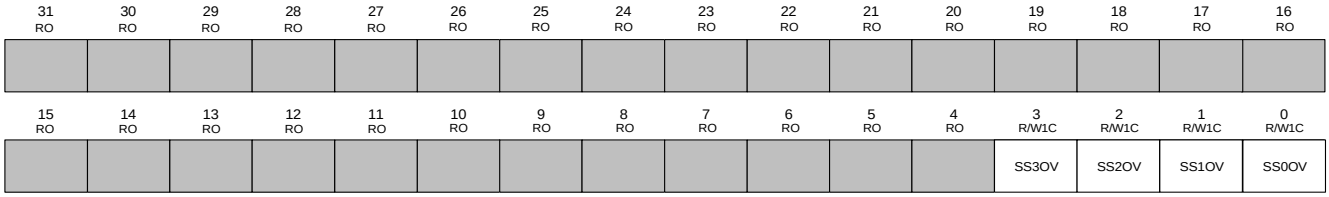


Offset: 0x0014

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*TOIS*	R/W1C	0	Timeout Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Timeout interrupt has been signaled.
3	*SS3IS*	R/W1C	0	ADC Sample Sequencer 3 (SS3) Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: SS3 interrupt has been signaled.
2	*SS2IS*	R/W1C	0	ADC Sample Sequencer 2 (SS2) Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: SS2 interrupt has been signaled.
1	*SS1IS*	R/W1C	0	ADC Sample Sequencer 3 (SS1) Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: SS1 interrupt has been signaled.
0	*SS0IS*	R/W1C	0	ADC Sample Sequencer 0 (SS0) Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: SS0 interrupt has been signaled.

4.8.4.7 ADC_OSTAT - ADC OVERFLOW STATUS

This register indicates overflow conditions in the sample sequencer FIFOs. Once the overflow condition has been handled by software, the condition can be cleared by writing 1 to the corresponding bit position.

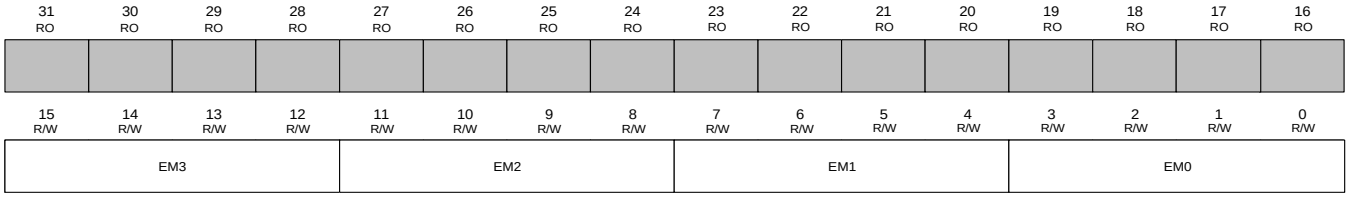


Offset: 0x0018

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	SS3OV	R/W1C	0	ADC Sample Sequencer 3 (SS3) FIFO Overflow status 0: The FIFO has not overflowed. 1: SS3 FIFO has bit an overflow condition where the FIFO is full and a write was requested. When an overflow is detected, the most recent write is dropped.
2	SS2OV	R/W1C	0	ADC Sample Sequencer 2 (SS2) FIFO Overflow status 0: The FIFO has not overflowed. 1: SS2 FIFO has bit an overflow condition where the FIFO is full and a write was requested. When an overflow is detected, the most recent write is dropped.
1	SS1OV	R/W1C	0	ADC Sample Sequencer 3 (SS1) FIFO Overflow status 0: The FIFO has not overflowed. 1: SS1 FIFO has bit an overflow condition where the FIFO is full and a write was requested. When an overflow is detected, the most recent write is dropped.
0	SS0OV	R/W1C	0	ADC Sample Sequencer 0 (SS0) FIFO Overflow status 0: The FIFO has not overflowed. 1: SS0 FIFO has bit an overflow condition where the FIFO is full and a write was requested. When an overflow is detected, the most recent write is dropped.

4.8.4.8 ADC_EMUX - ADC EVENT MULTIPLEXER SELECT

The **ADCEMUX** selects the event (trigger) that initiates sampling for each sample sequencer. Each sample sequencer can be configured with a unique trigger source.

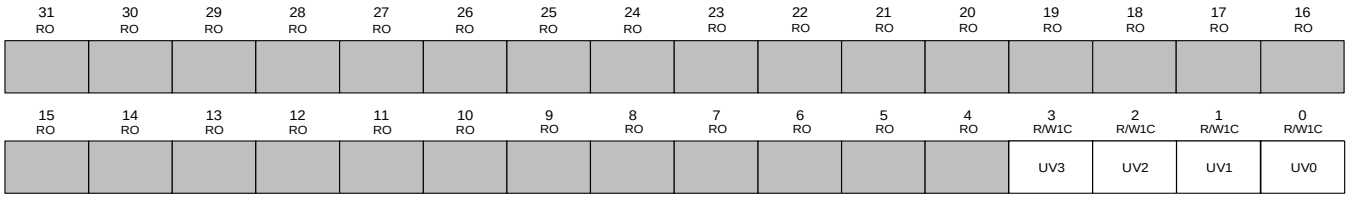


Offset: 0x001C

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:12	*EM3*	R/W	0	SS3 Trigger Select This field selects the trigger source for SS3. Same trigger selection table as in EM0 but only for SS3
11:8	*EM2*	R/W	0	SS2 Trigger Select This field selects the trigger source for SS2. Same trigger selection table as in EM0 but only for SS2
7:4	*EM1*	R/W	0	SS1 Trigger Select This field selects the trigger source for SS1. Same trigger selection table as in EM0 but only for SS1
3:0	*EM0*	R/W	0	SS0 Trigger Select This field selects the trigger source for SS0. Trigger selection table is listed as follows 0x0: Controller (Default) 0x1: GPIO 0x2: Always (Continuously sample) 0x3: RTC 0x4: Timer 0 0x5: Timer 1 0x6: Timer 2 0x7: Timer 3 0x8: Timer 4 0x9: Timer 5 0xA: reserved 0xB: reserved 0xC: Analog Comparator 0 0xD: Analog Comparator 1 0xE: Analog Comparator 2 0xF: Analog Comparator 3

4.8.4.9 ADC_USTAT - ADC UNDERFLOW STATUS

This register indicates underflow conditions in the sample sequencer FIFOs. The corresponding underflow condition is cleared by writing 1 to the relevant bit position.

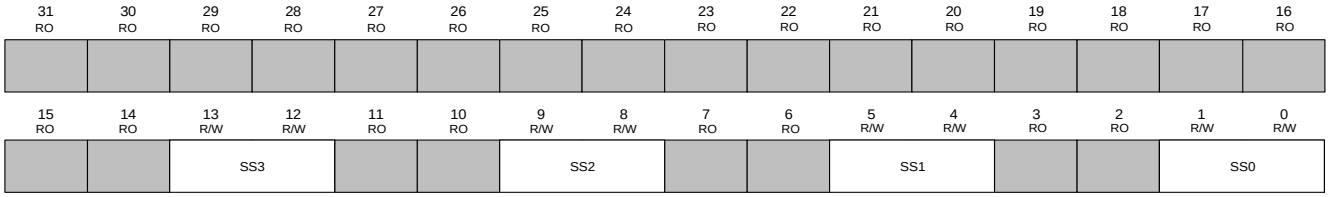


Offset: 0x0020

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x000_0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*UV3*	R/W1C	0	ADC Sample Sequencer 3 (SS3) FIFO Underflow status 0: The FIFO has not underflowed. 1: SS3 FIFO has bit an underflow condition where the FIFO is empty and a read was requested. The problematic read does not move the FIFO pointers, and 0s are returned
2	*UV2*	R/W1C	0	ADC Sample Sequencer 2 (SS2) FIFO Underflow status 0: The FIFO has not underflowed. 1: SS2 FIFO has bit an underflow condition where the FIFO is empty and a read was requested. The problematic read does not move the FIFO pointers, and 0s are returned
1	*UV1*	R/W1C	0	ADC Sample Sequencer 1 (SS1) FIFO Underflow status 0: The FIFO has not overflowed. 1: SS1 FIFO has bit an overflow condition where the FIFO is full and a write was requested. The problematic read does not move the FIFO pointers, and 0s are returned
0	*UV0*	R/W1C	0	ADC Sample Sequencer 0 (SS0) FIFO Underflow status 0: The FIFO has not overflowed. 1: SS0 FIFO has bit an overflow condition where the FIFO is full and a write was requested. The problematic read does not move the FIFO pointers, and 0s are returned

4.8.4.10 ADC_SSPRI - ADC SAMPLE SEQUENCER PRIORITY

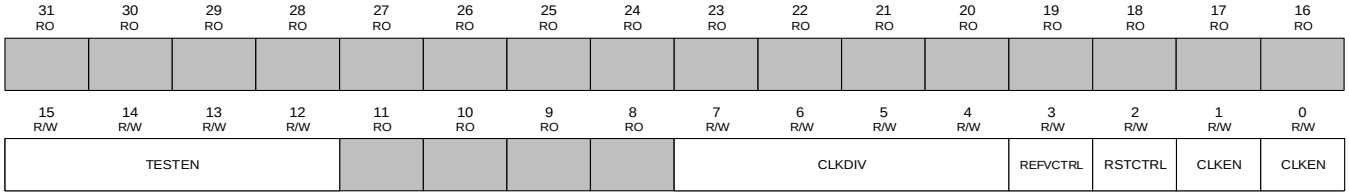
This register sets the priority for each of the sample sequencers. Out of reset, Sequencer 0 has the highest priority, and Sequencer 3 has the lowest priority. When reconfiguring sequence priorities, each sequence must have a unique priority for the ADC to operate properly.



Offset: 0x0028

Bits	Name	Type	Reset	Description
31:14	*reserved*	R0	0x0000_00	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
13:12	*SS3*	R/W	0x3	ADC Sample Sequencer 3 (SS3) Priority Same field description as in SS0 but only for SS3
11:10	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9:8	*SS2*	R/W	0x2	ADC Sample Sequencer 2 (SS2) Priority Same field description as in SS0 but only for SS2
7:6	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5:4	*SS1*	R/W	0x1	ADC Sample Sequencer 1 (SS1) Priority Same field description as in SS0 but only for SS1
3:2	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1:0	*SS0*	R/W	0x0	ADC Sample Sequencer 0 (SS0) Priority This field contains a binary-encoded value specifying the priority encoding of SS0. The priorities assigned to the sequencers must be uniquely mapped. 0x0: Highest Priority … 0x3: Lowest Priority

4.8.4.11 ADC_INI - ADC INITIAL CONTROL

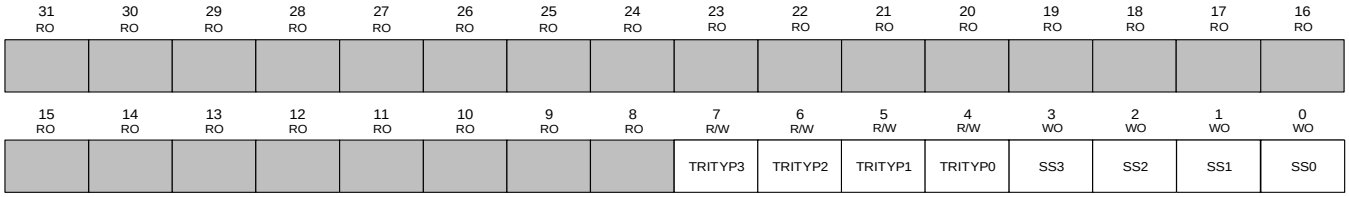


Offset: 0x002C

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:12	*TESTEN*	R/W	0x0	ADC macro Test mode Enable 0xF: Enable Macro Test Mode.
11:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:4	*CLKDIV*	R/W	0x0	ADC converter Clock Division 0 to 15 division ratio where 0 indicates no division is implemented.
3	*REFVCTRL*	R/W	0	Reference Voltage Control 0: VDD 3.3V is chosen as reference voltage. 1: VREF is chosen as reference voltage.
2	*RSTCTRL*	R/W	0	ADC FIFO Reset Control 0: FIFO will not be reset if it is not empty 1: FIFO will be reset whether it is empty while starting a new Sample Sequencer. The reset will only take action if one of the following conditions is met: When the FIFO is overflow or when the last Sample Sequencer is reached its endpoint.
1	*CLKEN*	R/W	0	ADC Clock Enable Note that to modify CLKDIV, user should first set this bit to 0 and re-enable this bit after modification on CLKDIV.
0	*ADINEN*	R/W	0	ADC Analog Input Enable 1: Enable analog input

4.8.4.12 ADC_PSSI - ADC PROCESSOR SAMPLE SEQUENCE INITIATE

This register provides a mechanism for application software to initiate sampling in the sample sequencers. Sample sequences can be initiated individually or in any combination. When multiple sequences are triggered simultaneously, the priority encodings in **ADC_SSPRI** dictate execution order.

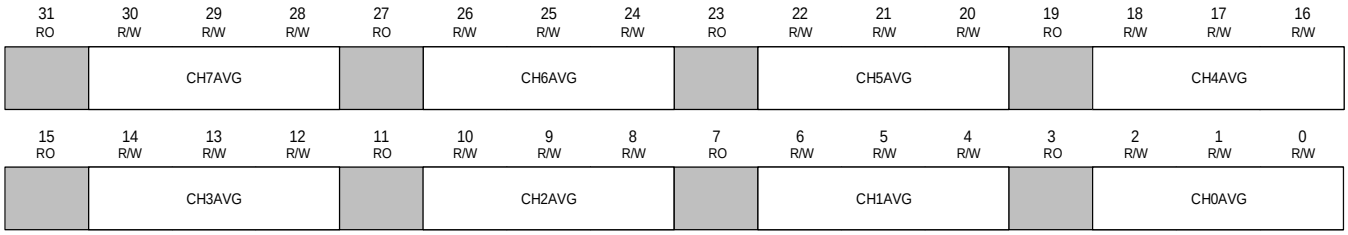


Offset: 0x0030

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	*TRITYP3*	R/W	0	ADC Sample Sequencer 3 (SS3) Trigger Type select Same description as in TRITYP0 but only for SS3
6	*TRITYP2*	R/W	0	ADC Sample Sequencer 2 (SS2) Trigger Type select Same description as in TRITYP0 but only for SS2
5	*TRITYP1*	R/W	0	ADC Sample Sequencer 1 (SS1) Trigger Type select Same description as in TRITYP0 but only for SS1
4	*TRITYP0*	R/W	0	ADC Sample Sequencer 0 (SS0) Trigger Type select 0: Edge Trigger 1: Level Trigger
3	*SS3I*	WO	-	ADC Sample Sequencer 3 (SS3) Initiate Same description as in SS0 but only for SS3
2	*SS2I*	WO	-	ADC Sample Sequencer 2 (SS2) Initiate Same description as in SS0 but only for SS2
1	*SS1I*	WO	-	ADC Sample Sequencer 1 (SS1) Initiate Same description as in SS0 but only for SS1
0	*SS0I*	WO	-	ADC Sample Sequencer 0 (SS0) Initiate 1: Triggers sampling on SS0 if the sequencer is enabled in the ADC_ACTSS.

4.8.4.13 ADC_AVGC - ADC AVERAGING CHANNEL CONTROL

This register controls the hardware sampling sequence average value, channel by channel.
If CHxAVG is 0, the sample is passed directly through without any averaging.
If CHxAVG=6, then 2^6 = 64 consecutive ADC samples are averaged to generate one result in the sequencer FIFO.
If CHxAVG = 7, it provides an unpredictable results.



Offset: 0x0034

Bits	Name	Type	Reset	Description
31	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
30:28	*CH7AVG*	R/W	0X0	Channel 7 Averaging Control Same description as in CH0AVG but only for Channel 7
27	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26:24	*CH6AVG*	R/W	0X0	Channel 6 Averaging Control Same description as in CH0AVG but only for Channel 6
23	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
22:20	*CH5AVG*	R/W	0X0	Channel 5 Averaging Control Same description as in CH0AVG but only for Channel 5
19	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
18:16	*CH4AVG*	R/W	0X0	Channel 4 Averaging Control Same description as in CH0AVG but only for Channel 4
15	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
14:12	*CH3AVG*	R/W	0X0	Channel 3 Averaging Control Same description as in CH0AVG but only for Channel 3
11	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10:8	*CH2AVG*	R/W	0X0	Channel 2 Averaging Control Same description as in CH0AVG but only for Channel 2
7	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:4	*CH1AVG*	R/W	0X0	Channel 1 Averaging Control Same description as in CH0AVG but only for Channel 1

Bits	Name	Type	Reset	Description
3	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*CH0AVG*	R/W	0X0	Channel 0 Averaging Control Specifies the amount of hardware averaging that will be applied to ADC samples. 0x0 No hardware oversampling 0x1 2x hardware oversampling 0x2 4x hardware oversampling 0x3 8x hardware oversampling 0x4 16x hardware oversampling 0x5 32x hardware oversampling 0x6 64x hardware oversampling 0x7 Reserved

4.8.4.14 ADC_PGAC - ADC PGA GAIN CONTROL

This register controls PGA (Programmable-gain Amplifier) gain value by writing the corresponding channel value. The ADC_PGA Control register is used to configure the gain of the on-chip non-inverting OP-AMP. Each ADC channel can be assigned with different gain values as shown in the PGA control register bit mapping. To change the PGA Control register the ADC sequencer must be stopped first by writing '0' to ADC_ACTSS register. Then update the ADC_PGA control register. Afterwards, enable the ADC sequencer by writing '1' to ADC_ACTSS.

The gain value varies under different PGA mode which is controlled by ADC_PGAC register.

- Non-inverting PGA mode:
- Differential PGA mode:

Figure 4.8-2: PGA Block Diagram

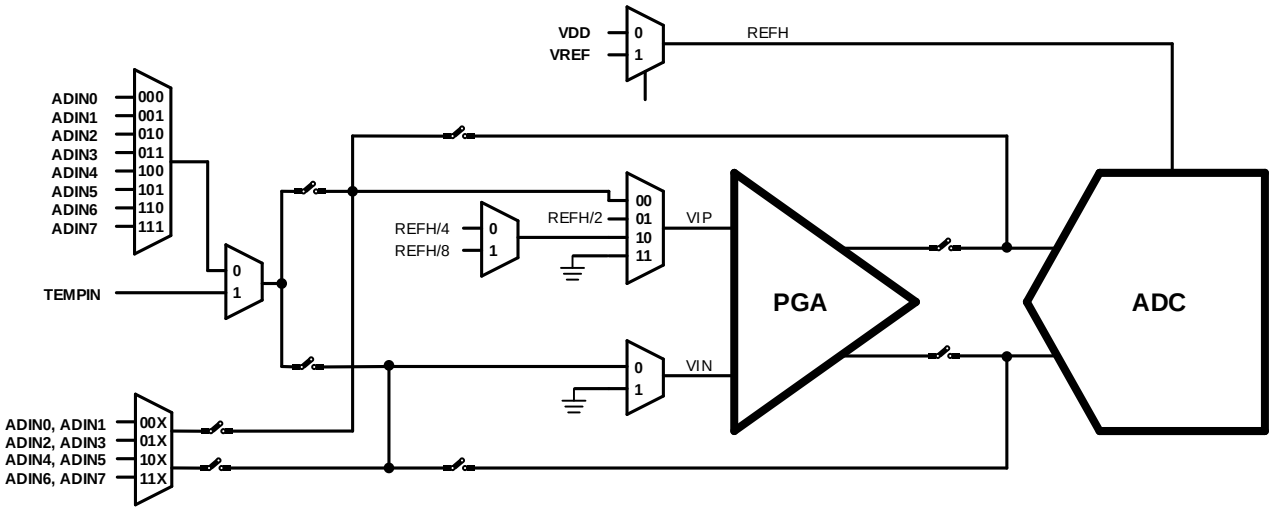
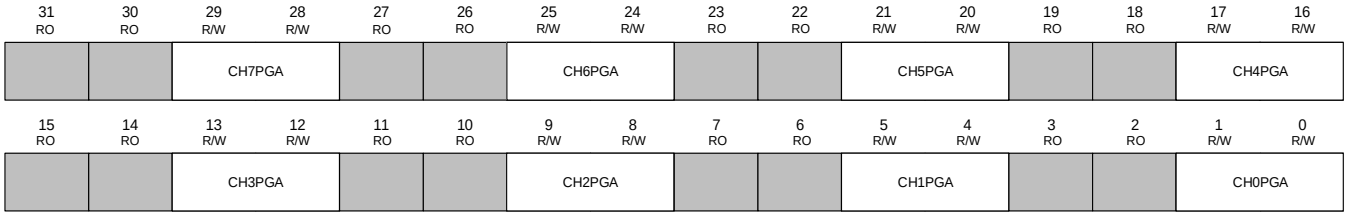


Table 4.8-2: Gain Amplification under different channel value

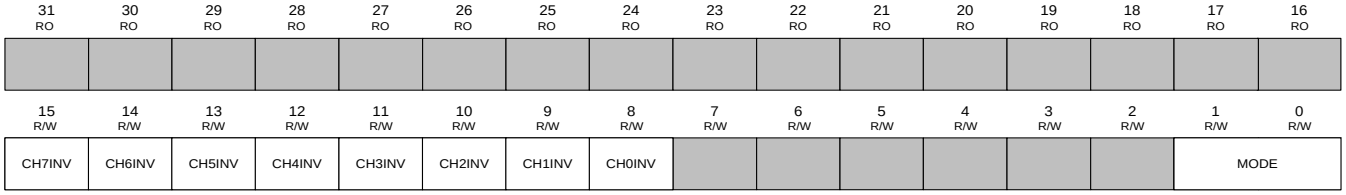
Channel Value (BIT[0],BIT[1])	Gain amplification
00	x1 (bypass)
01	x2
10	X4
11	X8



Offset: 0x0038

Bits	Name	Type	Reset	Description
31:30	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
29:28	*CH7PGA*	R/W	0x0	Channel 7 PGA control
27:26	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
25:24	*CH6PGA*	R/W	0x0	Channel 6 PGA control
23:22	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
21:20	*CH5PGA*	R/W	0x0	Channel 5 PGA control
19:18	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
17:16	*CH4PGA*	R/W	0x0	Channel 4 PGA control
15:14	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
13:12	*CH3PGA*	R/W	0x0	Channel 3 PGA control
11:10	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9:8	*CH2PGA*	R/W	0x0	Channel 2 PGA control
7:6	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5:4	*CH1PGA*	R/W	0x0	Channel 1 PGA control
3:2	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1:0	*CH0PGA*	R/W	0x0	Channel 0 PGA Control. When PGA Mode 2'b00/2'b11 00: X1 gain. 01: X2 gain. 10: X4 gain. 11: X8 gain. When PGA Mode 2'b01 00: X1 gain. 01: X1 gain. 10: X3 gain. 11: X7 gain.

4.8.4.15 ADC_PGAC - ADC PGA MODE CONTROL

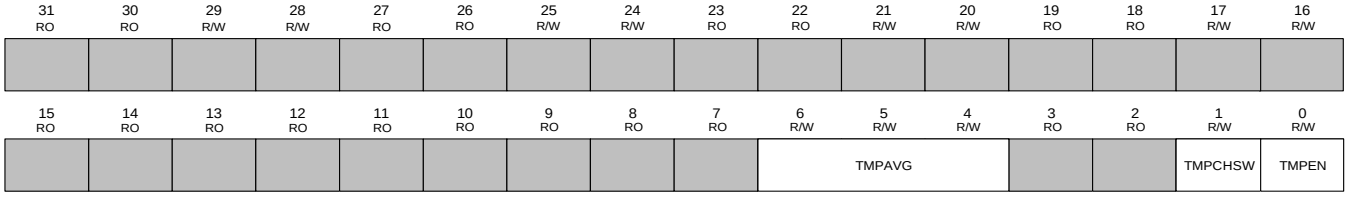


Offset: 0x003C

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15	*CH7INV*	R/W	0x0	Channel 7 Invert control Same description as in CH0INV but only for Channel 7.
14	*CH6INV*	R/W	0x0	Channel 6 Invert control Same description as in CH0INV but only for Channel 6.
13	*CH5INV*	R/W	0x0	Channel 5 Invert control Same description as in CH0INV but only for Channel 5.
12	*CH4INV*	R/W	0x0	Channel 4 Invert control Same description as in CH0INV but only for Channel 4.
11	*CH3INV*	R/W	0x0	Channel 3 Invert control Same description as in CH0INV but only for Channel 3.
10	*CH2INV*	R/W	0x0	Channel 2 Invert control Same description as in CH0INV but only for Channel 2.
9	*CH1INV*	R/W	0x0	Channel 1 Invert control Same description as in CH0INV but only for Channel 1.
8	*CH0INV*	R/W	0x0	Channel 0 Invert control 0: No inversion in channel 0 1: Channel 0 is inverted.
7:2	*reserved*	R/W	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1:0	*MODE*	R/W	0x0	PGA Mode Control 00: Non-Inverting PGA Mode 01: Inverting PGA Mode 10: ADC Differential Mode 11: Differential PGA Mode
3:2	*CH23DIFF*	R/W	0x0	Channel 3 and 2 Differential Control 0: No differential control between channel 3 and 2 1: Differential control is activated.
1:0	*CH01DIFF*	R/W	0x0	Channel 1 and 0 Differential Control 0: No differential control between channel 1 and 0 1: Differential control is activated.

4.8.4.16 ADC_TMC - ADC TEMPERATURE CONTROL

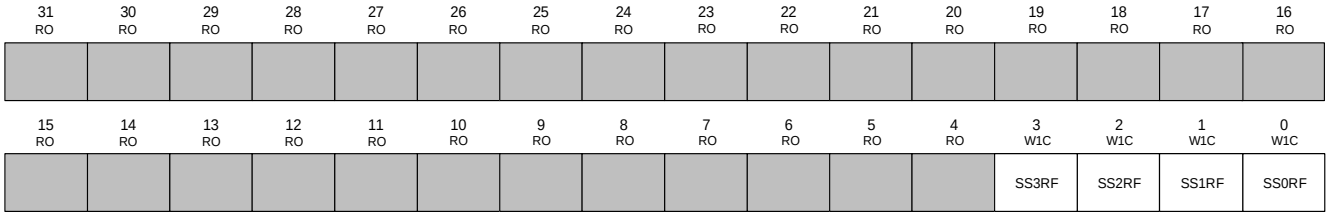
This register controls ADC temperature detection control.



Offset: 0x0040

Bits	Name	Type	Reset	Description
31:7	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:4	*TMPAVG*	R/W	0x0	Temperature Averaging Control
3:2	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*TMPCHSW*	R/W	0	Temperature Channel Switch 1: Switch all ADC channel to temperature input.
0	*TMPEN*	R/W	0	Temperature Module Enable 1: Enable temperature module.

4.8.4.17 ADC_FRF - ADC FIFO REFRESH.

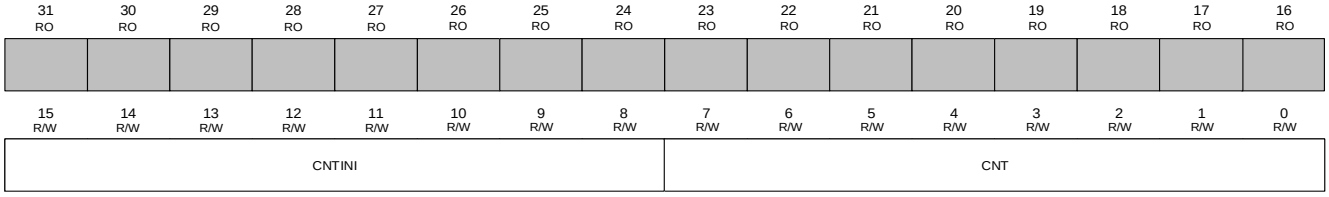


Offset: 0x0044

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*SS3RF*	W1C	0	ADC Sample Sequencer 3 (SS3) FIFO Refresh 1: SS3 FIFO reset
2	*SS2RF*	W1C	0	ADC Sample Sequencer 2 (SS2) FIFO Refresh 1: SS2 FIFO reset
1	*SS1RF*	W1C	0	ADC Sample Sequencer 1 (SS1) FIFO Refresh 1: SS1 FIFO reset
0	*SS0RF*	W1C	0	ADC Sample Sequencer 0 (SS0) FIFO Refresh 1: SS0 FIFO reset

4.8.4.18 ADC_WAIT - ADC WAIT COUNTER

This register controls ADC end of conversion to convert the start of the next waiting time.

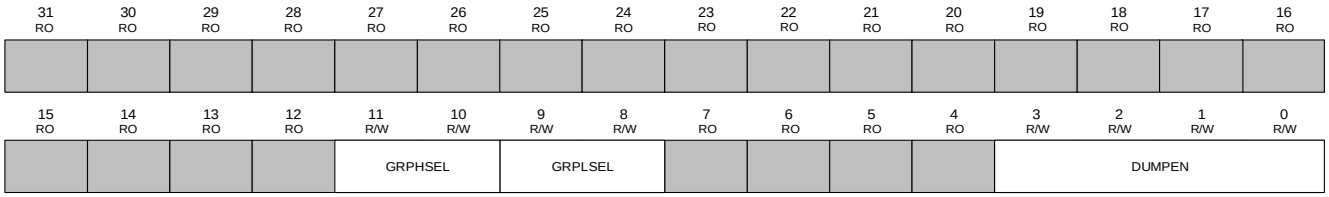


Offset: 0x0044

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:8	*CNTINI*	R/W	0	ADC Wait Counter Initial Value
7:0	*CNT*	R/W	0	ADC Wait Counter Value To set the wait cycle from adc_end to adc_start.

4.8.4.19 ADC_DMACTRL - ADC DMA CONTROL REGISTER

This register controls ADC's DMA request as it can send two request to the DMA controller. SS0 and SS1 are grouped as Group Low; SS2 and SS3 are grouped as Group High. If GRPHSEL or GRPLSEL is equal to 0x3, the DMA request will not be activated.

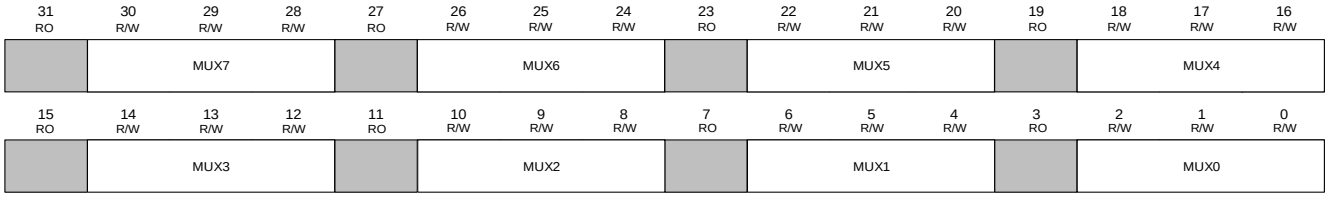


Offset: 0x004C

Bits	Name	Type	Reset	Description
31:12	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11:10	*GRPHSEL*	R/W	0	Group High Sample Sequencer Selection 0: SS2 is selected for DMA channel. 1: SS3 is selected for DMA channel.
9:8	*GRPLSEL*	R/W	0	Group Low Sample Sequencer Selection 0: SS0 is selected for DMA channel. 1: SS1 is selected for DMA channel.
7:4	*reserved*	RO	0x0000_000	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:0	*DUMPEN*	R/W	0	Dumping Enable 1: A single DMA request is asserted at the end of each A/D conversion.

4.8.4.20 ADC_SSMUX0 - ADC SAMPLE SEQUENCE INPUT MULTIPLEXER SELECT 0

This register defines the analog input configuration for each sample in a sequence executed with Sample Sequencer 0. This register is 32 bits wide and contains information for eight possible samples.



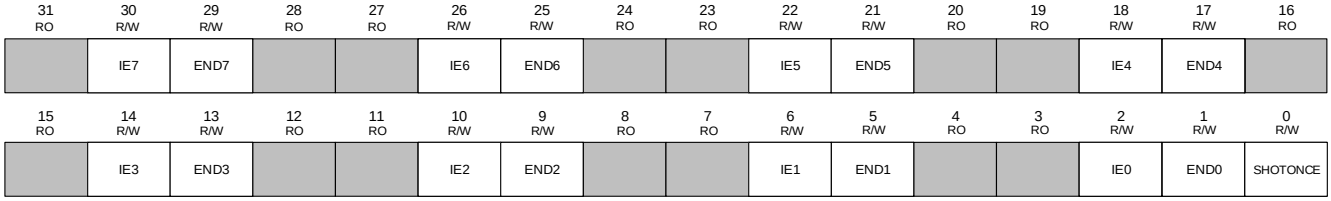
Offset: 0x0050

Bits	Name	Type	Reset	Description
31	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
30:28	*MUX7*	R/W	0x0	8TH Sample Input Select This field is used during the eighth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
27	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26:24	*MUX6*	R/W	0x0	7TH Sample Input Select This field is used during the eighth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
23	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
22:20	*MUX5*	R/W	0x0	6TH Sample Input Select This field is used during the eighth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
19	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
18:16	*MUX4*	R/W	0x0	5TH Sample Input Select This field is used during the eighth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
15	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
14:12	*MUX3*	R/W	0x0	4TH Sample Input Select This field is used during the eighth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
11	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

Bits	Name	Type	Reset	Description
10:8	*MUX2*	R/W	0x0	3TH Sample Input Select This field is used during the eighth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:4	*MUX1*	R/W	0x0	2TH Sample Input Select This field is used during the eighth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
3	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*MUX0*	R/W	0x0	1TH Sample Input Select This field is used during the eighth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
10	*IE2*	R/W	0	3rd Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the third sample's conversion.
9	*END2*	R/W	0	3rd Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The third sample is the last sample of the sequence. .
8:7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6	*IE1*	R/W	0	2nd Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the second sample's conversion.
5	*END1*	R/W	0	2nd Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The second sample is the last sample of the sequence. .
4:3	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*IE0*	R/W	0	1st Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the first sample's conversion.
1	*END0*	R/W	0	1st Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The first sample is the last sample of the sequence. .
0	*SHOTONCE*	R/W	0	One-shot Mode 1: Each trigger will only sample once and use one shot-mode in the FIFO.

4.8.4.21 ADC_SSCTL0 - ADC SAMPLE SEQUENCE CONTROL 0

This register contains the configuration information for each sample for a sequence executed with a sample sequencer. When configuring a sample sequence, the END bit must be set at some point, whether it be after the first sample, last sample, or any sample in between. This register is 32-bitswide and contains information for eight possible samples.



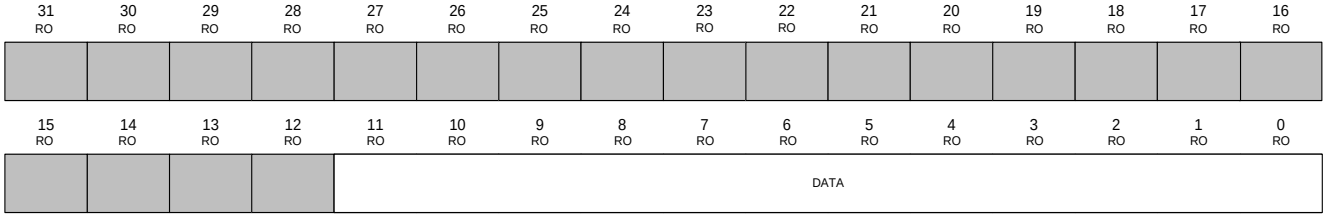
Offset: 0x0054

Bits	Name	Type	Reset	Description
31	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
30	*IE7*	R/W	0	8th Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the eighth sample's conversion.
29	*END7*	R/W	1	8th Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The eighth sample is the last sample of the sequence. .
28:27	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26	*IE6*	R/W	0	7th Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the seventh sample's conversion.
25	*END6*	R/W	0	7th Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The seventh sample is the last sample of the sequence. .
24:23	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
22	*IE5*	R/W	0	6th Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the sixth sample's conversion.
21	*END5*	R/W	0	6th Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The sixth sample is the last sample of the sequence. .
20:19	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
18	*IE4*	R/W	0	5th Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the fifth sample's conversion.
17	*END4*	R/W	0	5th Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The fifth sample is the last sample of the sequence. .

Bits	Name	Type	Reset	Description
16:15	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
14	*IE3*	R/W	0	4th Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the fourth sample's conversion.
13	*END3*	R/W	0	4th Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The fourth sample is the last sample of the sequence. .
12:11	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10	*IE2*	R/W	0	3rd Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the third sample's conversion.
9	*END2*	R/W	0	3rd Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The third sample is the last sample of the sequence. .
8:7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6	*IE1*	R/W	0	2nd Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the second sample's conversion.
5	*END1*	R/W	0	2nd Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The second sample is the last sample of the sequence. .
4:3	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*IE0*	R/W	0	1st Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the first sample's conversion.
1	*END0*	R/W	0	1st Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The first sample is the last sample of the sequence. .
0	*SHOTONCE*	R/W	0	One-shot Mode 1: Each trigger will only sample once and use one shot-mode in the FIFO.

4.8.4.22 ADC_SSFIFON - ADC SAMPLE SEQUENCE RESULT FIFO 0/1/2/3

This register contains the conversion results for samples collected with the sample sequencer (the **ADC_SSFIFO0** register is used for Sample Sequencer 0, **ADC_SSFIFO1** for Sequencer 1, **ADC_SSFIFO2** for Sequencer 2, and **ADC_SSFIFO3** for Sequencer 3). Reads of this register return conversion result data in the order sample 0, sample 1, and so on, until the FIFO is empty. If the FIFO is not properly handled by software, overflow and underflow conditions are registered in the **ADC_OSTAT** and **ADC_USTAT** registers.

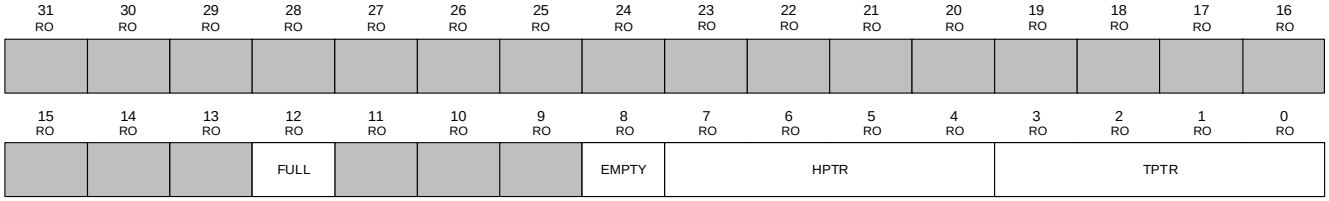


Offset:
ADC_SSFIFO0: 0x0058
ADC_SSFIFO1: 0x0078
ADC_SSFIFO2: 0x0098
ADC_SSFIFO3: 0x00B8

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11:0	DATA	RO	0x0	Conversion Result Data
11:9	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.8.4.23 ADC_SSFSTATN - ADC SAMPLE SEQUENCE FIFO 0/1/2/3 STATUS

This register provides a window into the sample sequencer, providing full/empty status information as well as the positions of the head and tail pointers. The reset value of 0x100 indicates an empty FIFO. The **ADC_SSFSTAT0** register provides status on FIFO0, **ADC_SSFSTAT1** on FIFO1, **ADC_SSFSTAT2** on FIFO2, and **ADC_SSFSTAT3** on FIFO3.



Offset:

ADC_SSFIFO0: 0x005C

ADC_SSFIFO1: 0x007C

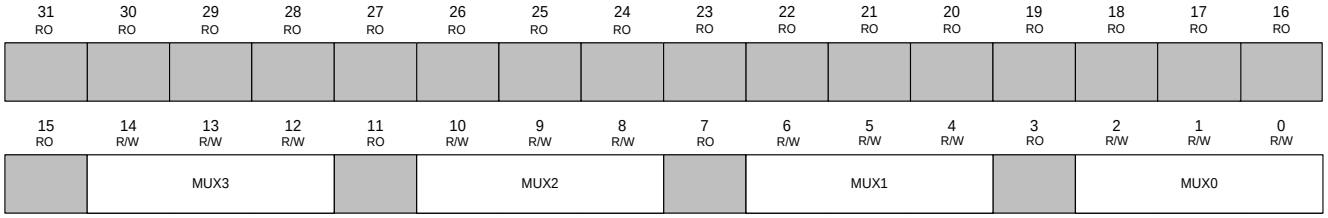
ADC_SSFIFO2: 0x009C

ADC_SSFIFO3: 0x00BC

Bits	Name	Type	Reset	Description
31:13	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
12	FULL	RO	0	FIFO Full 0: The FIFO is not currently full. 1: The FIFO is currently full.
11:9	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
8	EMPTY	RO	1	FIFO Empty 0: The FIFO is not currently empty. 1: The FIFO is currently empty.
7:4	HPTR	RO	0x0	FIFO Head Pointer The field contains the current “head” pointer index, which is the next entry to be written, for the FIFO. 0x0 - 0x7: FIFO 0 0x0 - 0x3: FIFO 1 and FIFO 2 0x0: FIFO 3
3:0	TPTR	RO	0x0	FIFO Tail Pointer The field contains the current “tail” pointer index, which is the next entry to be read, for the FIFO. 0x0 - 0x7: FIFO 0 0x0 - 0x3: FIFO 1 and FIFO 2 0x0: FIFO 3

4.8.4.24 ADC_SSMUX1 / ADC_SSMUX2 - ADC SAMPLE SEQUENCE INPUT MULTIPLEXER SELECT 1/2

This register defines the analog input configuration for each sample in a sequence executed with Sample Sequencer 1 or 2. These registers are 16-bits wide and contain information for 4 possible samples. The **ADC_SSMUX1**register affects Sample Sequencer 1 and the **ADC_SSMUX2** register affects Sample Sequencer 2.



Offset:

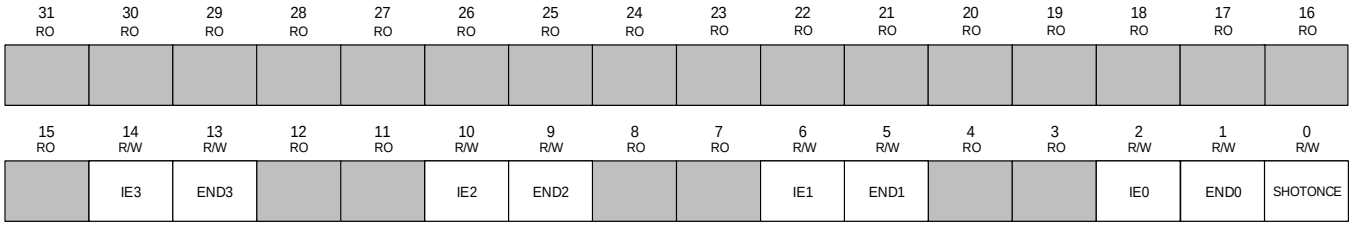
ADC_SSMUX1: 0x0070

ADC_SSMUX2: 0x0090

Bits	Name	Type	Reset	Description
31:15	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
14:12	*MUX3*	R/W	0x0	4th Sample Input Select This field is used during the fourth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
11	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10:8	*MUX2*	R/W	0x0	3rd Sample Input Select This field is used during the third sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:4	*MUX1*	R/W	0x0	2nd Sample Input Select This field is used during the second sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
3	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*MUX0*	R/W	0x0	1st Sample Input Select This field is used during the first sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.

4.8.4.25 ADC_SSCTL1 /ADC_SSCTL2 - ADC SAMPLE SEQUENCE CONTROL 1/2

These registers contain the configuration information for each sample for a sequence executed with Sample Sequencer 1 or 2. When configuring a sample sequence, the *END* bit must be set at some points, whether it is after the first sample, last sample, or any sample in between. These registers are 16-bits wide and contain information for four possible samples. The **ADC_SSCTL1** register configures Sample Sequencer1 and the **ADC_SSCTL2** register configures Sample Sequencer 2.



Offset:

ADC_SSCTL1: 0x0074

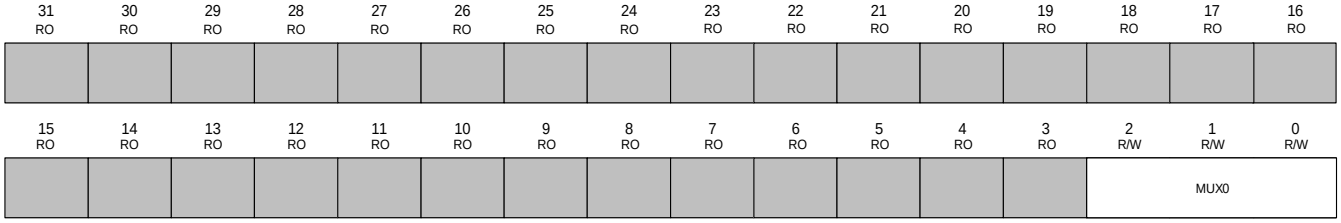
ADC_SSCTL2: 0x0094

Bits	Name	Type	Reset	Description
31:15	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
14	*IE3*	R/W	0	4th Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the fourth sample's conversion.
13	*END3*	R/W	1	4th Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The fourth sample is the last sample of the sequence. .
12:11	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10	*IE2*	R/W	0	3rd Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the third sample's conversion.
9	*END2*	R/W	0	3rd Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The third sample is the last sample of the sequence. .
8:7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6	*IE1*	R/W	0	2nd Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the second sample's conversion.
5	*END1*	R/W	0	2nd Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The second sample is the last sample of the sequence. .
4:3	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

Bits	Name	Type	Reset	Description
2	*IE0*	R/W	0	1st Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the first sample's conversion.
1	*END0*	R/W	0	1st Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The first sample is the last sample of the sequence. .
0	*SHOTONCE*	R/W	0	One-shot Mode 1: Each trigger will only sample once and use one shot-mode in the FIFO.

4.8.4.26 ADC_SSMUX3 - ADC SAMPLE SEQUENCE INPUT MULTIPLEXER SELECT 3

This register defines the analog input configuration for a sample executed with Sample Sequencer3. This register is 3-bits wide and contains information for one possible sample. See the **ADC_SSMUX0** register for detailed bit descriptions.

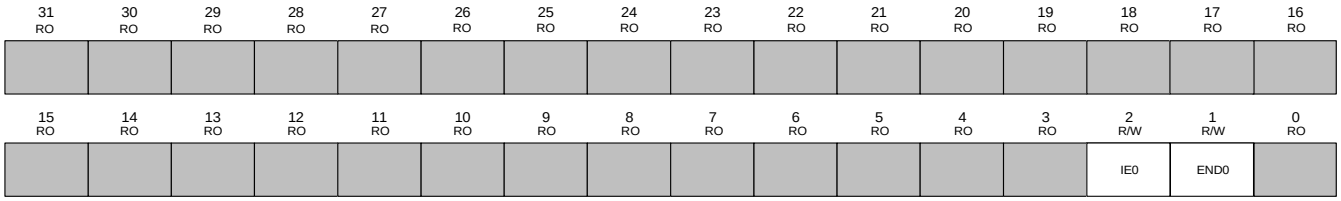


Offset: 0x00B0

Bits	Name	Type	Reset	Description
31:3	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*MUX0*	R/W	0x0	1st Sample Input Select This field is used during the first sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.

4.8.4.27 ADC_SSCTL3 - ADC SAMPLE SEQUENCE CONTROL 3

This register configures the sampling sequence 3 sampling interrupt the boot sequence and the end point, because the sampling sequence 3 is only a sampling sequence, so the end point END0 default setting is 1.



Offset: 0x00B4

Bits	Name	Type	Reset	Description
31:3	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*IE0*	R/W	0	1st Sample Interrupt Enable 0: The raw interrupt is not asserted to the interrupt controller. 1: The raw interrupt signal is asserted at the end of the first sample's conversion.
1	*END0*	R/W	1	1st Sample is End of Sequence 0: Another sample in the sequence is the final sample. 1: The first sample is the last sample of the sequence. .
0	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.9 DIGITAL TO ANALOG CONVERTER (DAC)

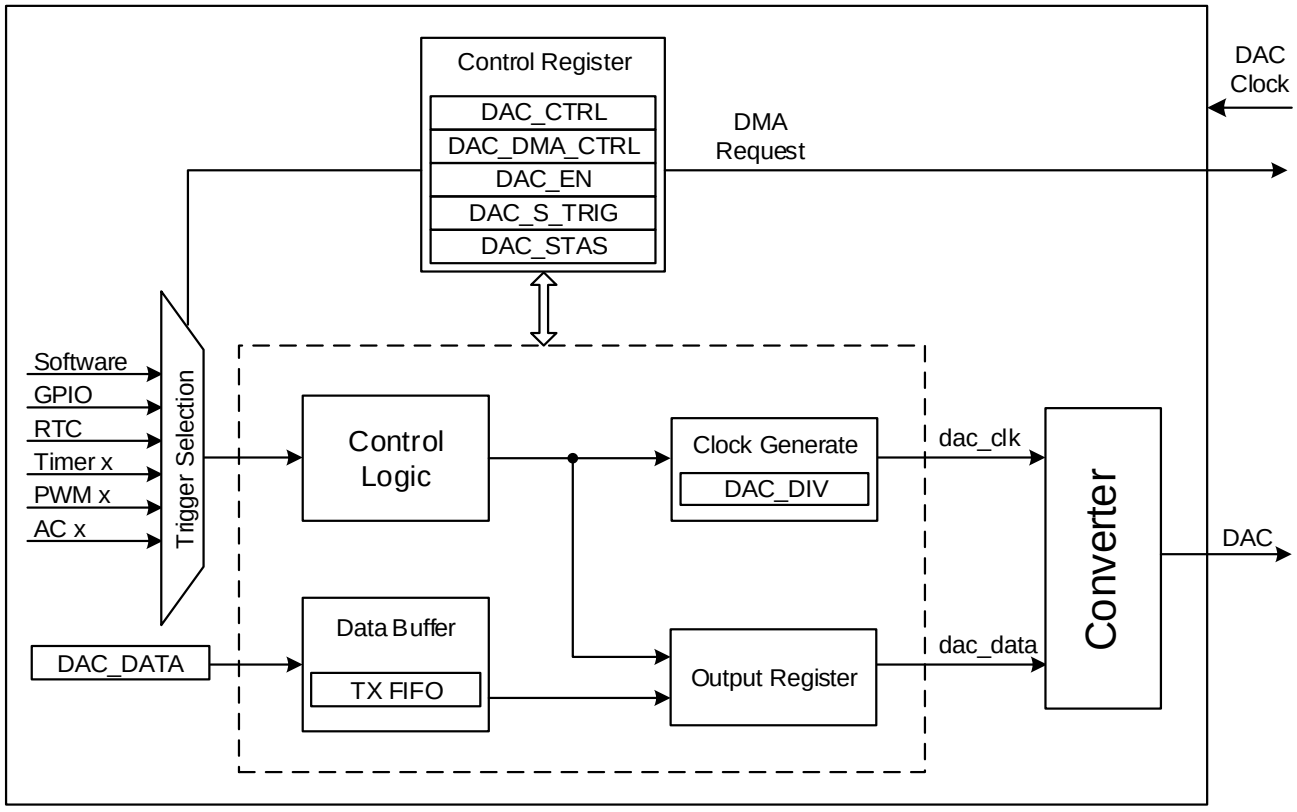
The DAC module is a 12-bit, voltage output digital-to-analog converter. The DAC may be used in conjunction with the DMA controller. DAC has one output channel with its own converter.

The DAC module provides the following features:

- 12-bits DAC converter
- Support External, RTC, Timer, PWM, Comparator and Software Triggers for conversion.
- Support up to 200 KHz conversion.
- DMA capability
- Provide Saw Tooth wave and Triangular wave generation.
- Provide data output invert function

4.9.1 BLOCK DIAGRAM

Figure 4.9-1: GPTM Module Block Diagram



4.9.2 FUNCTIONAL DESCRIPTION

OUTPUT VOLTAGE

Digital inputs are converted to output voltages on a linear conversion between 0.55 V and 2.75 V.The analog output voltages on each DAC channel pin are determined by the following equation:

DAC_{out} = [(2.2 * DATA) / 4095] + 0.55 (V)

TRIGGER SELECTION

If the **DAC_CTRL**.*TRIEN* control bit is set, conversion can then be triggered by an external event (Timer counter, PWM, RTC and Comparator). The **DAC_CTRL**.*TRISEL [3:0]* control bits determine which out of 15 possible events will trigger conversion.

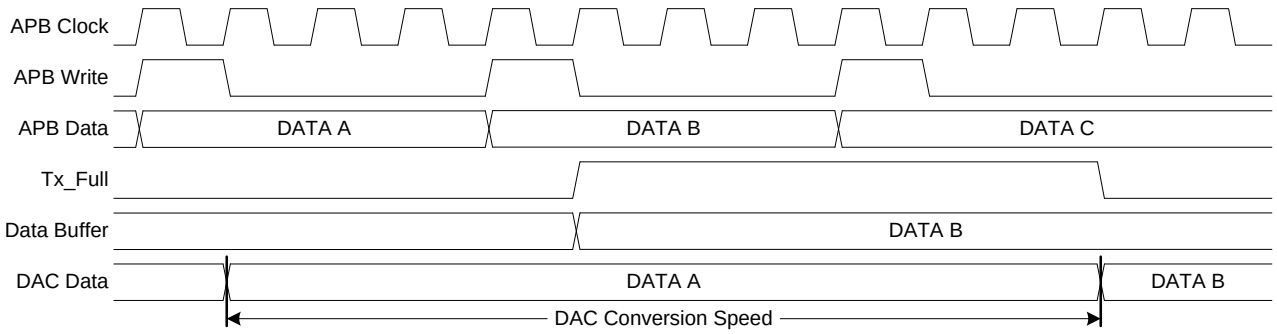
Each time a DAC interface detects a rising edge on the selected trigger, the data stored into the **DAC_DATA** register are transferred. The interface dac_data is updated one APB cycles after the trigger occurs.

Note: **DAC_CTRL** Register cannot be changed when the **DAC_EN** register is set.

DATA TRANSMISSION

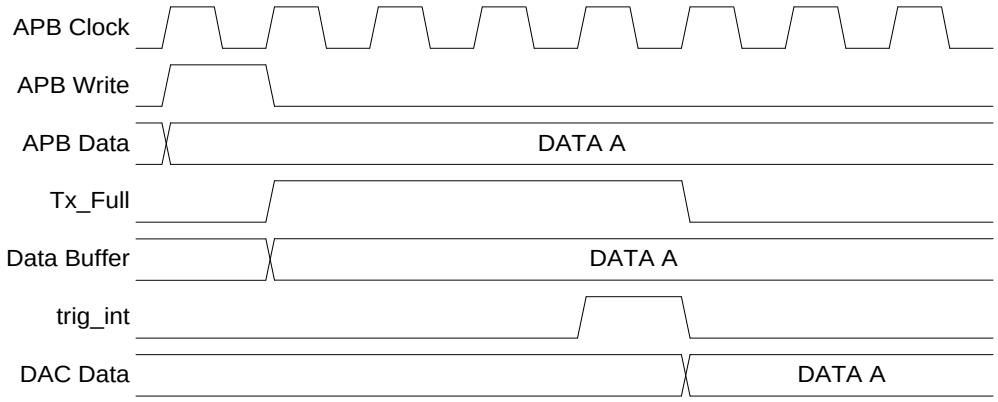
The DAC integrates a data buffer in data transmission. When **DAC_CTRL**.*TRIEN* is not set, the data stored into the **DAC_DATA** register are automatically transferred to data output logic after 1 pclk cycle. The register **DAC_DATA** can also be written at this moment, and the new data will be stored in data buffer and be transferred to data output logic after the pervious data has been completely transferred.

Figure 4.9-2: Timing diagram for Data Transmission, when DAC_CTRL.*TRIEN* = 0



If **DAC_CTRL**.*TRIEN* bit is set, when a data is written to **DAC_DATA** register, the data will saved in the data buffer and waiting for the trigger to occurs, the data transmission will start after 1 pclk cycle. At this time, any write operation to **DAC_DATA** register is useless.

Figure 4.9-3: Timing diagram for Data Transmission, when DAC_CTRL.*TRIEN* = 1



WAVE GENERATION

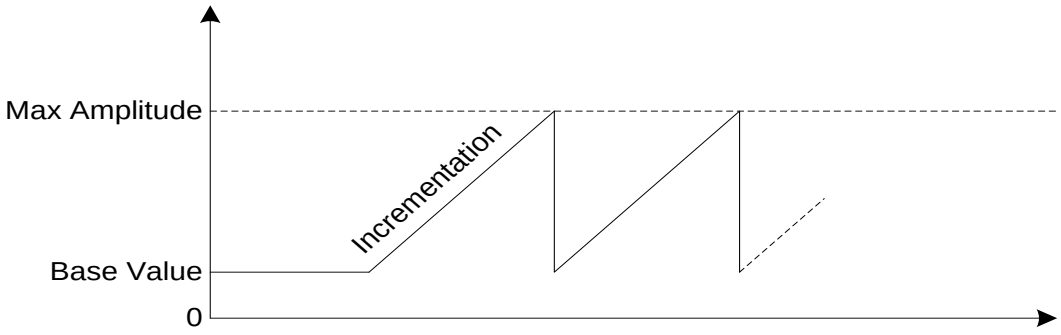
DAC, configured by **DAC_CTRL**.*WAVE [1:0]*, can generate two types of waveforms.

- Saw Tooth Wave
- Triangle Wave

It is possible to add a small-amplitude triangular waveform on a DC or slowly varying signal. The amplitude is configured through the *MAMP [3:0]* bits in the **DAC_CTRL** register. An internal counter is incremented, decremented or first incremented then decremented depends on the setting of **DAC_CTRL**.*WAVE [1:0]* bits. The value of this counter is then added to the dac_data without overflow and the sum is stored into the data buffer.

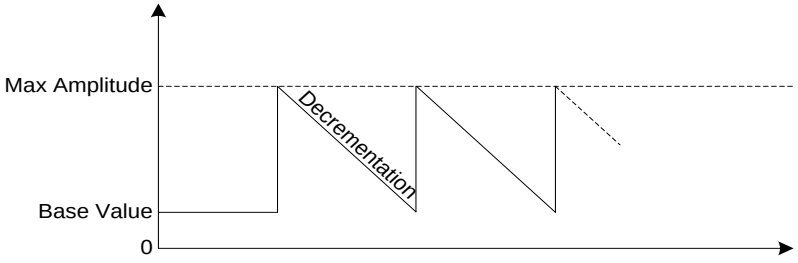
In the up-counting mode, the counter is incremented as long as it is less than the maximum amplitude defined by the **DAC_CTRL**.*MAMP [3:0]* bits. Once the configured amplitude is reached, the counter goes back to 0, then incremented again and so on.

Figure 4.9-4: DAC saw tooth wave generation (up-counting)



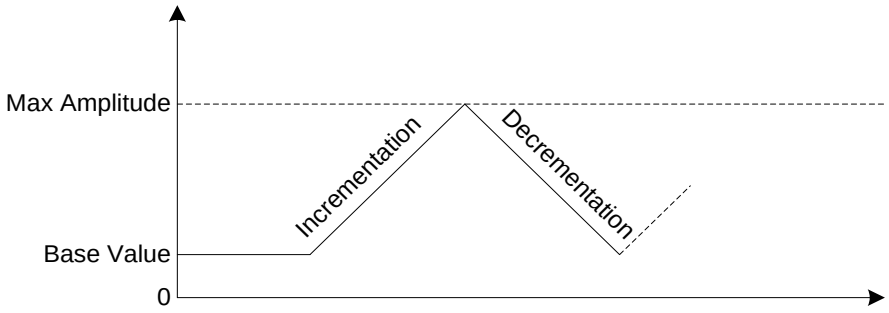
In the down-counting mode, the counter is decremented down to zero. Once the zero is reached, the counter is reloaded to the maximum amplitude defined by the **DAC_CTRL**.*MAMP [3:0]* bits, then decremented again and so on.

Figure 4.9-5: DAC saw tooth wave generation (down-counting)



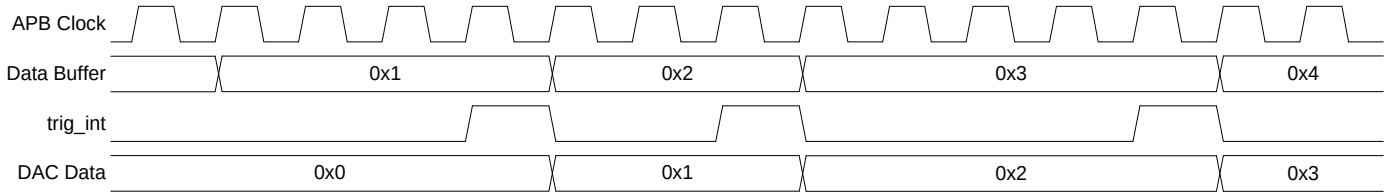
In the triangle mode, the counter is incremented as long as it is less than the maximum amplitude defined by the **DAC_CTRL**.*MAMP [3:0]*. Once the configured amplitude is reached, the counter is decremented down to 0, then incremented again and so on.

Figure 4.9-6: DAC saw tooth wave generation (up/down-counting)



It is possible to reset the wave generation by clear the **DAC_EN** register.

Figure 4.9-7: DAC conversion (trigger enabled) with wave generation



DATA OUTPUT INVERT

Enable **DAC_CTRL.***ODINV* bit, then the output data will be inverted. For example, write 0xA, 0xB, and 0xFF3 to **DAC_DATA**, then the output data will be 0xFF5, 0xFF4, and 0xC.

*Note**: When in the wave generation mode, this bit has no effect.***

DMA CONTROL

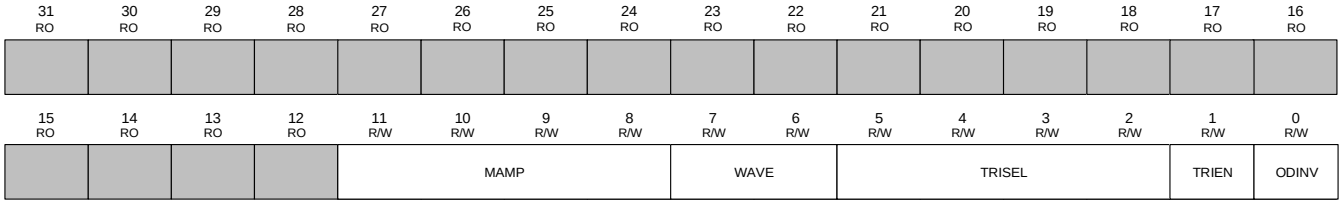
To use DMA to write date to DAC, setting the *DMATXEN* bit in **DAC_DMA_CTRL** register. When data buffer is empty, DAC will send the request to the DMA and wait for DMA to write single data.

4.9.3 DAC REGISTER MAP

Base Address: 0x4802_8000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	DAC_CTRL	R/W	0x0000_0000	DAC Control Register	168
0x0004	DAC_DIV	R/W	0x0000_0000	DAC Divider Register	169
0x0008	DAC_STAS	RO	0x0000_0000	DAC Status Register	169
0x000C	DAC_EN	R/W	0x0000_0000	DAC Enable Register	170
0x0010	DAC_DMA_CTRL	R/W	0x0000_0000	DAC DMA Control Register	170
0x0014	DAC_S_TRIG	R/W1C	0x0000_0000	DAC Software Trigger Register	171
0x0018	DAC_DATA	R/W	0x0000_0000	DAC Data Register	171

4.9.3.1 DAC_CTRL - DAC CONTROL REGISTER

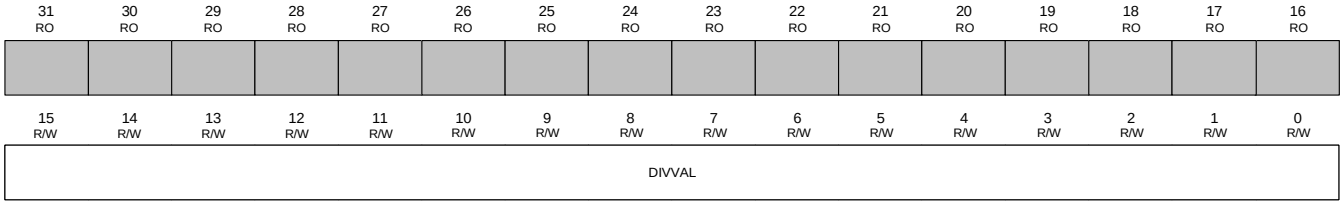


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:12	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11:8	*MAMP*	R/W	0x0	DAC Mask Amplitude Select 0000: Unmask bit0 / amplitude equal to 1 0001: Unmask bits[1:0] / amplitude equal to 3 0010: Unmask bits[2:0] / amplitude equal to 7 0011: Unmask bits[3:0] / amplitude equal to 15 0100: Unmask bits[4:0] / amplitude equal to 31 0101: Unmask bits[5:0] / amplitude equal to 63 0110: Unmask bits[6:0] / amplitude equal to 127 0111: Unmask bits[7:0] / amplitude equal to 255 1000: Unmask bits[8:0] / amplitude equal to 511 1001: Unmask bits[9:0] / amplitude equal to 1023 1010: Unmask bits[10:0] / amplitude equal to 2047 ≥1011: Unmask bits[11:0] / amplitude equal to 4095
7:6	*WAVE*	R/W	0x0	DAC Wave Generate Select 00: Wave generation disable 01: Saw tooth wave (Up-counting) 10: Saw tooth wave (Down-counting) 11: Triangle Wave
5:2	*TRISEL*	R/W	0x0	DAC Trigger Source Select 0x0: Software Trigger 0x1: GPIO 0x2: Reserved 0x3: RTC 0x4: Reserved 0x5: Reserved 0x6: Reserved 0x7: Reserved 0x8: Reserved 0x9: Reserved 0xA: Reserved 0xB: Reserved 0xC: Analog Comparator 0 0xD: Analog Comparator 1 0xE: Analog Comparator 2 0xF: Analog Comparator 3
1	*TRIEN*	R/W	0	DAC Trigger Enable 0: DAC channel trigger disabled and data written into DAC_DATA register is transferred one pclk cycle later to the data output logic 1: Enable DAC trigger and data transfer from DAC_DATA one cycle later to the data output logic.

Bits	Name	Type	Reset	Description
0	*ODINV*	R/W	0	DAC Output Data Invert 0: Data leave the DAC without modified 1: Invert DAC output data.

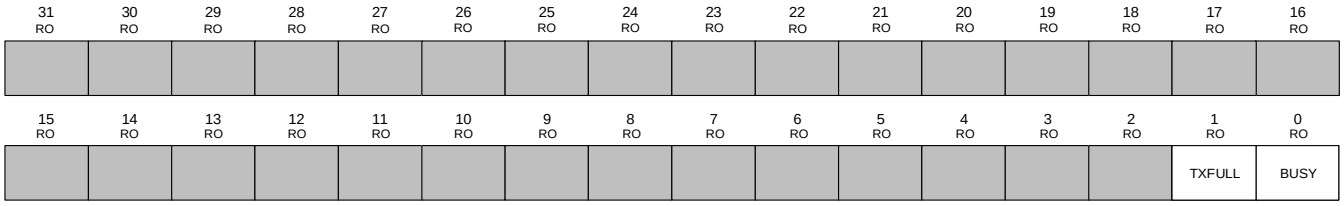
4.9.3.2 DAC_DIV - DAC DIVIDER REGISTER



Offset: 0x0004

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*DIVVAL*	R/W	0	DAC Divider Value, the value equal to 0 is not allowed.

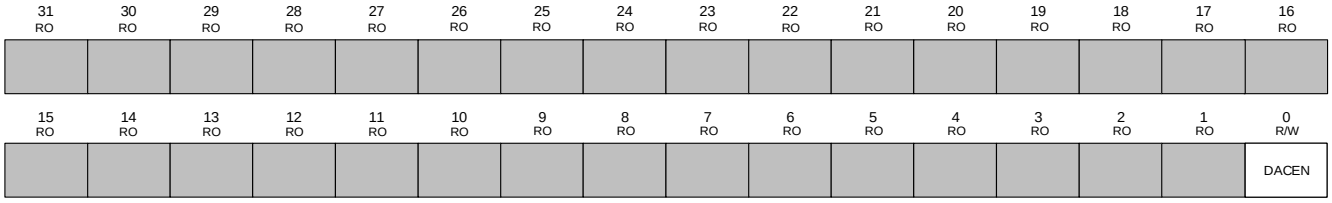
4.9.3.3 DAC_STAS - DAC STATUS REGISTER



Offset: 0x0004

Bits	Name	Type	Reset	Description
31:2	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*TXFULL*	RO	0	DAC Data Buffer (TX FIFO) is Full 0: DAC Tx FIFO contains at least one data 1: DAC Tx FIFO is full
0	*BUSY*	RO	0	DAC Busy Flag 0: DAC is idle 1: DAC is busy

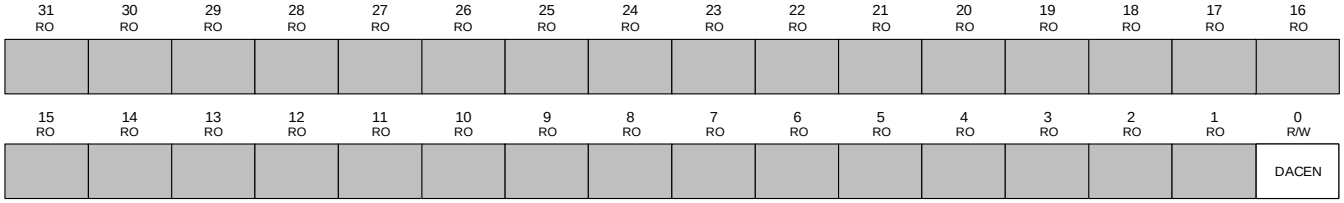
4.9.3.4 DAC_EN - DAC ENABLE REGISTER



Offset: 0x000C

Bits	Name	Type	Reset	Description
31:1	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*DACEN*	R/W	0	DAC Enable. 0: Disable 1: Enable

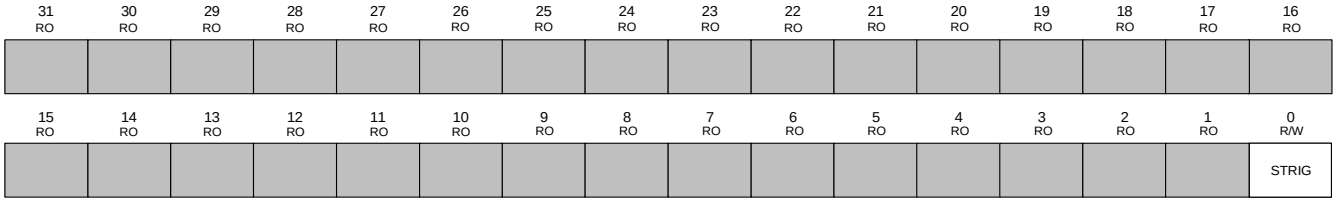
4.9.3.5 DAC_DMA_CTLR - DAC DMA CONTROL REGISTER



Offset: 0x0010

Bits	Name	Type	Reset	Description
31:1	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*DMATXEN*	R/W	0	DAC DMA Tx Enable. 0: Disable 1: Enable

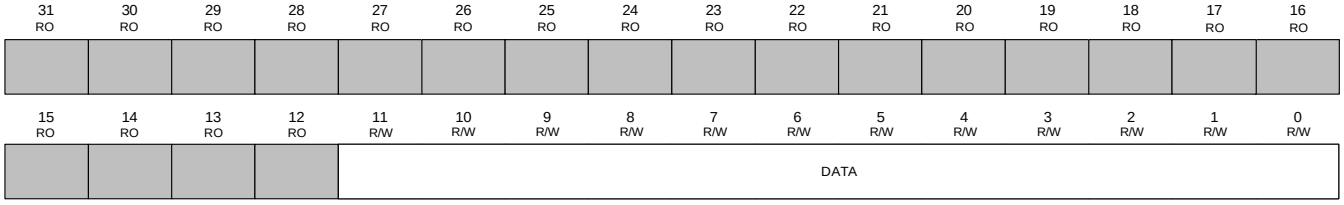
4.9.3.6 DAC_S_TRIG - DAC SOFTWARE TRIGGER REGISTER



Offset: 0x0014

Bits	Name	Type	Reset	Description
31:1	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*STRIG*	R/W1C	0	DAC Software Trigger

4.9.3.7 DAC_DATA - DAC DATA REGISTER



Offset: 0x0018

Bits	Name	Type	Reset	Description
31:12	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11:0	*DATA*	R/W	0x0	DAC Data Register

4.10 GENERAL PURPOSE TIMERS (GPT)

The general-purpose timers consist of a 16-bit auto-reload counter driven by a programmable prescaler.

They may be used for a variety of purposes, including measuring the pulse lengths 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 APB clock controller prescalers.

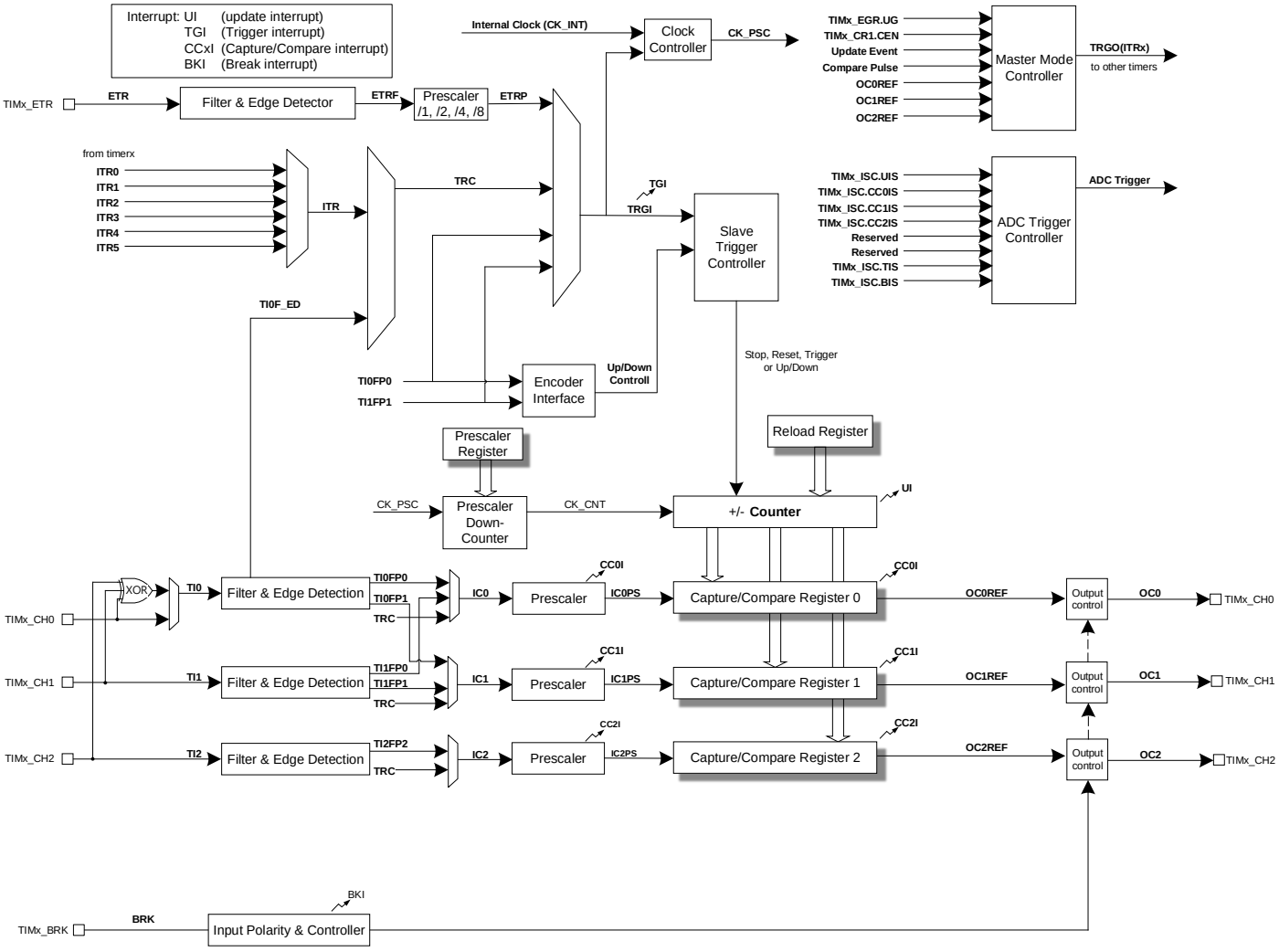
The advanced-control (TIM4, TIM5) and general-purpose (TIM0, TIM1, TIM2, TIM3) timers are completely independent, and do not share any resources.

General-purpose TIMx timer features include:

- 16-bit up, down, up/down auto-reload counter.
- 16-bit programmable prescaler used to divide (also “on the fly”) the counter clock frequency by any factor between 1 and 65535.
- Up to 3 independent channels for:
  - Input Capture
  - Output Compare
  - PWM generation (Edge- and Center-aligned Modes)
  - One-Shot mode output
- Synchronization circuit to control the timer with external signals and to interconnect several timers.
- Interrupt generation with 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 (capture register latch counter register)
  - Output compare (counter register match compare register)
  - Break event
  - Overcapture event
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management

4.10.1 BLOCK DIAGRAM

Figure 4.10-1: General-purpose timer block diagram



4.10.2 FUNCTIONAL DESCRIPTION

4.10.2.1 TIMER INITIALIZATION

The following bits are enable-protected, meaning that it can only be written when the Timer is disabled (**TIMx_CR1**.*CEN* is zero);

- Main output enable bit in the Break register (**TIMx_BDTR.***MOE*). For timers , the MOE bit controls all outputs and acts a circuit breaker in case of break detection on the break input (global disable of all PWM outputs)
- Capture/Compare x output enable bits in the Capture/Compare enable register (**TIMx_CCER**.*CCnE*)

4.10.2.2 TIMER-BASE UNIT

The main block of the programmable timer is a 16-bit counter with its related auto-reload register. The counter can count up but also 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 timer-base unit includes:

- Counter register (**TIMx_CNTR**)
- 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. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the *UDIS* bit equals 0 in the **TIMx_CR1** register. It can also be generated by software. The generation of the update event is described in detail for each configuration.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (*CEN*) in **TIMx_CR1** register is set (refer also to the slave mode controller description to get more details on counter enabling).

Note that the actual counter enable signal CNT_EN is set 1 clock cycle after *CEN* bit in the **TIMx_CR1** register.

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. The prescaler counter is down-counting only. When the prescaler is set, the value of the counter will counting up or down as soon as the prescaler counter reaches 0, otherwise, the counter updates along with the APB clock. **Figure 4.10-2** and **Figure 4.10-3** give some examples of the counter behavior when the prescaler ratio is changed on the fly.

Figure 4.10-2: Counter timing diagram with prescaler division change from 1 to 2

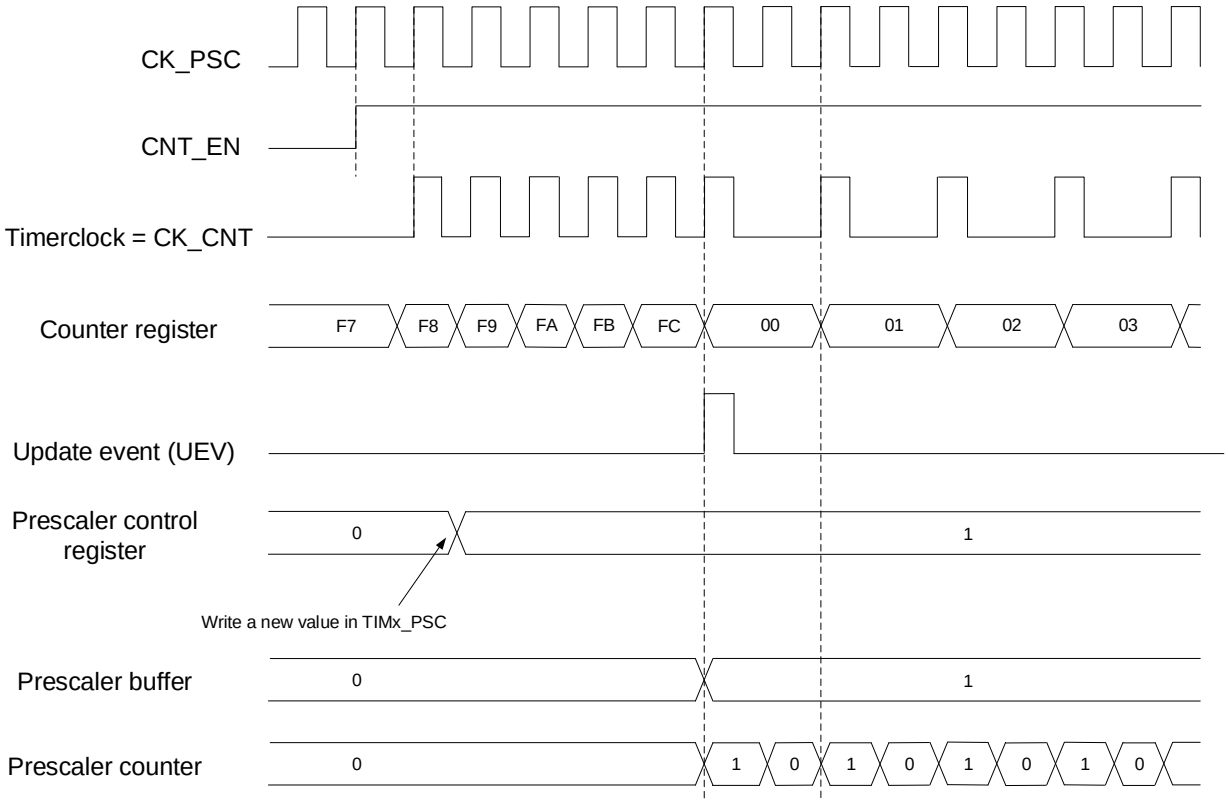
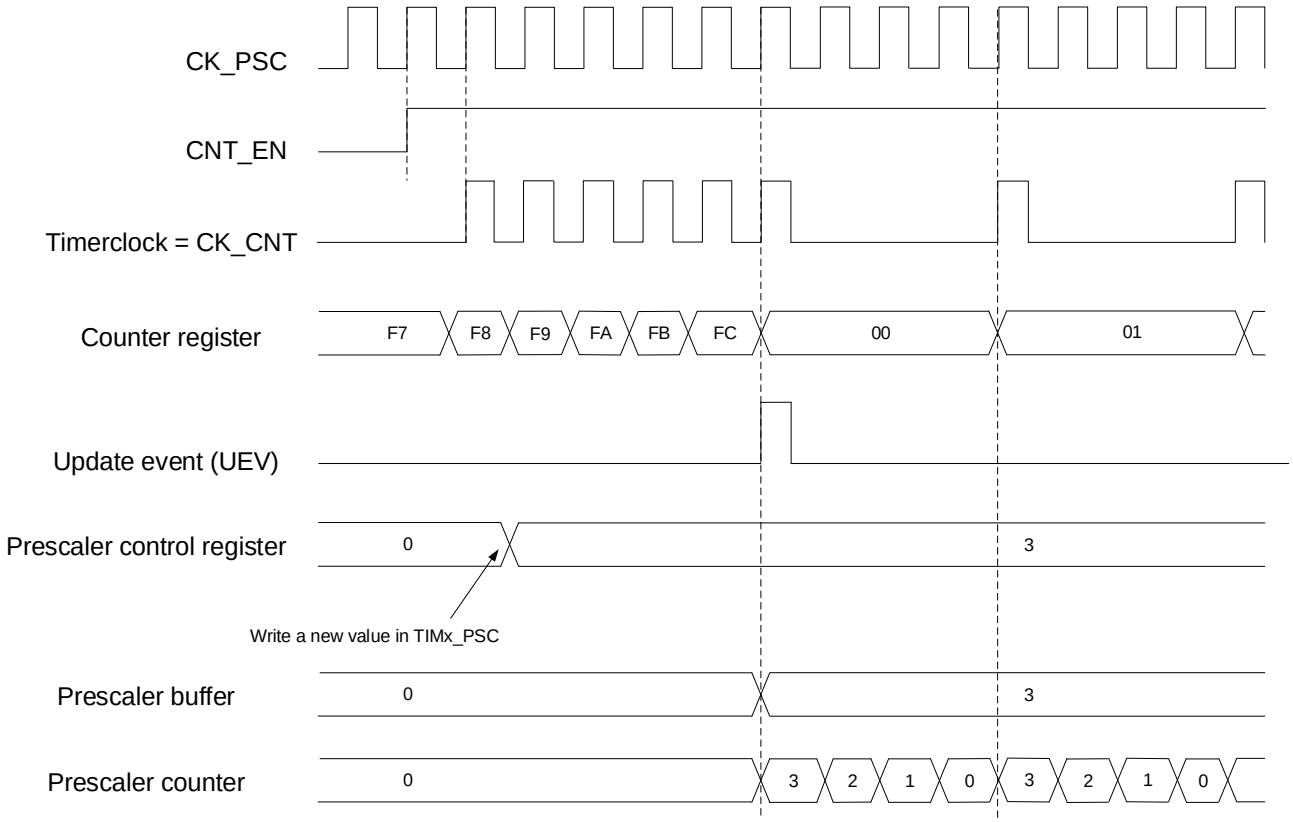


Figure 4.10-3: Counter timing diagram with prescaler division change from 1 to 4



4.10.2.3 COUNTER MODES

The main block of the programmable timer is a 16-bit counter with its related auto-reload register. The counter can count

UPCOUNTING MODE

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the **TIMx_ARR** register), then restarts from 0 and generates a counter overflow event.

An Update event can be generated at each counter overflow or by setting the UG bit in the **TIMx_EGR** register (by software or by using the slave mode controller).

The UEV event can be disabled by software by setting the *UDIS* bit in **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 the *UDIS* bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change).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 registers are updated and the update status (*UIS* bit in **TIMx_ISC** register) is set:

- The auto-reload 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 4.10-4: Counter timing diagram, internal clock divided by 1

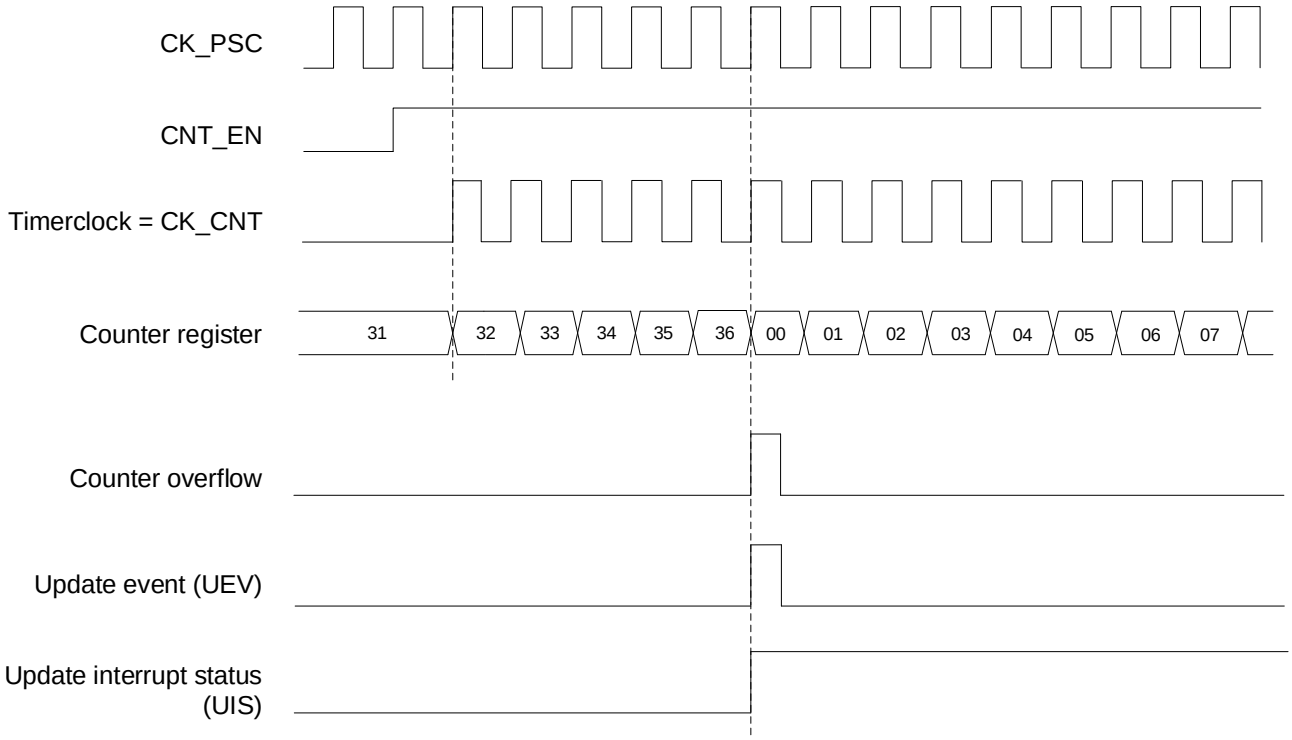


Figure 4.10-5: Counter timing diagram, internal clock divided by 2

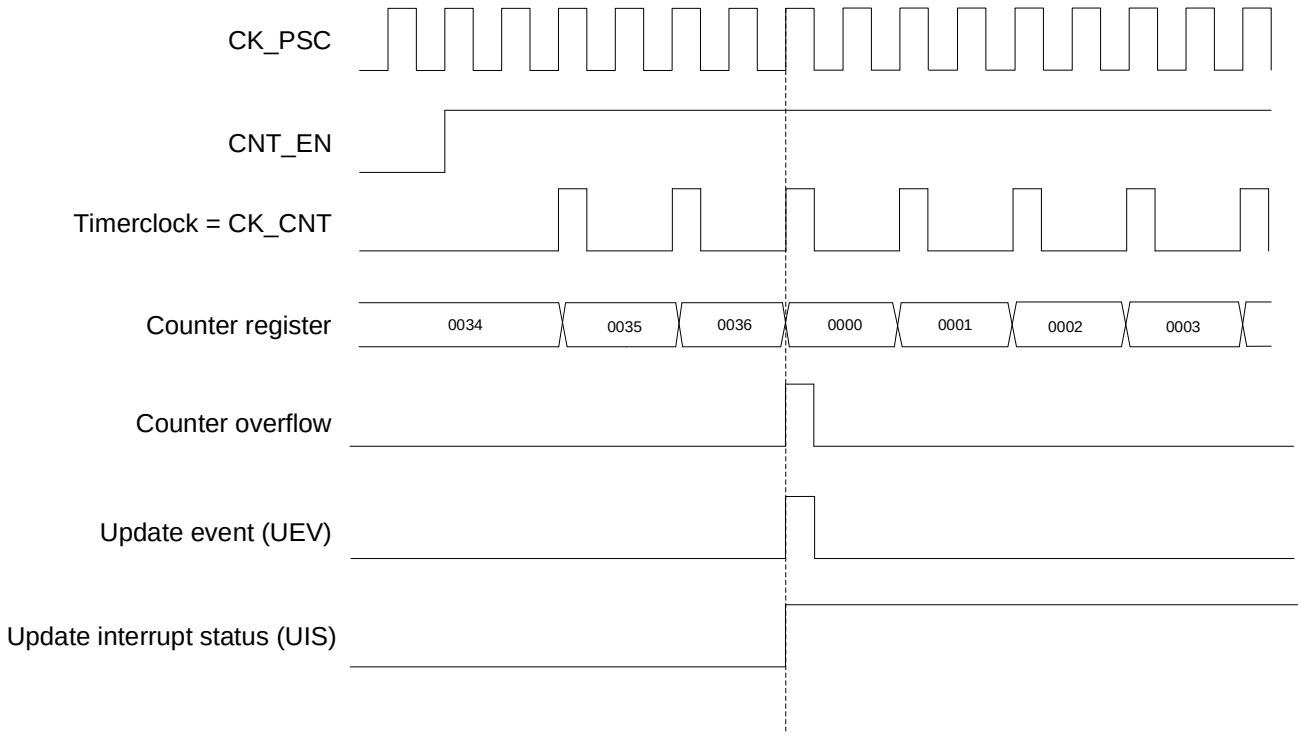


Figure 4.10-6: Counter timing diagram, internal clock divided by 4

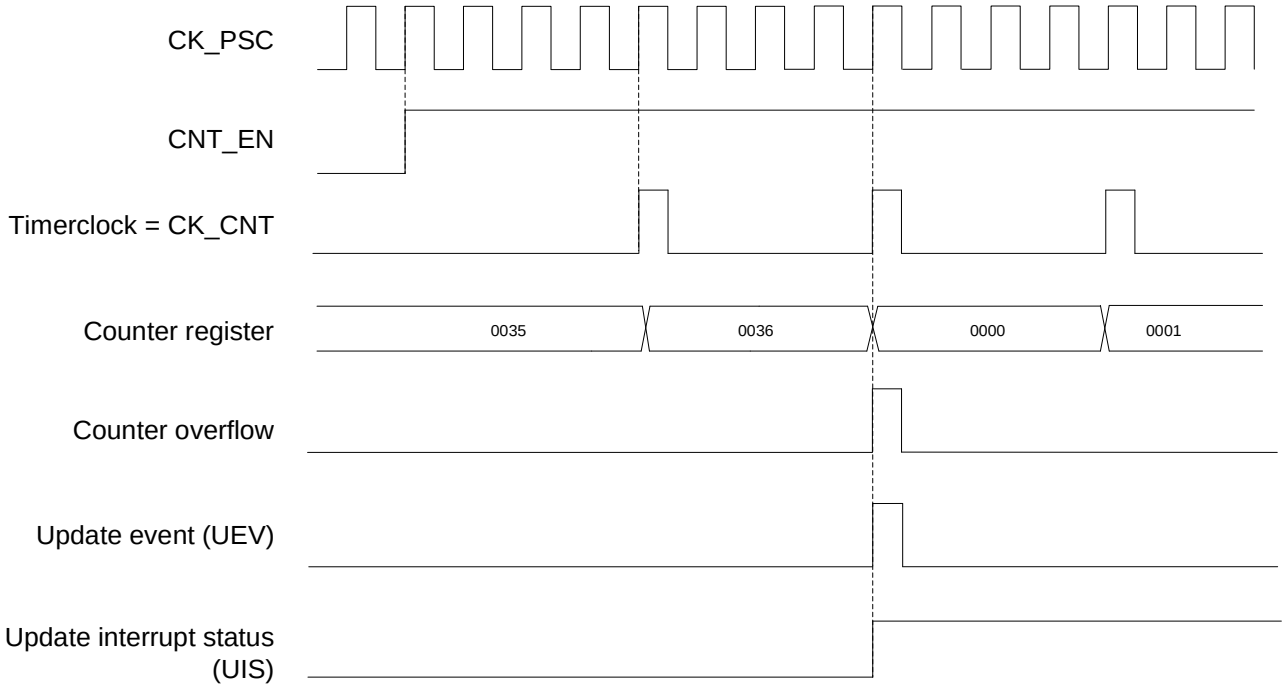


Figure 4.10-7: Counter timing diagram, internal clock divided by N

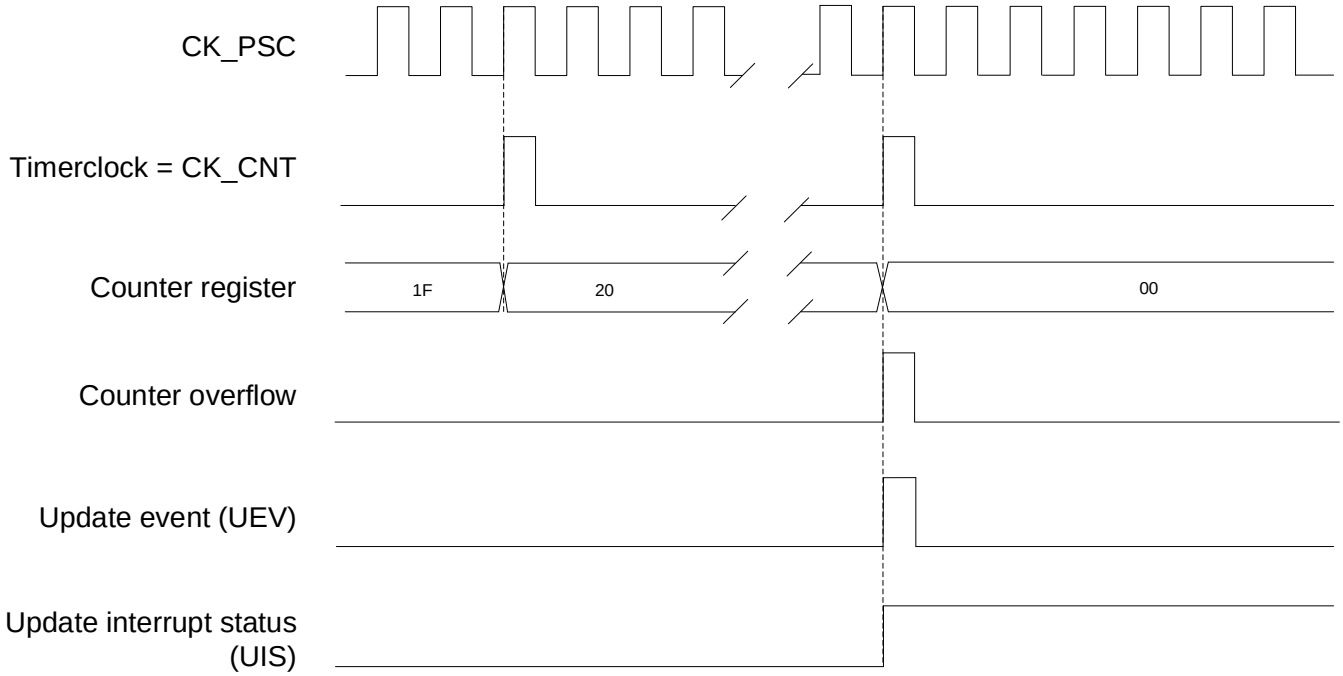


Figure 4.10-8: Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

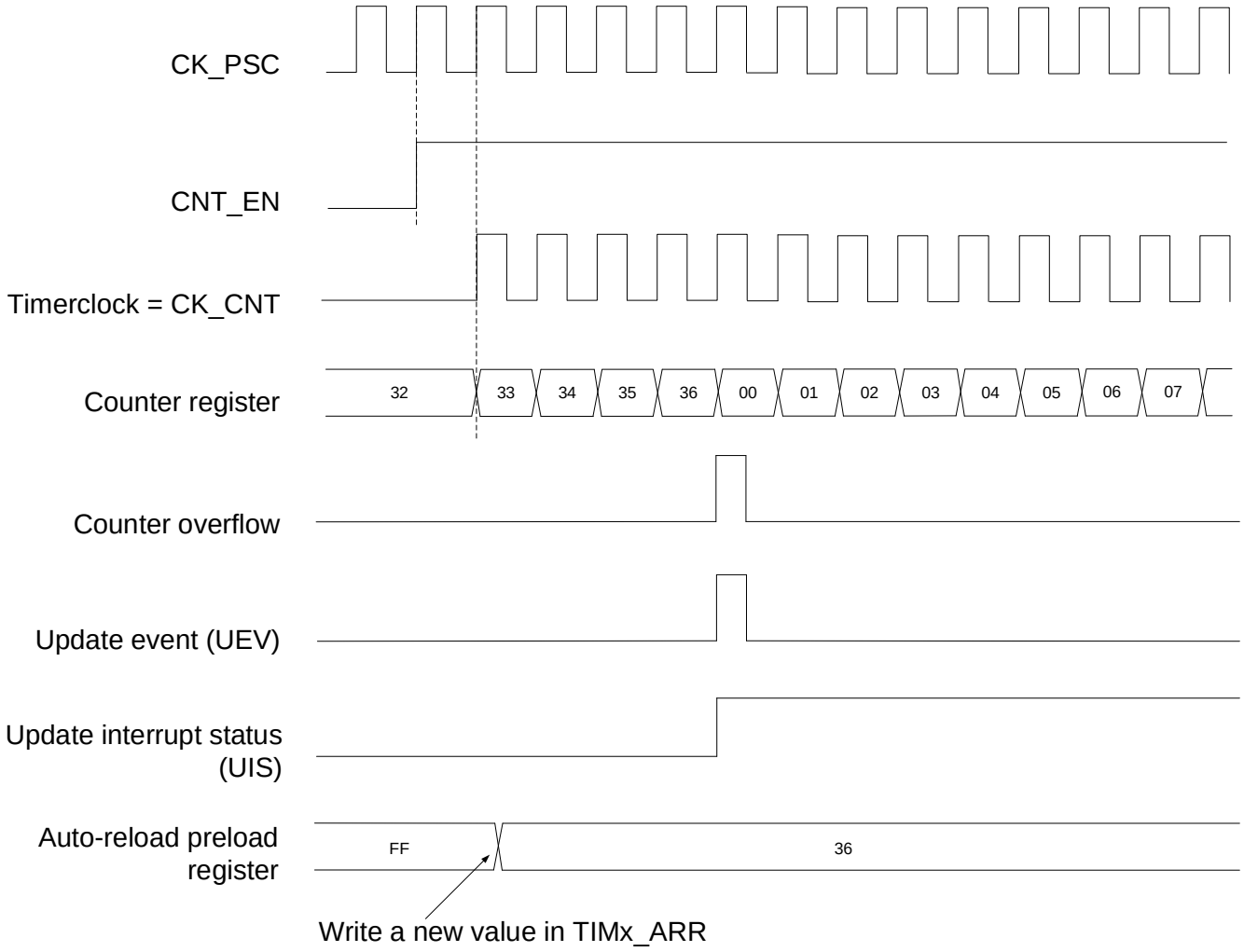
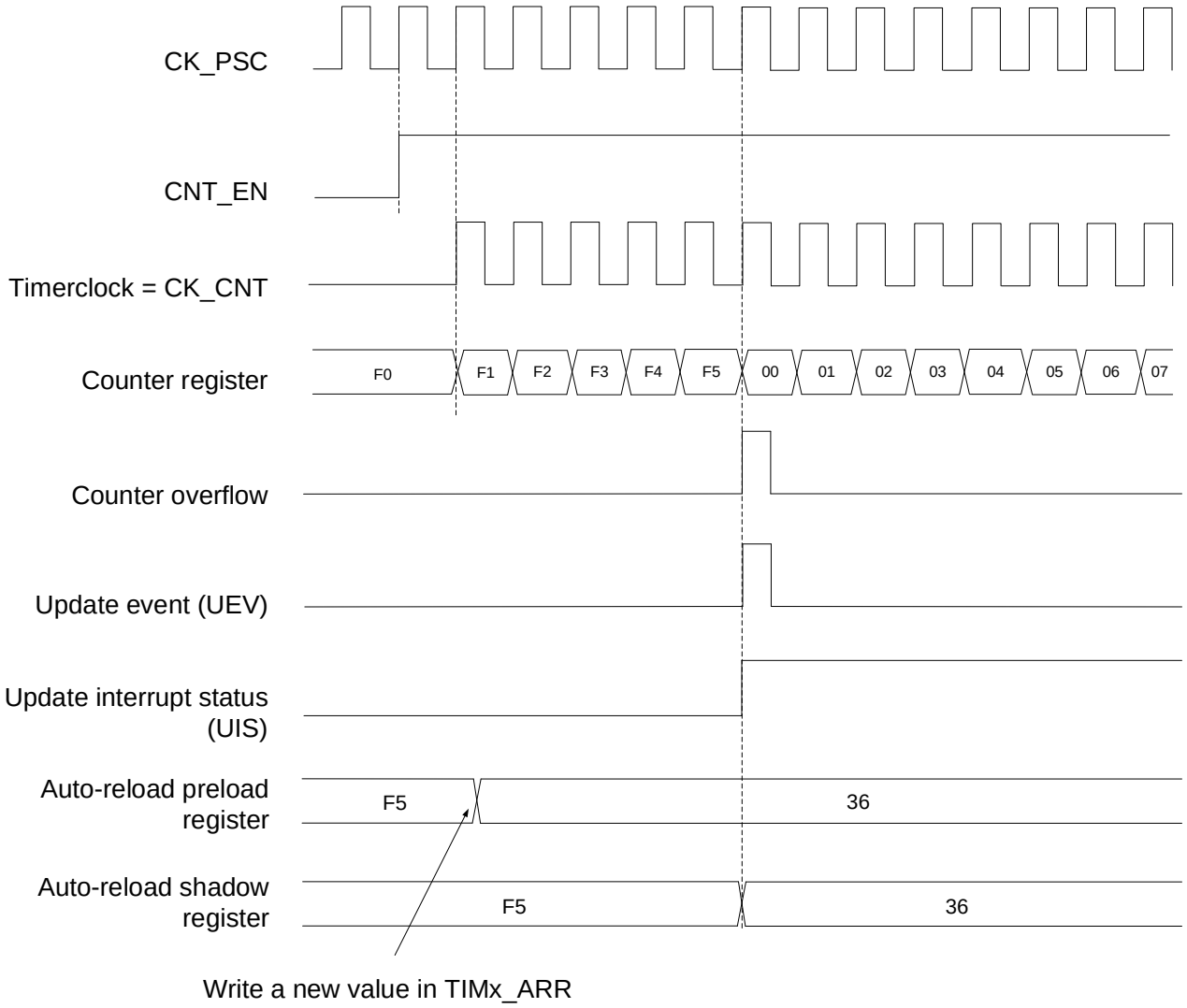


Figure 4.10-9: Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)



DOWNCOUNTING MODE

In downcounting mode, the counter counts from the auto-reload value (content of the **TIMx_ARR** register) down to 0, then restarts from the auto-reload value and generates a counter underflow event.

Setting the UG bit in the **TIMx_EGR** register (by software or by using the slave mode controller) also generates an update event.

The UEV update event can be disabled by software by setting the UDIS bit in **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 restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).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 registers are updated and the update status (UIS bit in TIMx_ISC register) is set:

- 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

The following figures show some examples of the counter behavior for different clock frequencies when **TIMx_ARR**=0x36.

Figure 4.10-10: Counter timing diagram, internal clock divided by 1

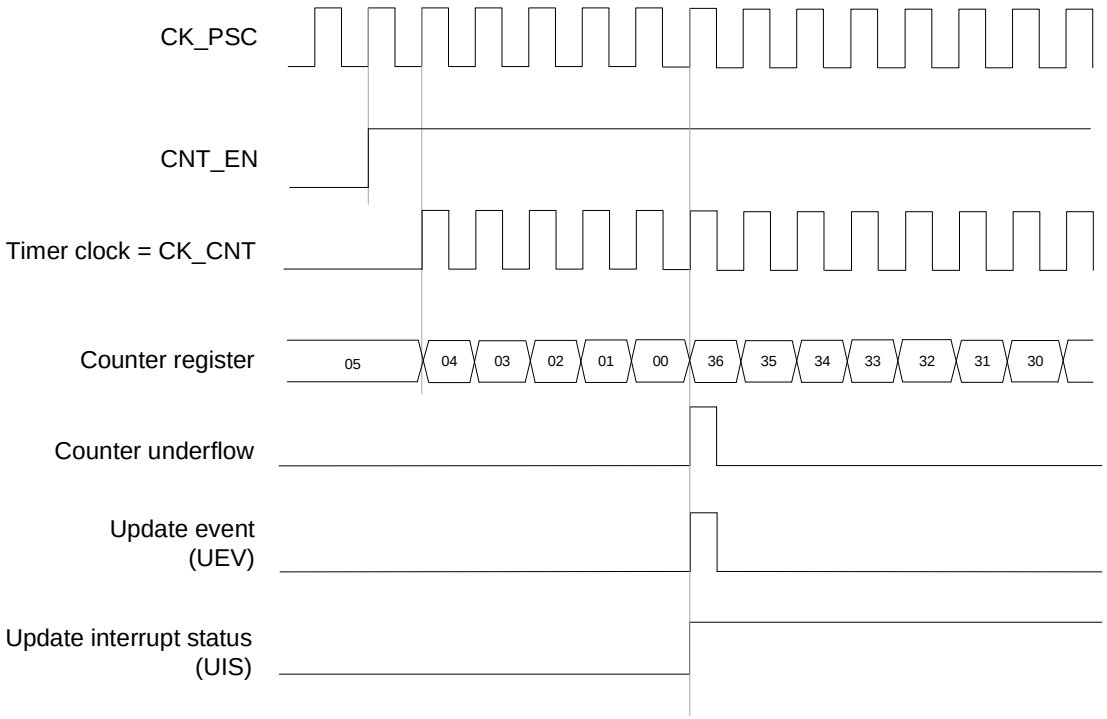


Figure 4.10-11: Counter timing diagram, internal clock divided by 2

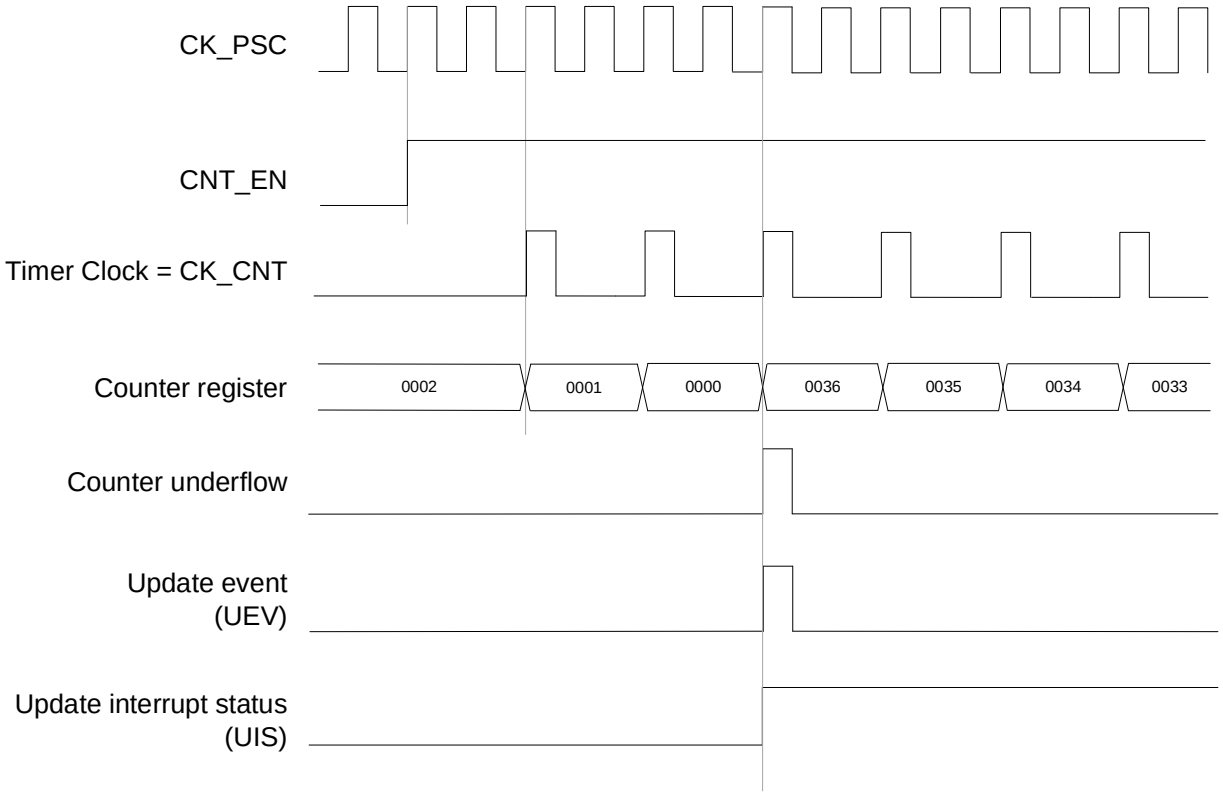


Figure 4.10-12: Counter timing diagram, internal clock divided by 4

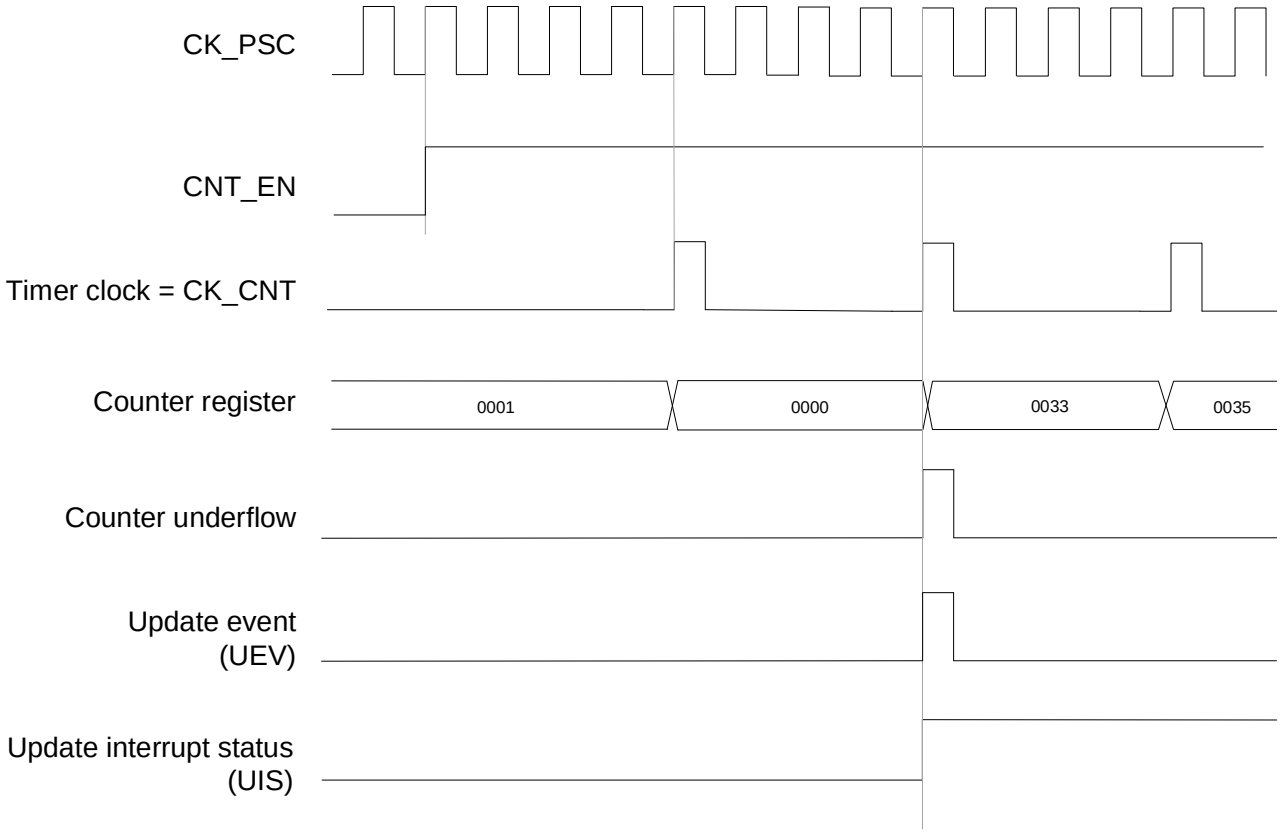


Figure 4.10-13: Counter timing diagram, internal clock divided by N

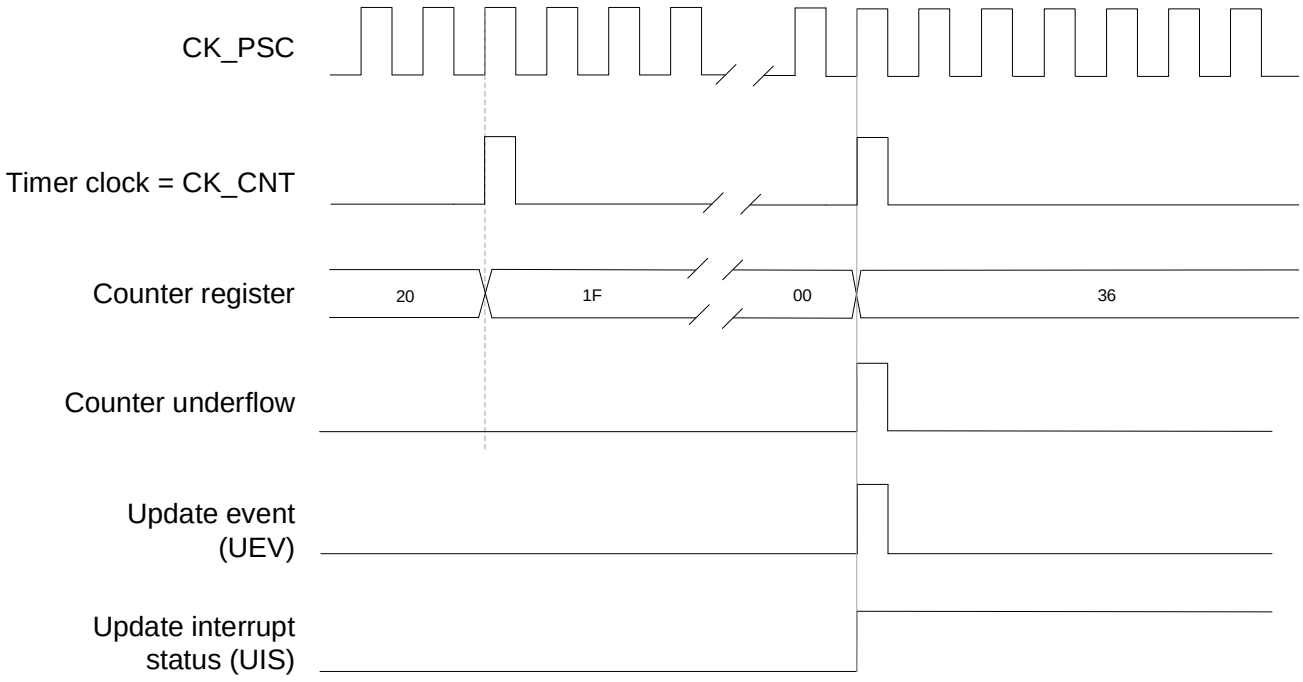
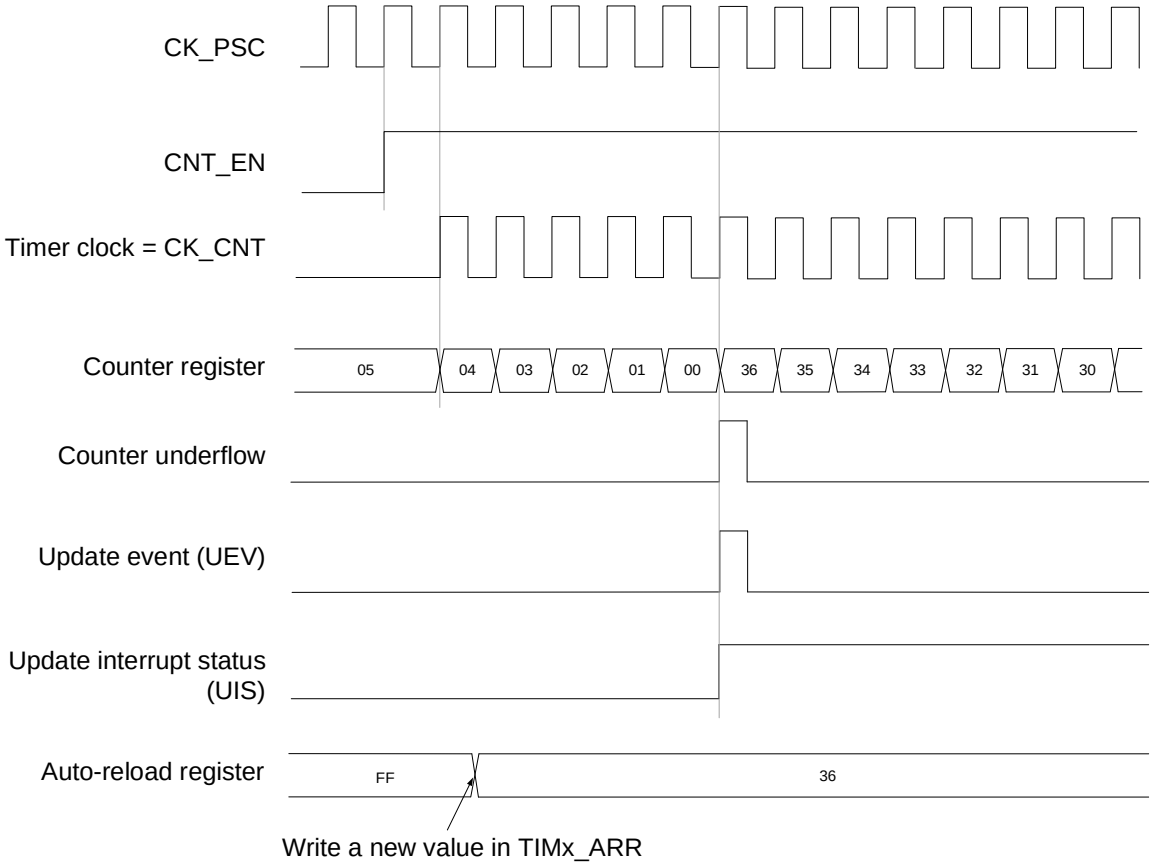


Figure 4.10-14: Counter timing diagram, update event when repetition counter is not used



CENTER-ALIGNED MODE (UP/DOWN COUNTING)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the **TIMx_ARR** register) - 1, generates a counter overflow event, then counts from the auto-reload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

Center-aligned mode is active when the CMS bits in **TIMx_CR1** register are not equal to '00'. The Output compare interrupt status of channels configured in output is set when: the counter counts down (Center aligned mode 1, CMS = “01”), the counter counts up (Center aligned mode 2, CMS = “10”) the counter counts up and 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 update 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 counting up and down, based on the current auto-reload value.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 registers are updated and the update status (UIS bit in TIMx_ISC register) is set:

- 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 if the update source is a counter overflow, the auto-reload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value).

Figure 4.10-15: Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

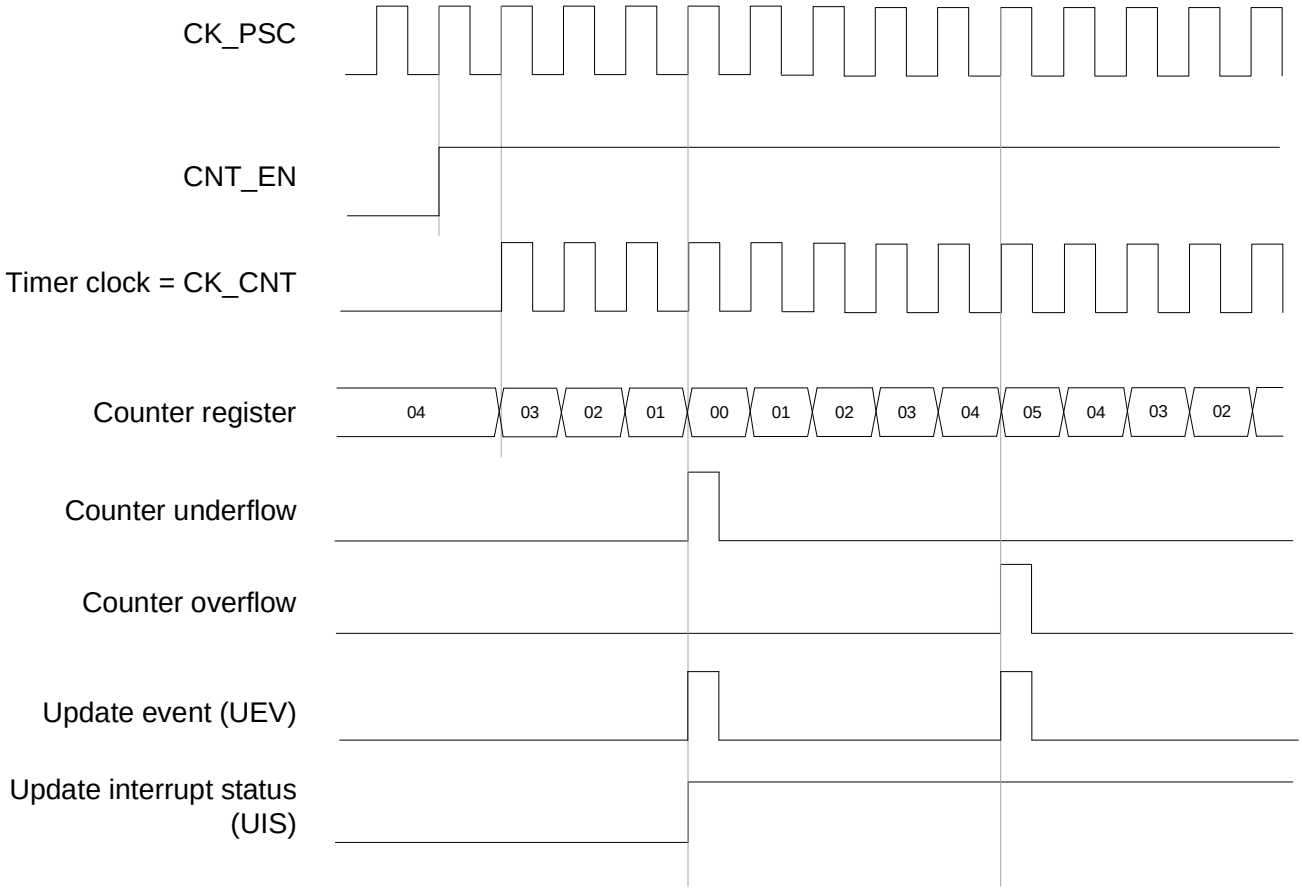


Figure 4.10-16: Counter timing diagram, internal clock divided by 2

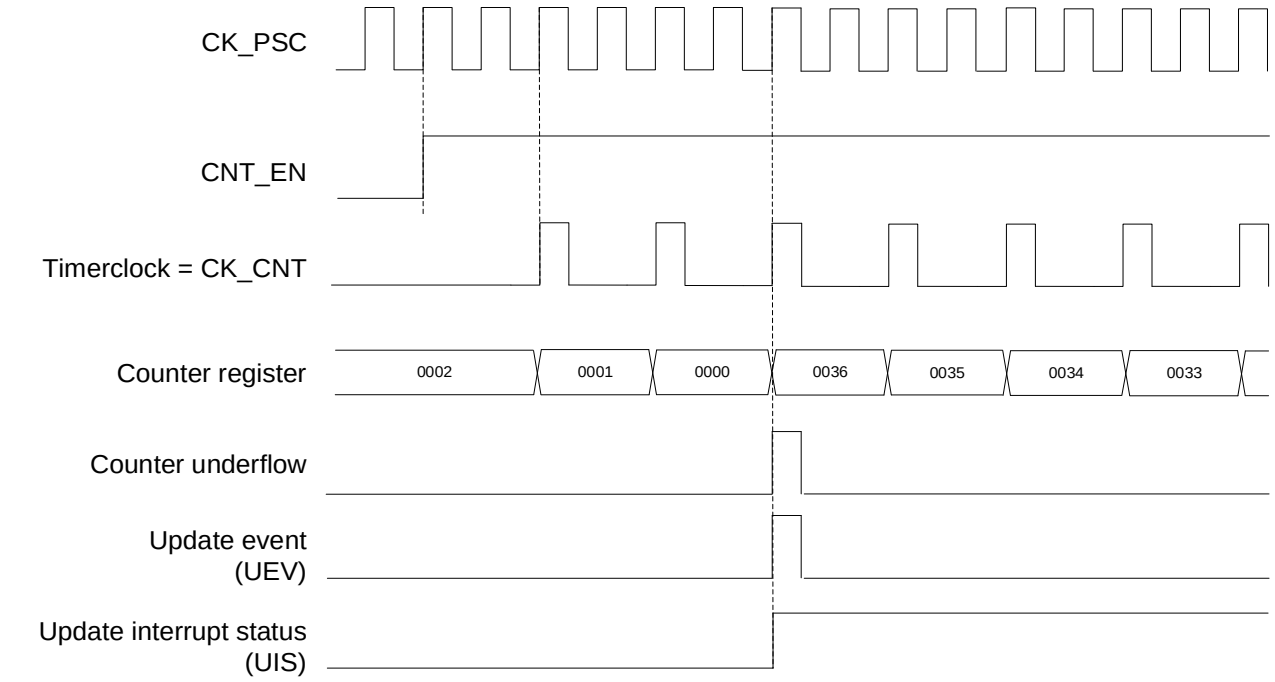
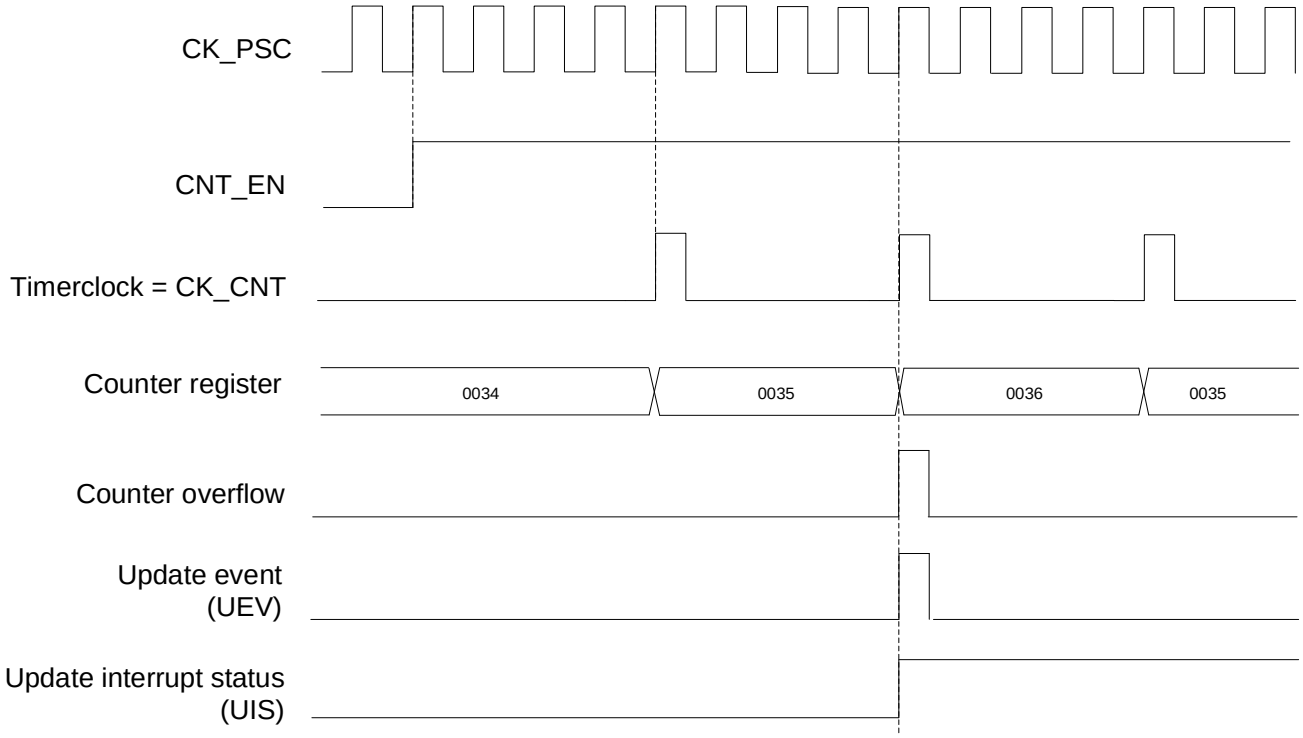


Figure 4.10-17: Counter timing diagram, internal clock divided by 4, TIMx_ARR = 0x36



Note: Here, center_aligned mode 2 or 3 is updated with an UIS on overflow

1. Center-aligned mode 2 or 3 is used with an UIS on overflow

Figure 4.10-18: Counter timing diagram, internal clock divided by N

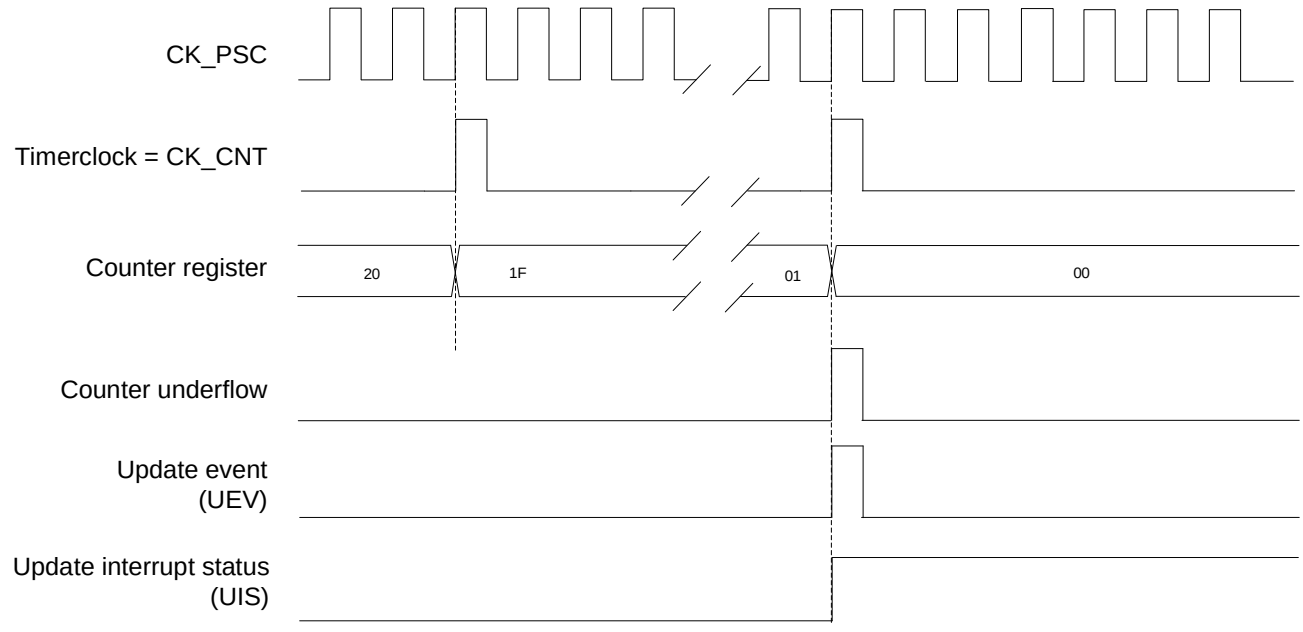


Figure 4.10-19: Counter timing diagram, update event with ARPE=1 (counter underflow)

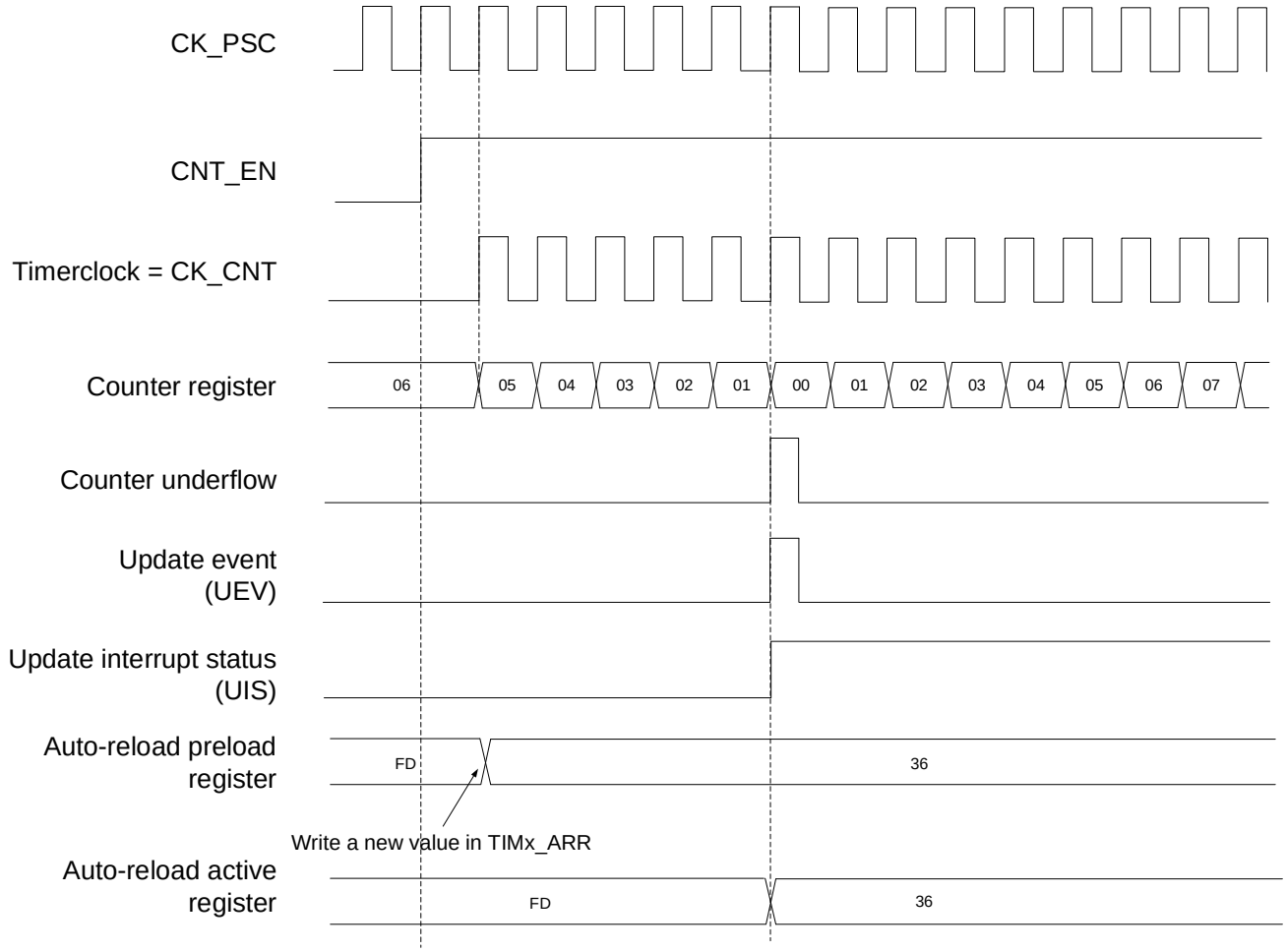
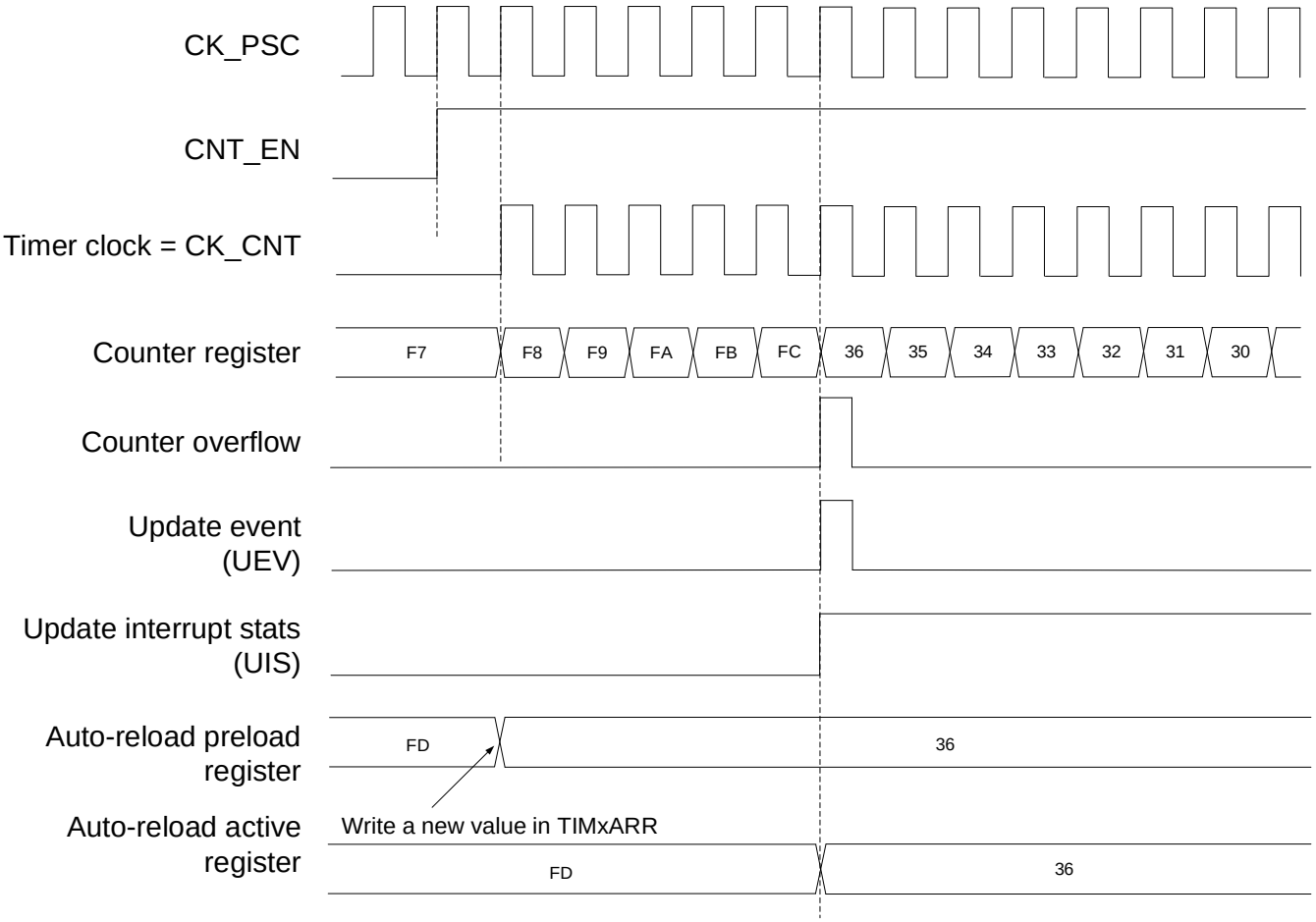


Figure 4.10-20: Counter timing diagram, Update event with ARPE=1 (counter overflow)



4.10.2.4 CLOCK SOURCES

The counter clock can be provided by the following clock sources:

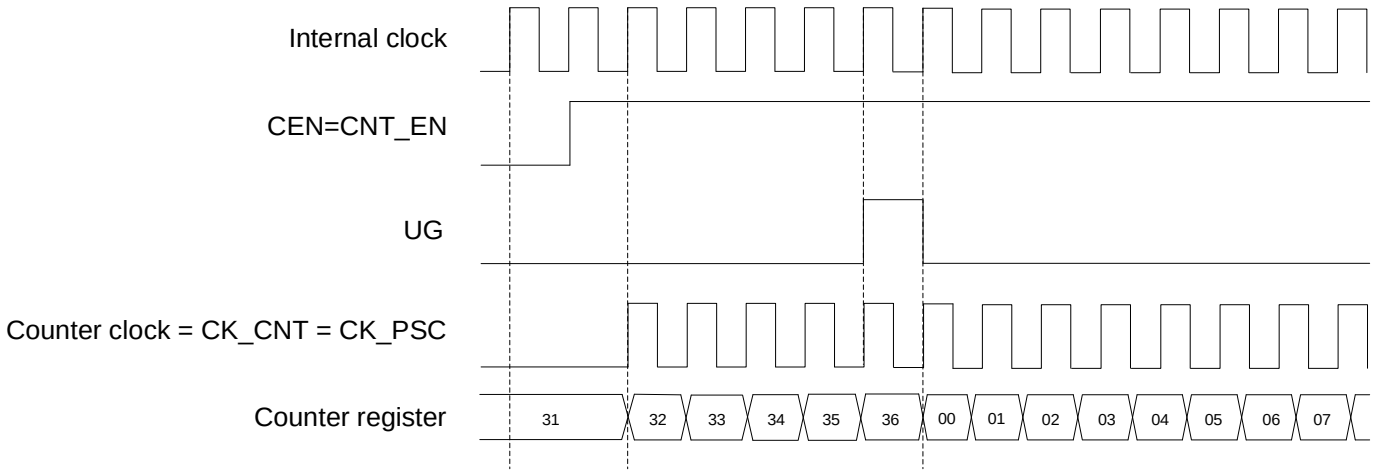
- Internal clock (CK_INT)
- External clock mode1: external input pin
- External clock mode2: external trigger input ETR
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, you can configure Timer 1 to act as a prescaler for Timer 2.

INTERNAL CLOCK SOURCE (CK_INT)

If the slave mode controller is disabled (SMS=000 in the **TIMx_SMCR** register), then the *CEN*, *DIR* (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 *UG* which remains cleared automatically). As soon as the *CEN* bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

Figure 10-21 shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

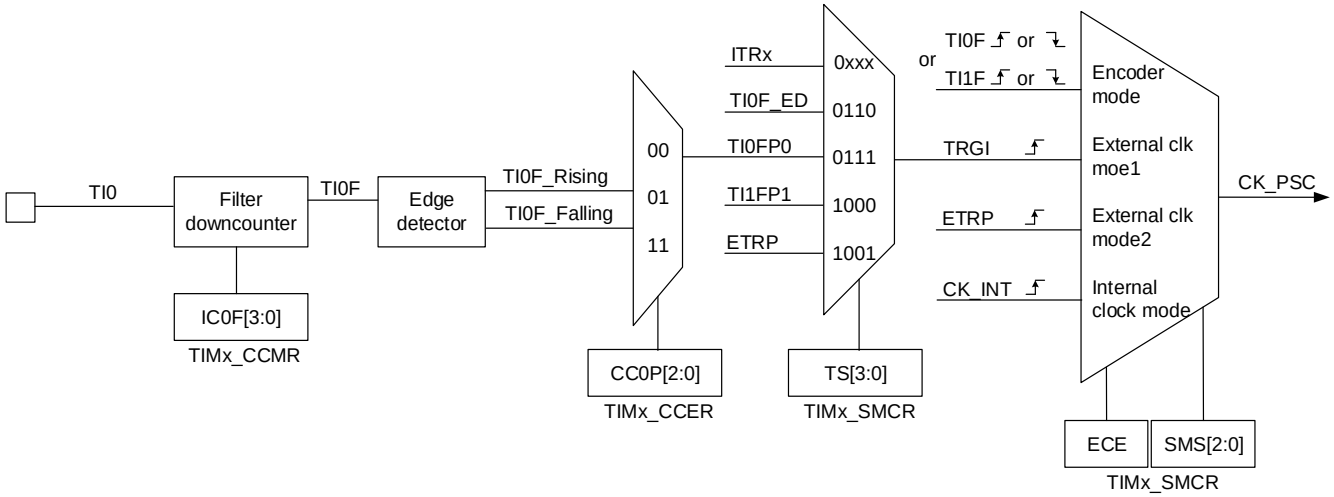
Figure 4.10-21: Control circuit in normal mode, internal clock divided by 1



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 4.10-22: TI0 external clock connection example



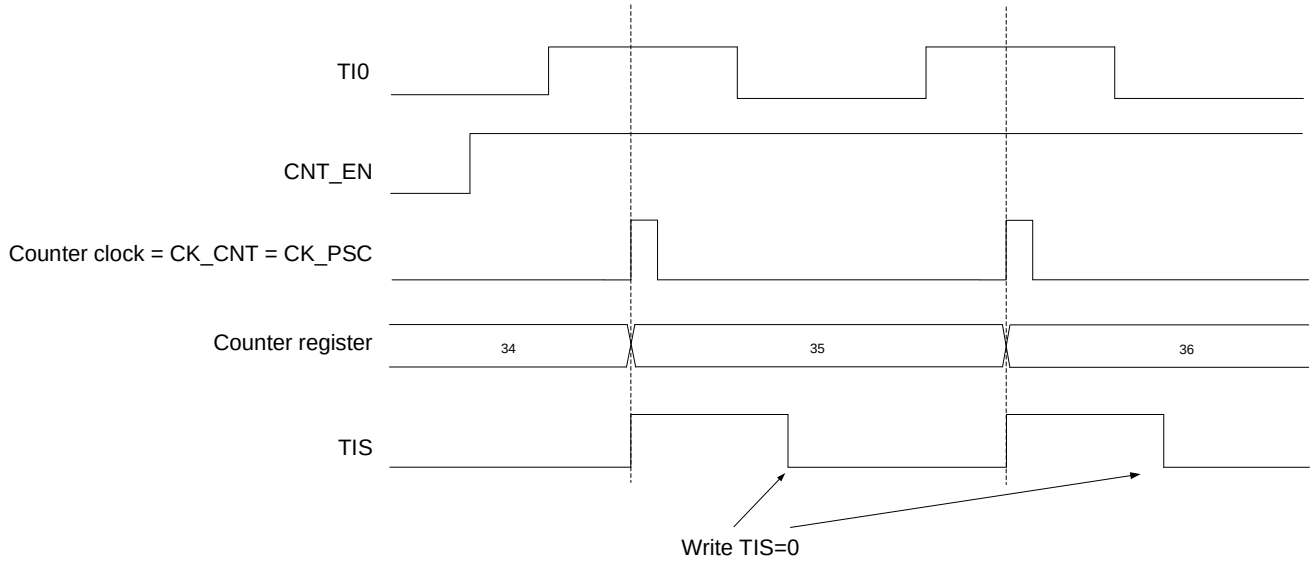
For example, to configure the upcounter to count in response to a rising edge on the TI0 input, use the following procedure:

1. Configure channel 0 to detect rising edges on the TI0 input by writing CC0S = '01' in the TIMx_CCMR register.
2. Configure the input filter duration by writing the IC0F[3:0] bits in the TIMx_CCMR register (if no filter is needed, keep IC0F=0000).
3. Select rising edge polarity by writing CC0P=00 in the TIMx_CCER register.
4. Configure the timer in external clock mode 1 by writing SMS=111 in the TIMx_SMCR register.
5. Select TI0 as the trigger input source by writing TS=0111 in the TIMx_SMCR register.
6. Enable the counter by writing CEN=1 in the TIMx_CR1 register.

When a rising edge occurs on TI0, the counter counts once and the TIS status is set.

The delay between the rising edge on TI0 and the actual clock of the counter is due to the resynchronization circuit on TI0 input.

Figure 4.10-23: Control circuit in external clock mode 1

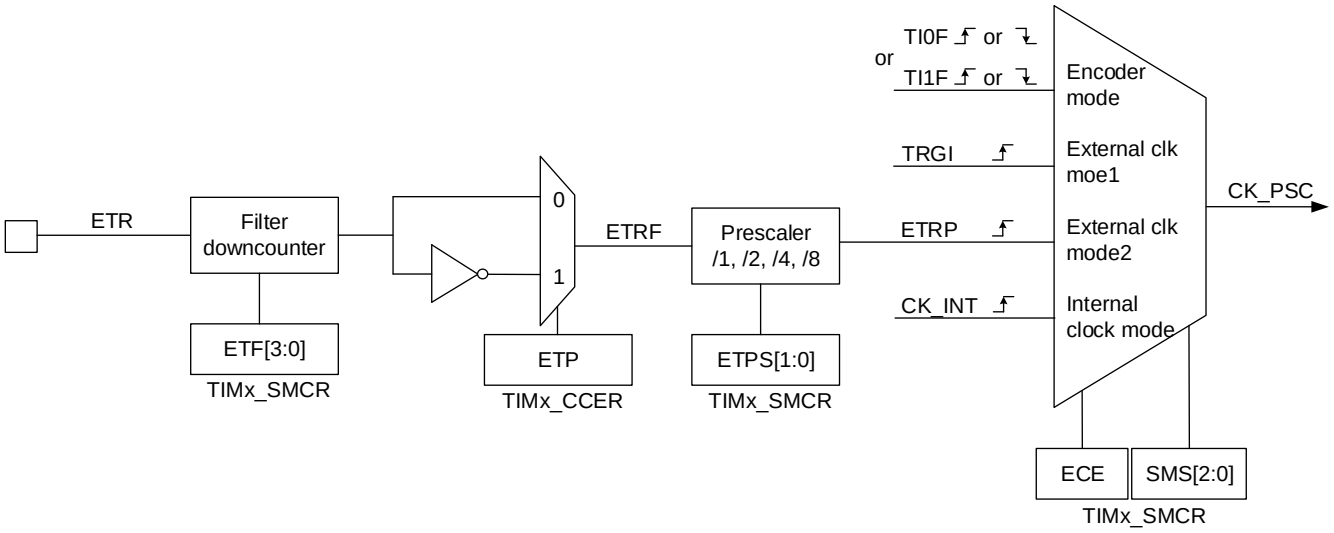


EXTERNAL CLOCK SOURCE MODE 2

This mode is selected by writing ECE=1 in the **TIMx_SMCR** register.

The counter can count at each rising or falling edge on the external trigger input ETR.

Figure 4.10-24: TI0 gives an overview of the external trigger input block.

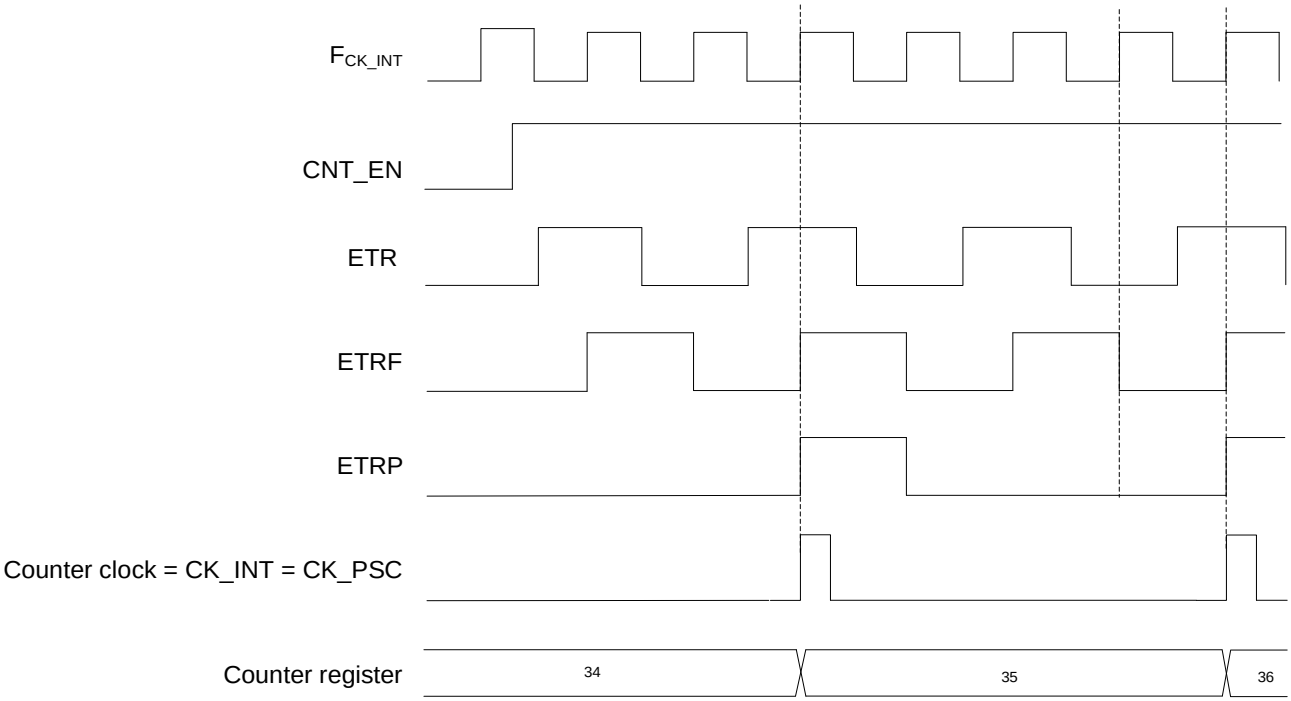


For example, to configure the upcounter to count each 2 rising edges on ETR, use the following procedure:

1. As no filter is needed in this example, write ETF[3:0]=0000 in the TIMx_SMCR register.
2. Set the prescaler by writing ETPS[1:0]=01 in the TIMx_SMCR register
3. Select rising edge detection on 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 TIMx_CR1 register.

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 4.10-25: Control circuit in external clock mode 2



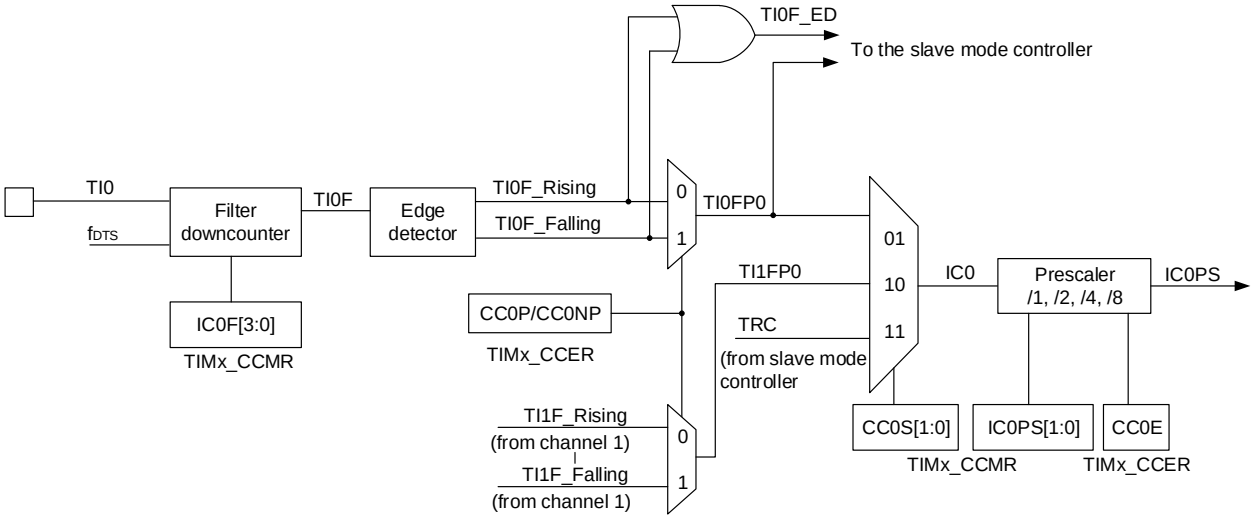
4.10.2.5 CAPTURE/COMPARE CHANNELS

Each Capture/Compare channel, is built around a capture/compare register (including a shadow register), an input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The following figure gives an overview of one Capture/Compare channel

The input stage samples the corresponding TIx input to generate a filter 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 4.10-26: Capture/compare channel (example: channel 0 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 4.10-27: Capture/compare channel 0 main circuit

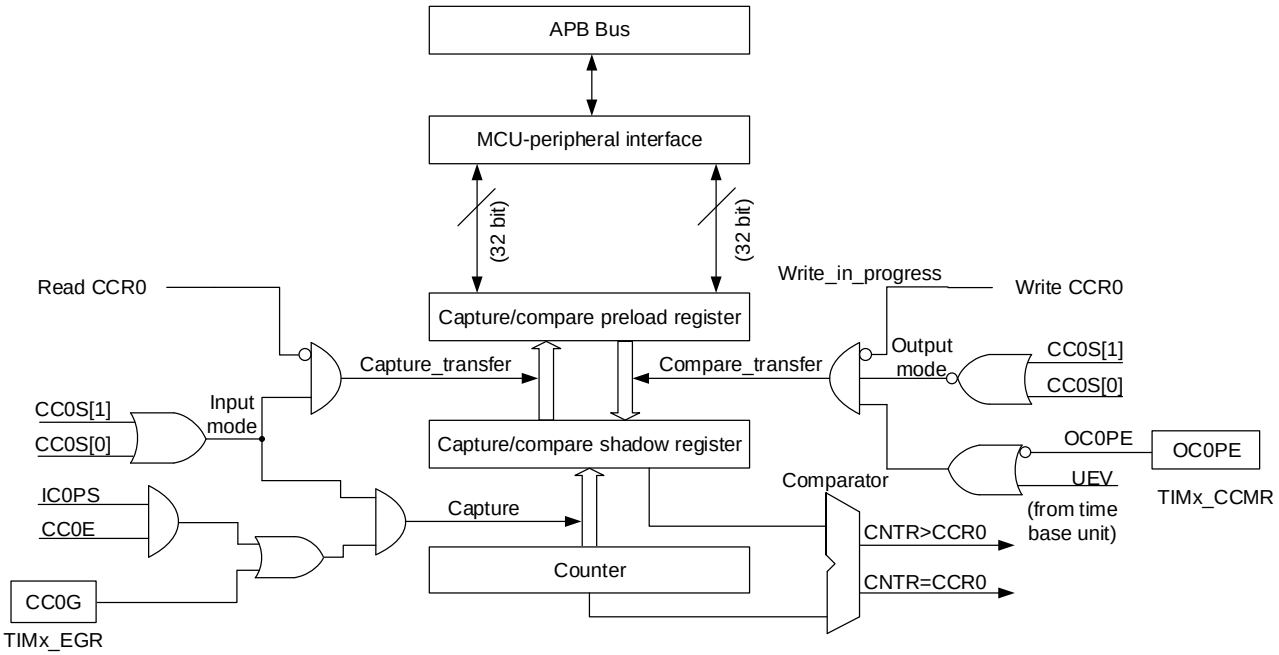
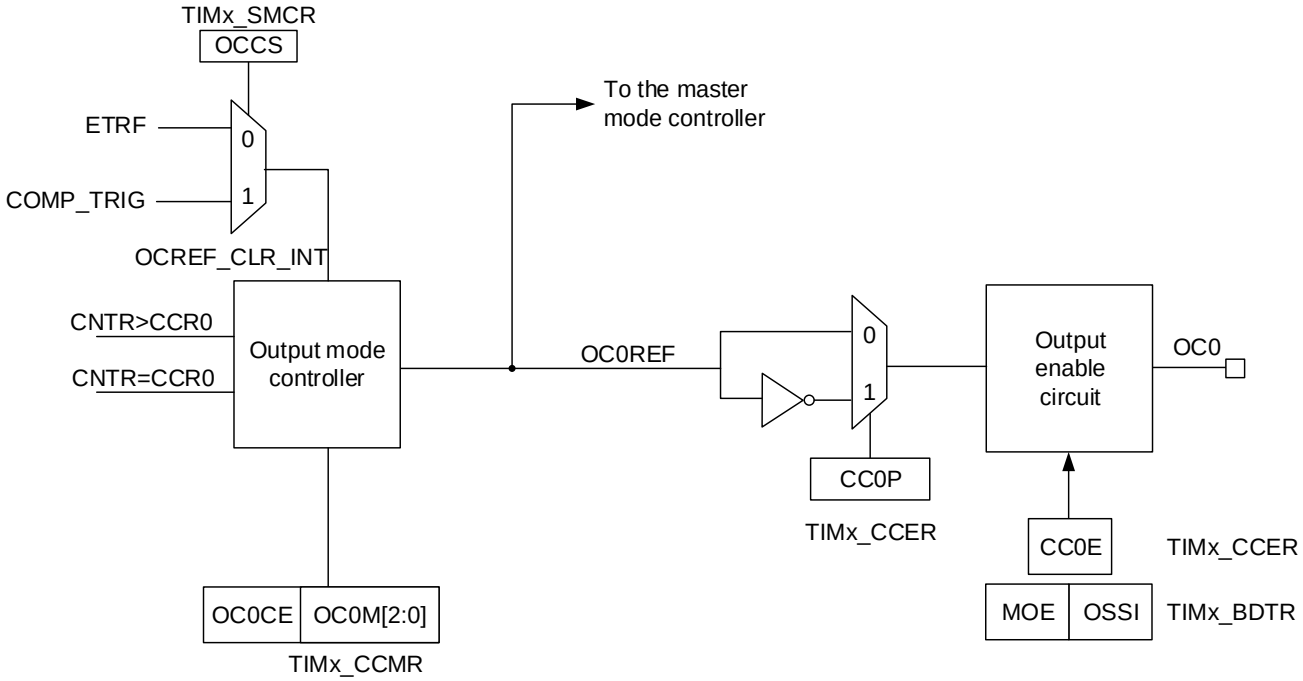


Figure 4.10-28: Output stage of capture/compare channel (channel 0 to 2)



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, 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.

4.10.2.6 INPUT CAPTURE MODE

In input capture mode, the Capture/Compare Registers (**TIMx_CCRx**) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding interrupt can be sent if they are enabled.

The following example shows how to capture the counter value in **TIMx_CCR0** when TI0 input rises. To do this, use the following procedure:

- Select the active input: TIMx_CCR0 must be linked to the TI0 input, so write the CC0S bits to 01 in the TIMx_CCMR register. As soon as CC0S becomes different from 00, the channel is configured in input and the TIMx_CCR0 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_CCMR register). Let's imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI0 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC0F bits to 0011 in the TIMx_CCMR register.
- Select the edge of the active transition on the TI0 channel by writing CC0P and CC0NP bits to 0 in the TIMx_CCER register (rising edge in this case).
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC0PS bits to '00' in the TIMx_CCMR register).
- Enable capture from the counter into the capture register by setting the CC0E bit in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC0IE bit in the TIMx_IER register.

When an input capture occurs:

- The TIMx_CCR0 register gets the value of the counter on the active transition.
- CC0IS status is set (interrupt status). CC0OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC0IE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture status. This is to avoid missing an overcapture which could happen after reading the status and before reading the data.

NOTE: ICx interrupt requests can be generated by software by setting the corresponding CCxG bit in the **TIMx_EGR** register.

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.

4.10.2.7 PWM INPUT MODE

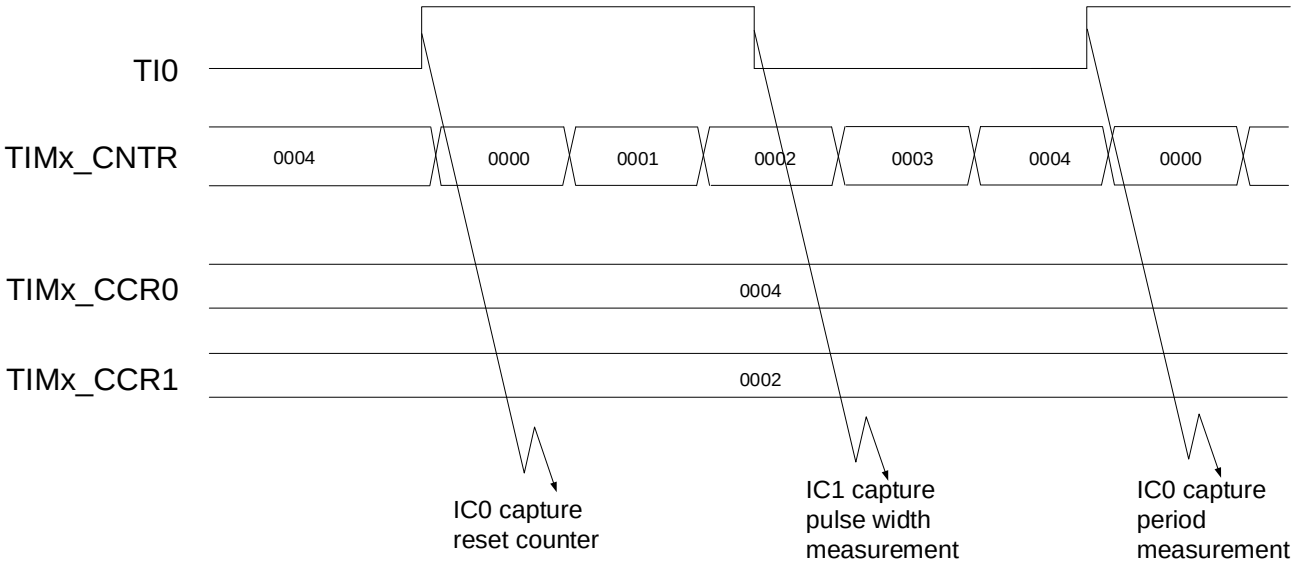
This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input.
- These 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 **TIMx_CCR0** register) and the duty cycle (in **TIMx_CCR1** register) of the PWM applied on TI0 using the following procedure (depending on CK_INT frequency and prescaler value):

- Select the active input for **TIMx_CCR0**: write the CC0S bits to 01 in the **TIMx_CCMR** register (TI0 selected).
- Select the active polarity for TI0FP0 (used both for capture in **TIMx_CCR0** and counter clear): write the *CC0P* bit to '0' (active on rising edge).
- Select the active input for **TIMx_CCR1**: write the *CC1S* bits to 10 in the **TIMx_CCMR** register (TI0 selected).
- Select the active polarity for TI0FP1 (used for capture in **TIMx_CCR1**): write the *CC1P* bit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 111 in the **TIMx_SMCR** register (TI0FP0 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 CC0E and CC1E bits to '1' in the **TIMx_CCER** register.

Figure 4.10-29: PWM input mode timing



4.10.2.8 FORCED OUTPUT MODE

In output mode (*CCxS* bits = 00 in the **TIMx_CCMR** register), each output compare signal (OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison 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_CCMR** 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 100 in the **TIMx_CCMR** register.

Anyway, the comparison between the **TIMx_CCRx** shadow register and the counter is still performed and allows the status to be set. Interrupt can be sent accordingly. This is described in the output compare mode section below.

4.10.2.9 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_CCMR** register). 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.
- Generates an interrupt if the corresponding interrupt is enable (CCXIE bit in the **TIMx_IER** register).

The **TIMx_CCRx** registers can be programmed with or without preload registers using the *OCxPE* bit in the **TIMx_CCMR** register.

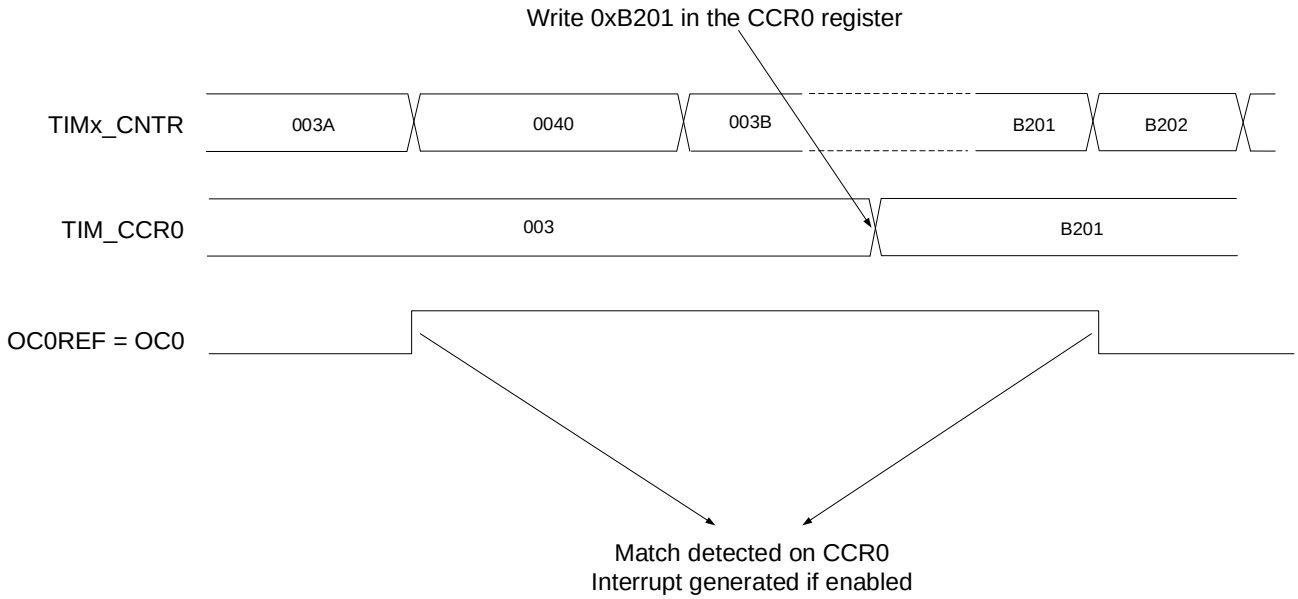
In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Shot mode).

Procedure:

- Select the counter clock (internal, external, prescaler).
- Write the desired data in the **TIMx_ARR** and **TIMx_CCRx** registers.
- Set the CCxIE bit in the **TIMx_IER** if an interrupt request is to be generated.
- Select the output mode. For example:
  - Write OCxM = 011 to toggle OCx output pin when CNTR matches CCRx
  - Write OCxPE = 0 to disable preload register
  - Write CCxP = 0 to select active high polarity
  - Write CCxE = 1 to enable the output
- Enable the counter by setting the CEN bit in the **TIMx_CR1** register.

The **TIMx_CCRx** register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE=’0’, else **TIMx_CCRx** shadow register is updated only at the next update event UEV). An example is given in

Figure 4.10-30: Output compare mode, toggle on OC0



4.10.2.10 PWM MODE

Pulse Width Modulation mode allows you to generate a signal with a frequency determined by the value of the **TIMx_ARR** register and a duty cycle determined by the value of 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_CCMR** register. You must enable the corresponding preload register by setting the OCxPE bit in the **TIMx_CCMR** register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the *ARPE* bit in the **TIMx_CR1** register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, you have to initialize all the registers by setting the UG bit in the **TIMx_EGR** register.

OCx polarity is software programmable using the *CCxP* bit in the **TIMx_CCER** register. It can be programmed as active high or active low. OCx output is enabled by a combination of the *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 (1 or 2), **TIMx_CNTR** and **TIMx_CCRx** are always compared to determine whether **TIMx_CCRx ≤TIMx_CNTR** or **TIMx_CNTR ≤TIMx_CCRx** (depending on the direction of the counter).

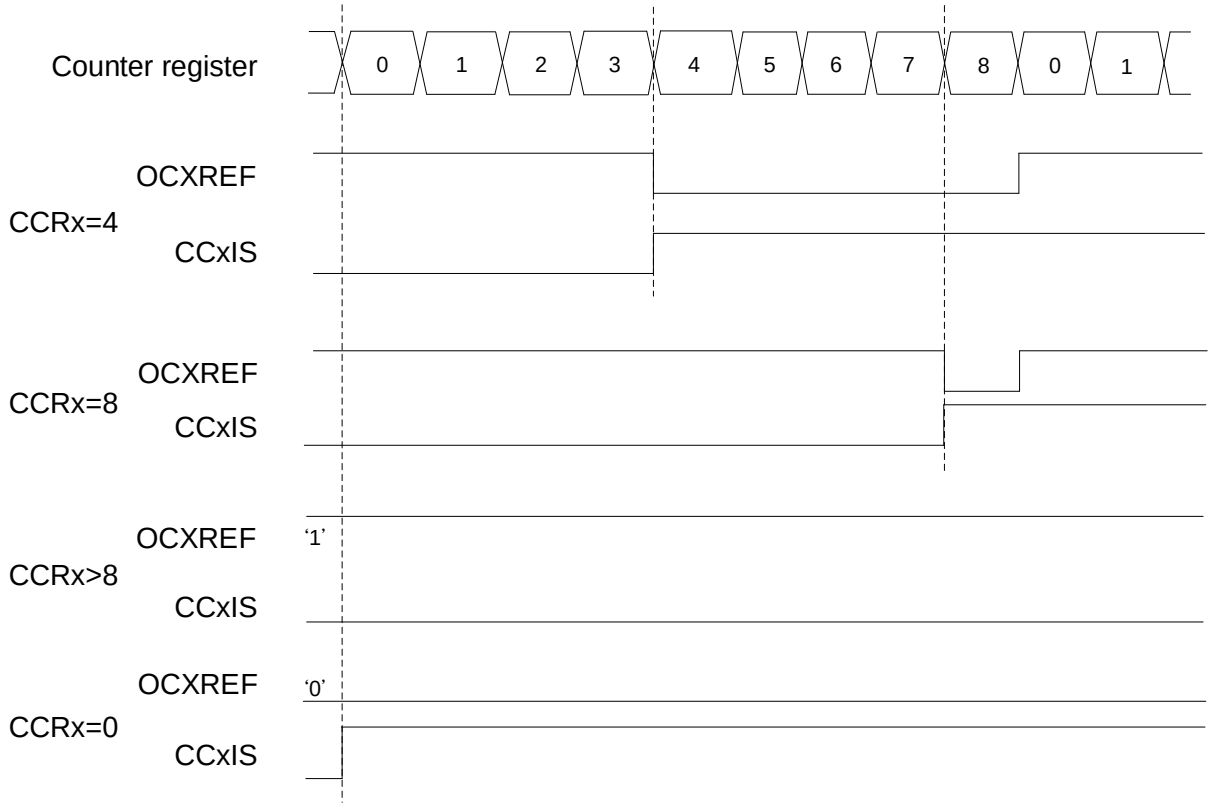
The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the *CMS* bits in the **TIMx_CR1** register.

PWM EDGE-ALIGNED MODE

- Upcounting configuration

Upcounting is active 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_CNTR < 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 10-31 shows some edge-aligned PWM waveforms in an example where **TIMx_ARR**=8.

Figure 4.10-31: Edge-aligned PWM waveforms (ARR=8)



- Downcounting configuration

Downcounting is active when DIR bit in **TIMx_CR1** register is high. In PWM mode 1, the reference signal OCxRef is low as long as **TIMx_CNTR** > **TIMx_CCRx** else it becomes 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.

PWM CENTER-ALIGNED MODE

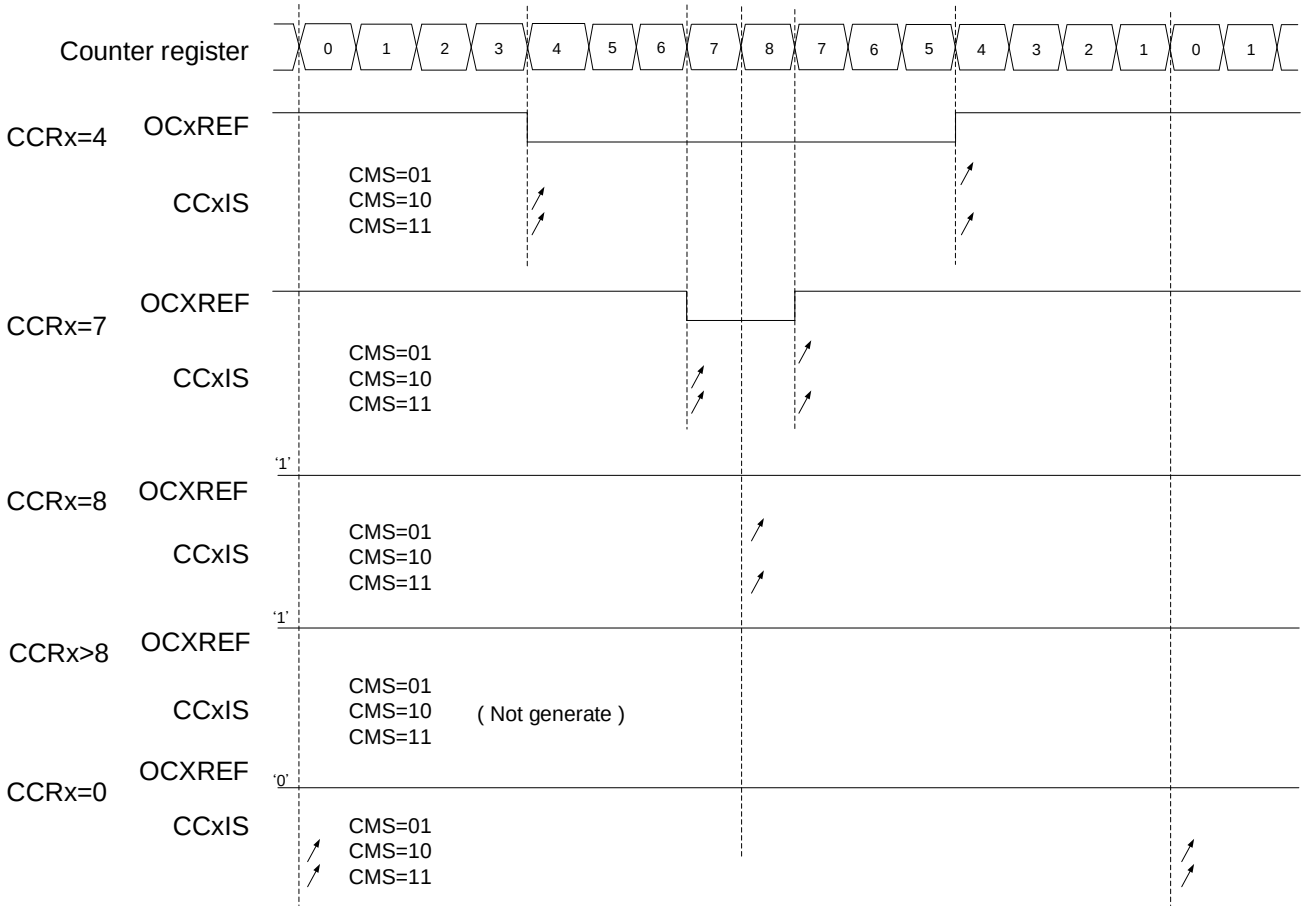
Center-aligned mode is active when the CMS bits in **TIMx_CR1** register are different from '00' (all the remaining configurations having the same effect on the OCxRef/OCx signals). The compare status is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the **TIMx_CR1** register is updated by hardware and must not be changed by software.

Figure 10-32 shows some center-aligned PWM waveforms in an example where:

- TIMx_ARR=8,
- PWM mode is the PWM mode 1,

The status is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS=01 in **TIMx_CR1** register.

Figure 4.10-32: Center-aligned PWM waveforms (ARR=8)



HINTS ON USING CENTER-ALIGNED MODE:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in 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.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:
  - The direction is not updated if you write a value in the counter that is greater than the auto-reload value (TIMx_CNTR>TIMx_ARR). For example, if the counter was counting up, it continues to count up.
  - The direction is updated if you write 0 or write the TIMx_ARR value in the counter but no Update Event UEV is generated.
- The safest way to use center-aligned mode is to generate an update by software (setting the *UG* bit in the **TIMx_EGR** register) just before starting the counter and not to write the counter while it is running.

4.10.2.11 ONE-SHOT MODE

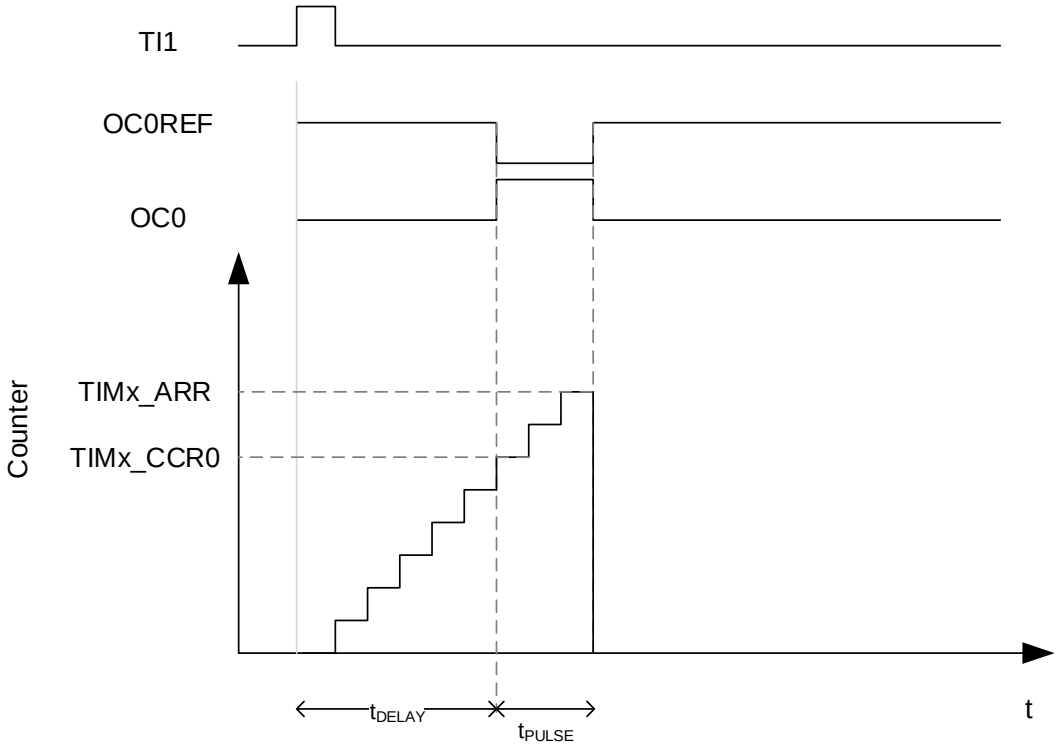
One-shot mode (OSM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. You select one-shot mode by setting the OSM 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:

- In upcounting: CNTR < CCRx ≤ ARR (in particular, 0 < CCRx)
- In downcounting: CNTR > CCRx

Figure 4.10-33: Example of one shot mode



For example you may want to generate a positive pulse on OC0 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI1 input pin.

Let's use TI1FP1 as trigger:

- Map TI1FP1 to TI1 by writing CC1S='01' in the **TIMx_CCMR** register.
- TI1FP1 must detect a rising edge, write CC1P='0' and CC1NP='0' in the **TIMx_CCER** register.
- Configure TI1FP1 as trigger for the slave mode controller (TRGI) by writing TS='1000' in the **TIMx_SMCR** register.
- TI1FP1 is used to start the counter by writing SMS to '110' in the **TIMx_SMCR** register (trigger mode).

The OSM 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 **TIMx_CCR0** register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (**TIMx_ARR** - **TIMx_CCR1**+1).
- Let's say you 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 you enable PWM mode 2 by writing OC0M=111 in the **TIMx_CCMR** register. You can optionally enable the preload registers by writing OC0PE='1' in the **TIMx_CCMR** register and ARPE in the **TIMx_CR1** register. In this case you have to write the compare value in the **TIMx_CCR0** 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 TI1. CC0P is written to '0' in this example.

In our example, the *DIR* and *CMS* bits in the **TIMx_CR1** register should be low.

You only want 1 shot (Single mode), so you write '1 in the OSM 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). When OSM bit in the **TIMx_CR1** register is set to '0', so the Repetitive Mode is selected.

4.10.2.12 CLEARING THE OCXREF SIGNAL ON AN EXTERNAL EVENT

One-shot mode (OSM) is a particular case of the previous modes. It allows the counter to be started in response to a

The OCxREF signal of a given channel can be cleared when a high level is applied on the OCREF_CLR_INPUT (OCxCE enable bit in the corresponding **TIMx_CCMR** register set to 1). OCxREF remains low until the next update event (UEV) occurs. This function can only be used in Output compare and PWM modes. It does not work in Forced mode.

OCREF_CLR_INPUT can be selected between the COMP_TRIG input and ETRF (ETR after the filter) by configuring the OCCS bit in the **TIMx_SMCR** register.

When ETRF is chosen, ETR must be configured as follows:

The OCxREF signal for a given channel can be driven low by applying a high level to the ETRF input (OCxCE enable bit of the corresponding **TIMx_CCMR** register set to '1'). The OCxREF signal remains low until the next update event, UEV, occurs.

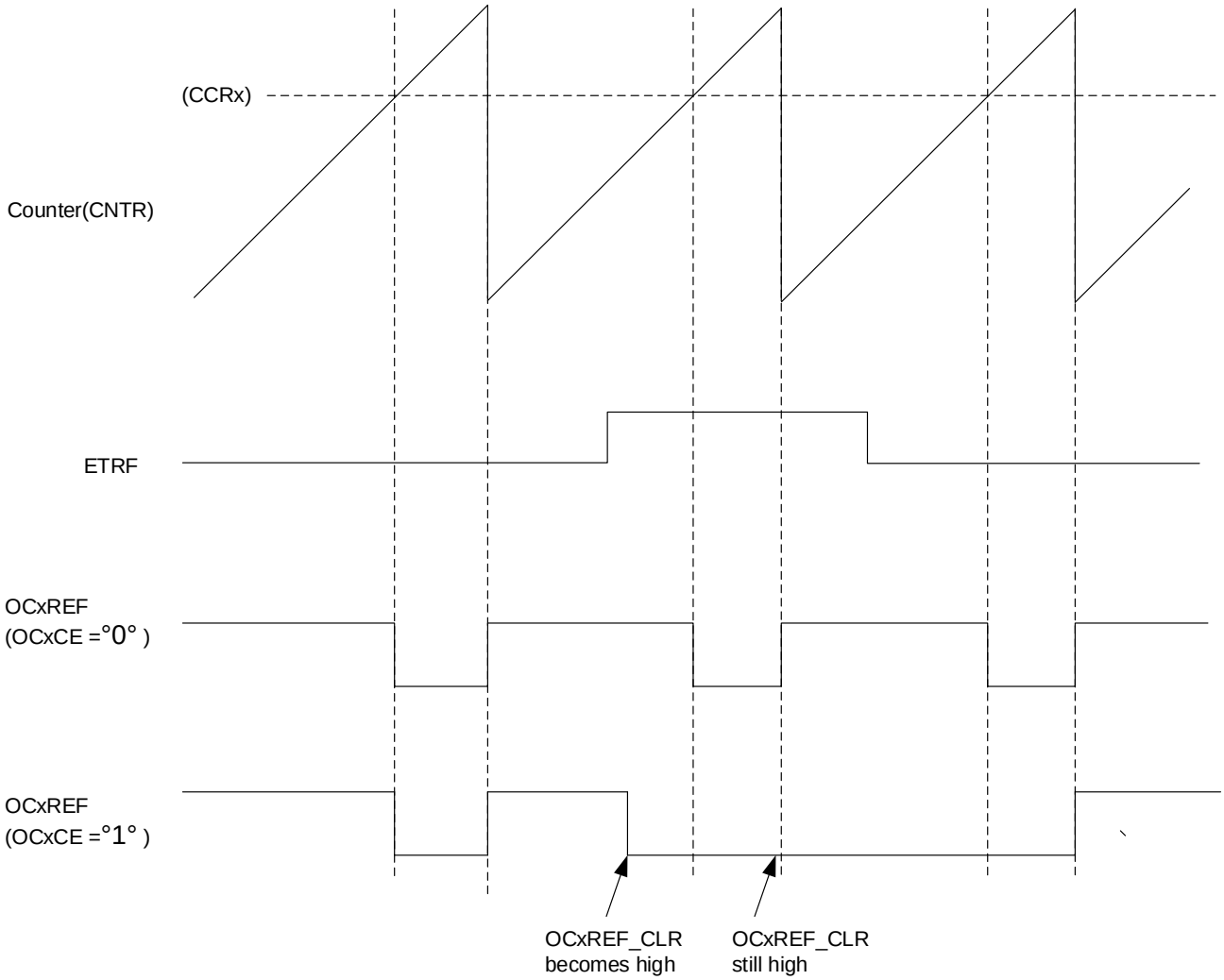
This function can only be used in output compare and PWM modes, and does not work in forced mode.

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follow:

1. The External Trigger Prescaler should be kept off: bits ETPS[1:0] of the **TIMx_SMCR** register set to '00'.
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 according to the user needs.

Figure 4.10-34 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 4.10-34: Clearing TIMx OCxREF



Note: In case of a PWM with a 100% duty cycle (if CCRx>ARR), then OCxREF is enabled again at the next counter overflow.

4.10.2.13 ENCODER INTERFACE MODE

To select Encoder Interface mode write SMS='001' in the **TIMx_SMCR** register if the counter is counting on TI0 edges only, SMS='010' if it is counting on TI1 edges only and SMS='011' if it is counting on both TI10 and TI1 edges.

The two inputs TI0 and TI1 are used to interface to an incremental encoder. The counter is clocked by each valid transition on TI0FP0 or TI1FP1 (TI0 and TI1 after input filter and polarity selection, TI0FP0=TI0 if not filtered and not inverted, TI1FP1=TI1 if not filtered and not inverted) assuming that it is enabled (CEN bit in **TIMx_CR1** register written to '1'). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the **TIMx_CR1** register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI0 or TI1), whatever the counter is counting on TI0 only, TI1 only or both TI0 and TI1.

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 you must configure **TIMx_ARR** before starting. In the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal.

In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor. The table summarizes the possible combinations, assuming TI0 and TI1 don't switch at the same time.

Table 4.10-1: Counting direction versus encoder signals

Active edge	Level on opposite signal (TI0FP0 for TI2, TI1FP1 for TI1)	TI0FP0 signal		TI1FP1 signal	
		Rising	Falling	Rising	Falling
Counting on TI0 only	High	Down	Up	No Count	No Count
	Low	Up	Down	No Count	No Count
Counting on TI1 only	High	No Count	No Count	Up	Down
	Low	No Count	No Count	Down	Up
Counting on TI0 and TI1	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be used to convert the encoder's differential outputs to digital signals. This greatly increases 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.

Figure 4.10-35 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:

- CC0S='01' (**TIMx_CCMR** register, TI0FP0 mapped on TI0).
- CC1S='01' (**TIMx_CCMR** register, TI1FP1 mapped on TI1).
- SMS='011' (**TIMx_SMCR** register, both inputs are active on both rising and falling edges).
- DIR'0' (**TIMx_CR1** register, set upcountung)
- CEN='1' (**TIMx_CR1** register, Counter enabled).

Figure 4.10-35: Example of counter operation in encoder interface mode (set DIR='0', before CEN=1)

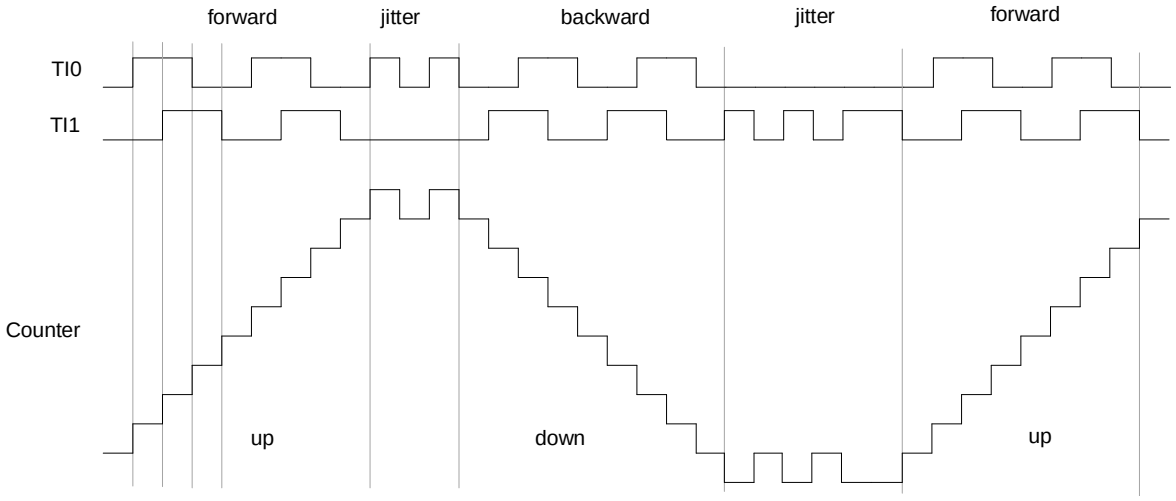
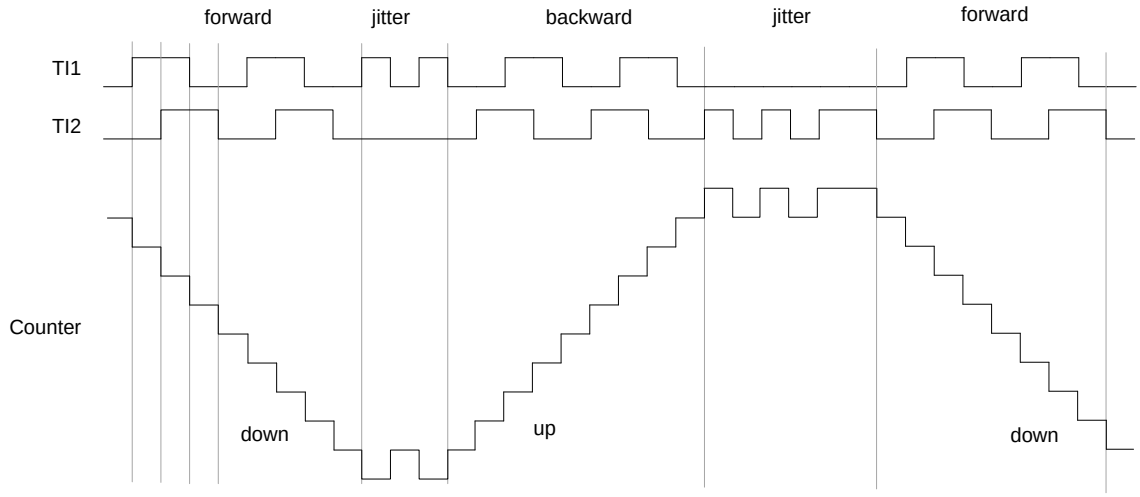


Figure 4.10-36: Example of encoder interface mode with (set DIR='1', before CEN=1)



The timer, when configured in Encoder Interface mode provides information on the sensor's current position. You can obtain dynamic information (speed, acceleration, and 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. You can do this 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).

4.10.2.14 TIMER INPUT XOR FUNCTION

The TI0S bit in the **TIMx_CR2** register, allows the input filter of channel 0 to be connected to the output of a XOR gate, combining the three input pins TIMx_CH0, TIMx_CH1 and TIMx_CH2.

The XOR output can be used with all the timer input functions such as trigger or input capture. An example of this feature used to interface Hall sensors is given in below section.

4.10.2.15 INTERFACING WITH HALL SENSORS

This is done using the advanced-control timers (TIM4/5) to generate PWM signals to drive the motor and another timer TIMx (TIM0/1/2/3) referred to as “interfacing timer” in Figure 4.10-37. The “interfacing timer” captures the 3 timer input pins (CC0, CC1, CC2) connected through a XOR to the TI0 input channel (selected by setting the TI0S bit in the **TIMx_CR2** register).

The slave mode controller is configured in reset mode; the slave input is TI0F_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.

On the “interfacing timer”, capture/compare channel 0 is configured in capture mode, capture signal is TRC. The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

The “interfacing timer” can select compare pulse which changes the configuration of the channels of the advanced-control timer (TIM4/5) (by triggering a COM event). The TIM0 timer is used to output compare pulse to drive the motor. To do this, the interfacing timer channel must be programmed so that a positive pulse is generated after a programmed delay (in output compare or PWM mode). This pulse is sent to the advanced-control timer (TIM4/5) through the TRGO output.

Example: you want to change the PWM configuration of your advanced-control timer TIM4 after a programmed delay each time a change occurs on the Hall inputs connected to one of the TIMx timers.

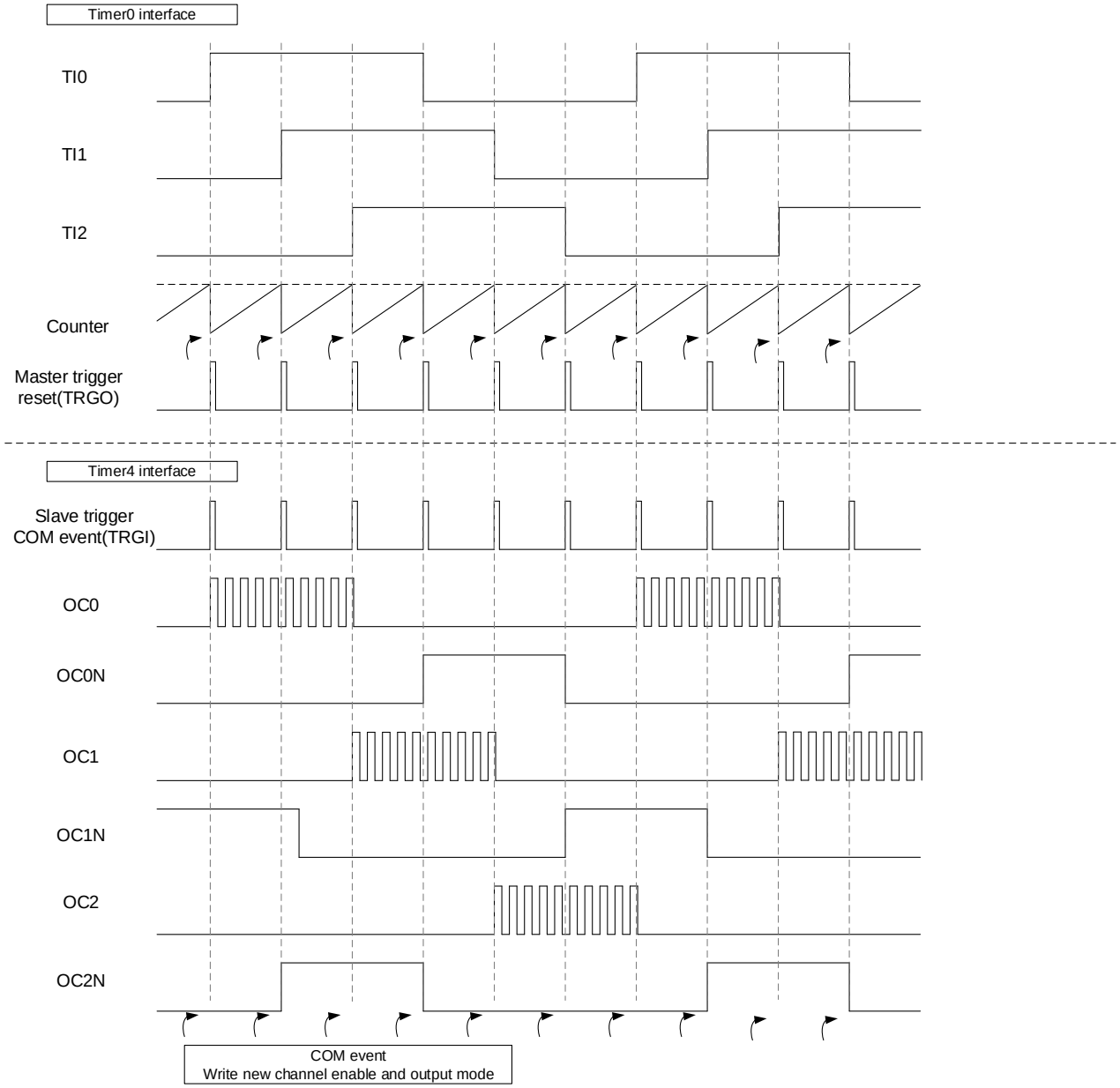
Master timer setting:

- Configure 3 timer inputs XORed to the TI0 input channel by writing the TI0S bit in the **TIMx_CR2** register to ‘1’
- Program the time base: write the **TIMx_ARR** to the max value (the counter must be cleared by the TI0 change. Set the prescaler to get a maximum counter period longer than the time between 2 changes on the sensors
- Configure the counter to be in the slave reset mode: write the SMS bit in the **TIMx_SMCR** register to 100
- Configure TI0FP0 as slave trigger: write the TS bit in the **TIMx_SMCR** register to 0111
- Configure TI0 compare pulse as master trigger(TRGO): write the MMS bit in the **TIMx_CR2** register to 011
- Slave timer setting:
- Program slave trigger (TRGI) select timer0: write the TS bits in the **TIMx_SMCR** register to ‘0000.
- Program channel 0 in PWM 1 mode with the desired delay: write the OC0M bits to ‘110’ and the CC0S bits to ‘00’ in the **TIMx_CCMR**1 register,
- Enable the CCxE, CCxNE and OCxM preload function: write the CCPC bit in the **TIMx_CR2** register to 1

Select the preload function are updated by setting the COMG bit or when an rising edge occurs on TRGI: write the CCUS bit in the **TIMx_CR2** register to 1.

In the advanced-control timer TIM4, the right ITR input must be selected as trigger input, the timer is programmed to generate PWM signals, the capture/compare control signals are preloaded (CCPC=1 in the **TIMx_CR2** register) and the COM event is controlled by the trigger input (CCUS=1 in the **TIMx_CR2** register). The PWM control bits (CCxE, OCxM) are written after a COM event for the next step (this can be done in an interrupt subroutine generated by the rising edge of TRGI).

Figure 4.10-37: Example of hall sensor interface



4.10.2.16 TIMER AND EXTERNAL TRIGGER SYNCHRONIZATION

The TIMx timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

SLAVE MODE: RESET MODE

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the UDIS bit in the TIMx_CR1 register is low, an update event UEV is generatedThen all the preloaded registers (**TIMx_ARR**, **TIMx_CCRx**) are updated.

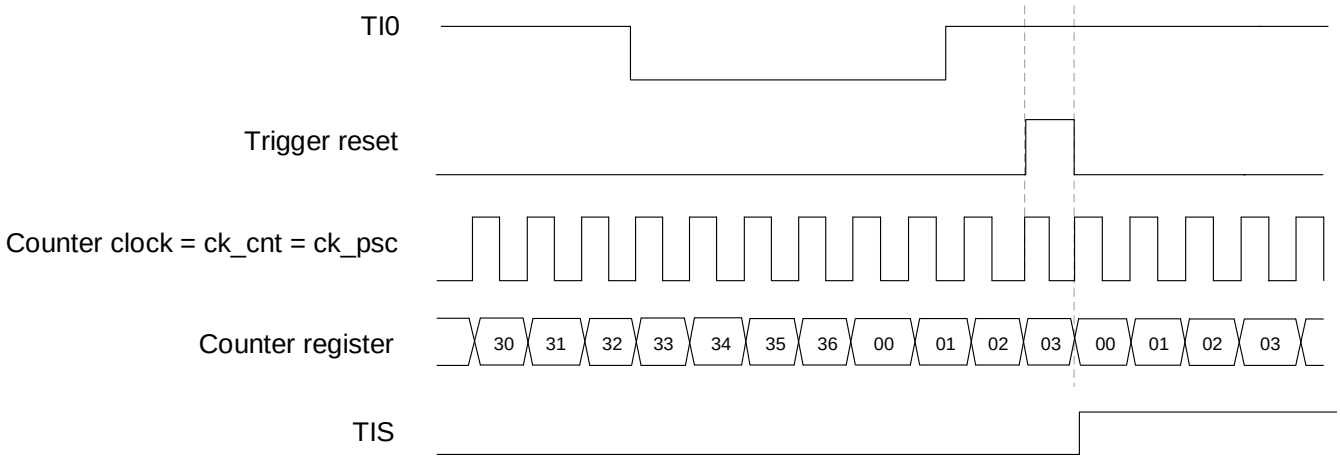
In the following example, the upcounter is cleared in response to a rising edge on TI0 input:

- Configure the channel 0 to detect rising edges on TI0. Configure the input filter duration (in this example, we don't need any filter, so we keep IC0F=0000). The capture prescaler is not used for triggering, so you don't need to configure it. The CC0S bits select the input capture source only, CC0S = 01 in the **TIMx_CCMR** register. Write CC0P=0 and CC0NP='0' in **TIMx_CCER** register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=100 in **TIMx_SMCR** register. Select TI0 as the input source by writing TS=0111 in **TIMx_SMCR** register.
- Start the counter by writing CEN=1 in the **TIMx_CR1** register

The counter starts counting on the internal clock, then behaves normally until TI0 rising edge. When TI0 rises, the counter is cleared and restarts from 0. In the meantime, the trigger status is set (TIS bit in the TIMx_ISC register) if enabled (depending on the TIE bits in TIMx_IER register).

Figure 10-38 shows this behavior when the auto-reload register **TIMx_ARR**=0x36. The delay between the rising edge on TI0 and the actual reset of the counter is due to the resynchronization circuit on TI0 input.

Figure 4.10-38: Control circuit in reset mode



SLAVE MODE: GATED MODE

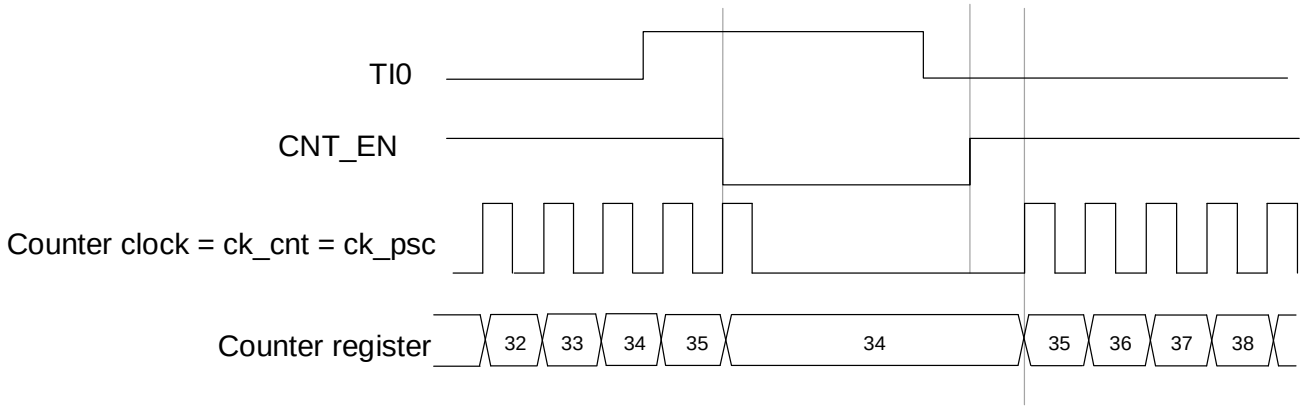
The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI0 input is low:

- Configure the channel 0 to detect low levels on TI0. Configure the input filter duration (in this example, we don't need any filter, so we keep IC0F=0000). The capture prescaler is not used for triggering, so you don't need to configure it. The CC0S bits select the input capture source only, CC0S=01 in **TIMx_CCMR** register. Write CC0P=1 and CC0NP='0' in **TIMx_CCER** register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS=101 in **TIMx_SMCR** register. Select TI0 as the input source by writing TS=0111 in **TIMx_SMCR** register.
- Enable the counter by writing CEN=1 in the **TIMx_CR1** register (in gated mode, the counter doesn't start if CEN=0, whatever is the trigger input level).

The counter starts counting on the internal clock as long as TI0 is low and stops as soon as TI0 becomes high. The TIS status in the TIMx_ISC register is not set in this mode.
The delay between the rising edge on TI0 and the actual stop of the counter is due to the resynchronization circuit on TI0 input.

Figure 4.10-39: Control circuit in gated mode



SLAVE MODE: TRIGGER MODE

The counter can start in response to an event on a selected input.

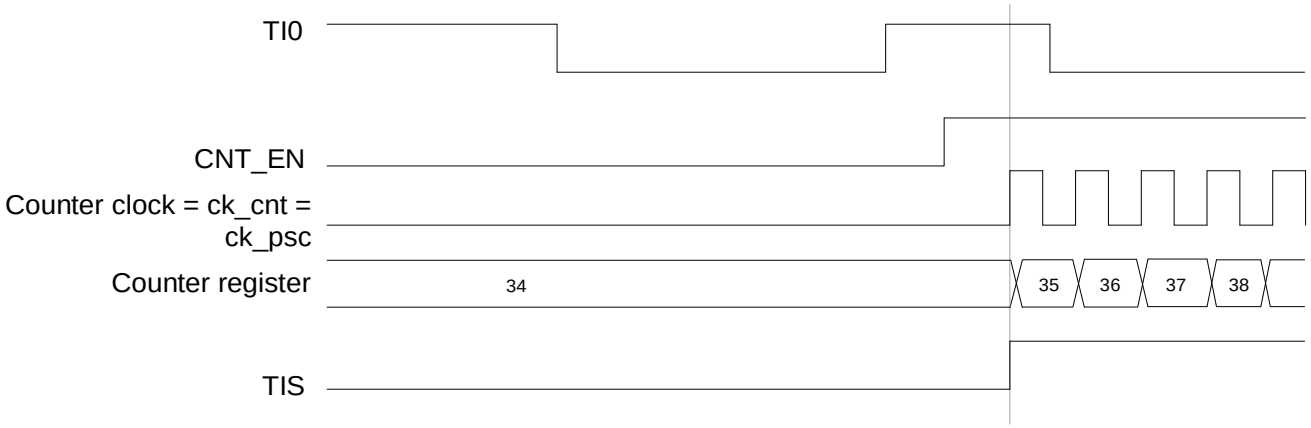
In the following example, the upcounter starts in response to a rising edge on TI0 input:

- Configure the channel 0 to detect rising edges on TI0. Configure the input filter duration (in this example, we don't need any filter, so we keep IC0F=0000). The capture prescaler is not used for triggering, so you don't need to configure it. The CC0S bits are configured to select the input capture source only, CC0S=01 in **TIMx_CCMR** register. Write CC0P=0 and CC0NP=0 in **TIMx_CCER** register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in **TIMx_SMCR** register. Select TI0 as the input source by writing TS=0111 in **TIMx_SMCR** register.
- Enable the counter by writing CEN=1 in the **TIMx_CR1** register (in trigger mode, the counter doesn't start if CEN=0, whatever is the trigger input level)

When a rising edge occurs on TI0, the counter starts counting on the internal clock and the TIS status is set.

The delay between the rising edge on TI0 and the actual start of the counter is due to the resynchronization circuit on TI0 input.

Figure 4.10-40: Control circuit in trigger mode



4.10.2.17 TIMER SYNCHRONIZATION

The TIM timers are linked together internally for timer synchronization or chaining. Refer to Advance timer for details.

4.10.2.18 DEBUG MODE

The TIMx timers are linked together internally for timer synchronization or chaining. Refer to Advance timer for details.

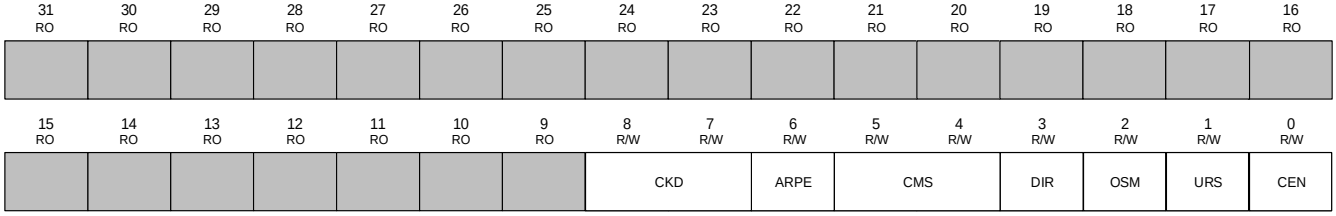
4.10.3 TIMER REGISTER MAP

Base address:

- Timer 0: 0x4000_0000
- Timer 1: 0x4000_0100
- Timer 2: 0x4000_0200
- Timer 3: 0x4000_0300

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	TIMx_CR1	R/W	0x0000_0000	TIMx Control Register 1	212
0x0004	TIMx_CR2	R/W	0x0000_0000	TIMx Control Register 2	214
0x0008	TIMx_SMCR	R/W	0x0000_0000	TIMx Slave Mode Control Register	216
0x000C	TIMx_EGR	WO	0x0000_0000	TIMx Event Generation Register	219
0x0010	TIMx_IER	WO	0x0000_0000	TIMx Interrupt Enable Register	220
0x0014	TIMx_IDR	WO	0x0000_0000	TIMx Interrupt Disable Register	221
0x0018	TIMx_IMR	RO	0x0000_0000	TIMx Interrupt Mask Register	222
0x001C	TIMx_RIS	RO	0x0000_0000	TIMx Raw Interrupt Status	223
0x0020	TIMx_ISC	R/W1C	0x0000_0000	TIMxInterrupt Status and Clear	224
0x0028	TIMx_CCMR	R/W	0x0000_0000	TIMx Capture/Compare Mode Register	225
0x002C	TIMx_CCER	R/W	0x0000_0000	TIMx Capture/Compare Enable Register	229
0x0030	TIMx_PSC	R/W	0x0000_0000	TIMx Prescaler	232
0x0034	TIMx_ARR	R/W	0x0000_0000	TIMx Auto-Reload Register	232
0x0038	TIMx_CNTR	R/W	0x0000_0000	TIMx Counter Register	233
0x003C	TIMx_CCR0	R/W	0x0000_0000	TIMx Capture/Compare Register 0	233
0x0040	TIMx_CCR1	R/W	0x0000_0000	TIMx Capture/Compare Register 1	234
0x0044	TIMx_CCR2	R/W	0x0000_0000	TIMx Capture/Compare Register 2	234
0x004C	TIMx_ADCTS	R/W	0x0000_0000	TIMx ADC Trigger Select	235
0x0050	TIMx_BDTR	R/W	0x0000_0000	TIMx Break and Dead-Time Register	236

4.10.3.1 TIMX_CR1 - TIM 0 / 1 / 2 / 3 CONTROL REGISTER 1

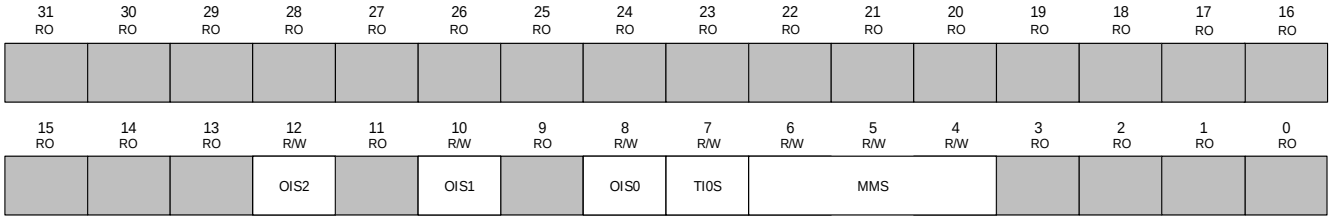


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:9	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
8:7	CKD	R/W	0x0	Clock Division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock (t_{DTS})used by the dead-time generators and the digital filters (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 program this value
6	ARPE	R/W	0	Auto-Reload Preload Enable 0: **TIMx_ARR** register is not buffered 1: **TIMx_ARR** register is buffered
5:4	CMS	R/W	0x0	Center-Aligned Mode Select 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt status of channels configured in output (CCxS=00 in **TIMx_CCMR** register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt status of channels configured in output (CCxS=00 in **TIMx_CCMR** register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt status of channels configured in output (CCxS=00 in **TIMx_CCMR** register) are set both when the counter is counting up or down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1).
3	DIR	R/W	0	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
2	OSM	R/W	0	One Shot Mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the CEN bit)

Bits	Name	Type	Reset	Description
1	UDIS	R/W	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).
0	CEN	R/W	0	Counter Enable 0: Counter disabled 1: Counter enabled 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.

4.10.3.2 TIMX_CR2 - TIM 0 / 1 / 2 / 3 CONTROL REGISTER 2

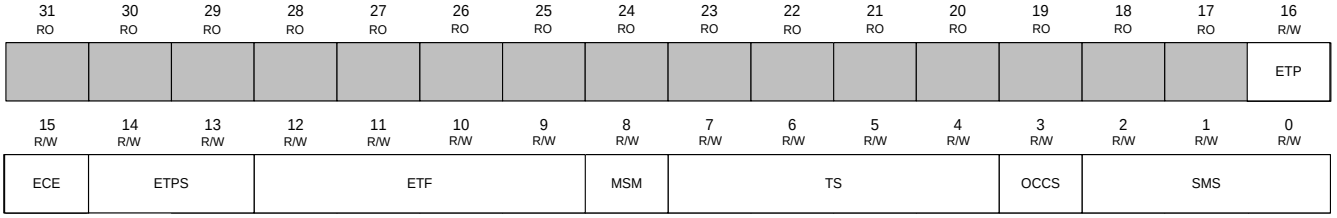


Offset: 0x0004

Bits	Name	Type	Reset	Description
31: 13	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
12	OIS2	R/W	0	Output Idle state 2 (OC2 output) Refer to OIS0 bit
11	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10	OIS1	R/W	0	Output Idle state 1 (OC1 output) Refer to OIS0 bit
9	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
8	OIS0	R/W	0	Output Idle state 0 (OC0 output) 0: OC0=0 (after a dead-time if OC0N is implemented) when MOE=0 1: OC0=1 (after a dead-time if OC0N is implemented) when MOE=0 Note: This bit cannot be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in **TIMx_BDTR** register).
7	TI0S	R/W	0	TI0 Selection 0: The TIMx_CH0 pin is connected to TI0 input 1: The TIMx_CH0, CH1 and CH2 pins are connected to the TI0 input (XOR combination)
6:4	MMS	R/W	0	Master mode selection(TRGO) These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the **TIMx_EGR** register is used as 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. 001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in **TIMx_SMCR** register). 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output (TRGO) send a positive

Bits	Name	Type	Reset	Description
				pulse when the CC0I flag is to be set (even if it was already high), as soon as a capture 0 or a compare 0 match counter occurred. (TRGO). 100: Compare - OC0REF signal is used as trigger output (TRGO) 101: Compare - OC1REF signal is used as trigger output (TRGO) 110: Compare - OC2REF signal is used as trigger output (TRGO)
3:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.10.3.3 TIMX_SMCR - TIM 0 / 1 / 2 / 3 SLAVE MODE CONTROL REGISTER



Offset: 0x0008

Bits	Name	Type	Reset	Description
31: 17	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	ETP	R/W	0	External Trigger Polarity This bit selects whether ETR or $\overline{ETR}$ is used for trigger operations 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.
15	ECE	R/W	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. Note: 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=1001). 2: It is possible to simultaneously use external clock mode 2 with the following slave modes: reset mode, gated mode and trigger mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 1001). 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
14:13	ETPS	R/W	0x0	External Trigger Prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8
12:9	ETF	R/W	0x0	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. The digital filter is made of an event counter in which N 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

Bits	Name	Type	Reset	Description
				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: $f_{SAMPLING} = f_{DTS} / 16$, 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 Note: Care must be taken that f_{DTS} is replaced in the formula by CK_INT when ETF[3:0] = 1, 2 or 3.
8	MSM	R/W	0	Master/Slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
7:4	TS	R/W	0	Trigger Selection This bit-field selects the trigger input to be used to synchronize the counter. 0000: Internal Trigger 0 (ITR0) 0001: Internal Trigger 1 (ITR1) 0010: Internal Trigger 2 (ITR2) 0011: Internal Trigger 3 (ITR3) 0100: Internal Trigger 4 (ITR4) 0101: Internal Trigger 5 (ITR5) 0110: TI0 Edge Detector (TI0F_ED) 0111: Filtered Timer Input 0 (TI0FP0) 1000: Filtered Timer Input 1 (TI1FP1) 1001: External Trigger input (ETRF) Note: These bits must be changed only when they are not used (e.g. when SMS=0000) to avoid wrong edge detections at the transition.

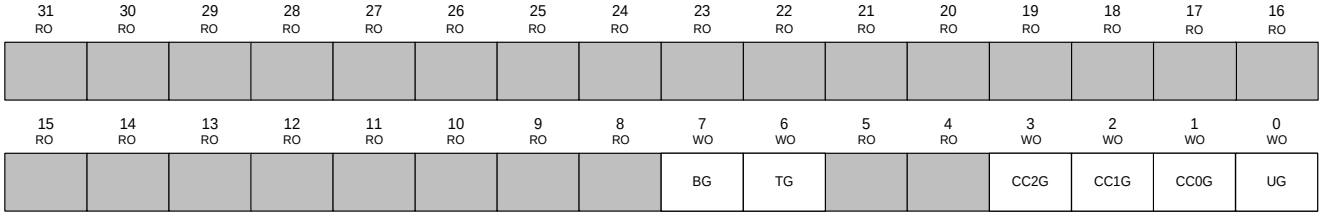
Table 4.10-2: TIMx Internal trigger connection

ITR0(TS = 0000)	ITR1(TS = 0001)	ITR2(TS = 0010)	ITR3(TS = 0011)	ITR4(TS = 0100)	ITR5(TS = 0101)
TIM0	TIM1	TIM2	TIM3	TIM4	TIM5

3	OCCS	R/W	0	OCREF Clear Selection This bit is used to select the OCREF clear source. 0: OCREF_CLR_INT is connected to ETRF 1: OCREF_CLR_INT is connected to COMP_TRIG
2:0	SMS	R/W	0x0	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).

Bits	Name	Type	Reset	Description
				000: Slave mode disabled - if CEN = '1' then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI0FP0 edge depending on TI1FP1 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI0FP0 level. 011: Encoder mode 3 - Counter counts up/down on both TI0FP0 and TI1FP1 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 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 at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI0F_ED is selected as the trigger input (TS='0110'). Indeed, TI0F_ED outputs 1 pulse for each transition on TI0F, whereas the gated mode checks the level of the trigger signal.

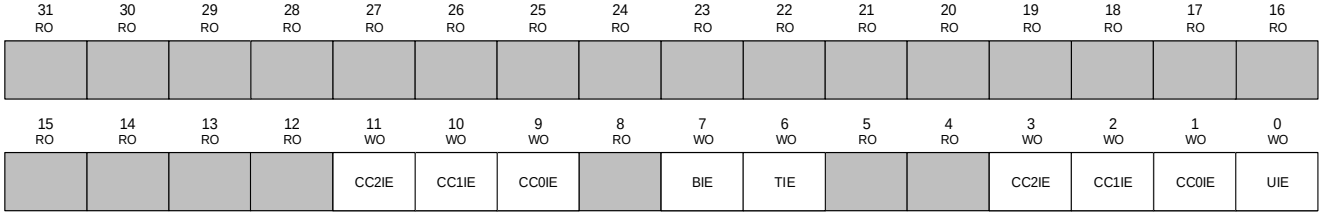
4.10.3.4 TIMX_EGR - TIM 0 / 1 / 2 / 3 EVENT GENERATION REGISTER



Offset: 0x000C

Bits	Name	Type	Reset	Description
31:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BG	WO	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 BIS status is set. Related interrupt can occur if enabled.
6	TG	WO	0	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware except slave gated mode. 0: No action 1: The TIS status is set in TIMx_ISC register. Related interrupt can occur if enabled.
5:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	CC2G	WO	0	Capture/Compare 2 generation Refer to CC0G description
2	CC1G	WO	0	Capture/Compare 1 generation Refer to CC0G description
1	CC0G	WO	0	Capture/Compare 0 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 0: **If channel CC0 is configured as output:** Corresponding interrupt is sent if enabled. **If channel CC0 is configured as input:** The current value of the counter is captured in **TIMx_CCR0** register. The corresponding interrupt is sent if enabled.
0	UG	WO	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the auto-reload value (**TIMx_ARR**) if DIR=1 (downcounting).

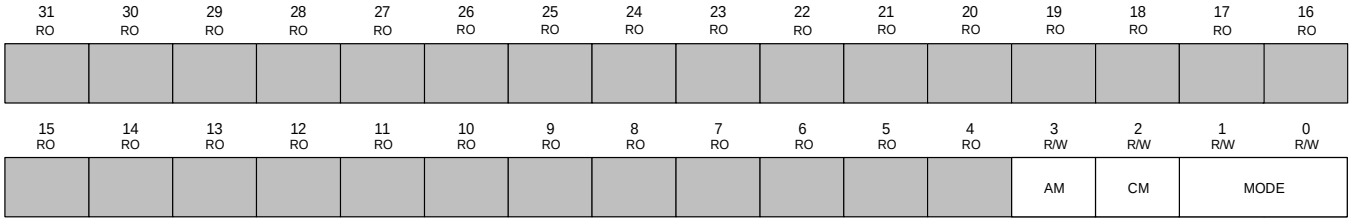
4.10.3.5 TIMX_IER - TIM 0 / 1 / 2 / 3 INTERRUPT ENABLE REGISTER



Offset: 0x0010

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2OIE	WO	0	Capture/Compare 2 overcapture Interrupt Enable 1: CC2O interrupt is enabled
10	CC1OIE	WO	0	Capture/Compare 1 overcapture Interrupt Enable 1: CC1O interrupt is enabled
9	CC0OIE	WO	0	Capture/Compare 0 overcapture Interrupt Enable 1: CC0O interrupt is enabled
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BIE	WO	0	Break Interrupt Enable 1: Break interrupt is enabled
6	TIE	WO	0	Trigger interrupt enable 1: Trigger interrupt is enabled
5:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	CC2IE	WO	0	Capture/Compare 2 Interrupt Enable 1: CC2 interrupt enabled
2	CC1IE	WO	0	Capture/Compare 1 Interrupt Enable 1: CC1 interrupt enabled
1	CC0IE	WO	0	Capture/Compare 0 Interrupt Enable 1: CC0 interrupt enabled
0	UIE	WO	0	Update Interrupt Enable 1: Update interrupt enabled

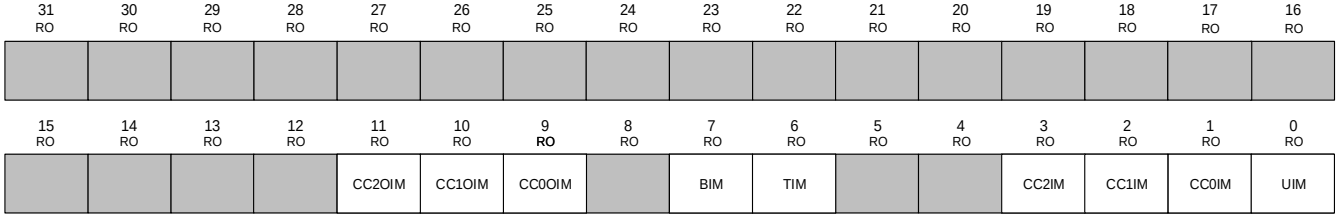
4.10.3.6 TIMX_IDR - TIM 0 / 1 / 2 / 3 INTERRUPT DISABLE REGISTER



Offset: 0x0014

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2OID	WO	0	Capture/Compare 2 overcapture Interrupt Disable 1: CC2O interrupt is disable
10	CC1OID	WO	0	Capture/Compare 1 overcapture Interrupt Disable 1: CC1O interrupt is disable
9	CC0OID	WO	0	Capture/Compare 0 overcapture Interrupt Disable 1: CC0O interrupt is disable
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BID	WO	0	Break Interrupt Disable 1: Break interrupt is disabled
6	TID	WO	0	Trigger Interrupt Disable 1: Trigger interrupt is disabled
5:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	CC2ID	WO	0	Capture/Compare 2 Interrupt Disable 1: CC2 interrupt is disabled
2	CC1ID	WO	0	Capture/Compare 1 Interrupt Disable 1: CC1 interrupt is disabled
1	CC0ID	WO	0	Capture/Compare 0 Interrupt Disable 1: CC0 interrupt is disabled
0	UID	WO	0	Update Interrupt Disable 1: Update interrupt is disabled

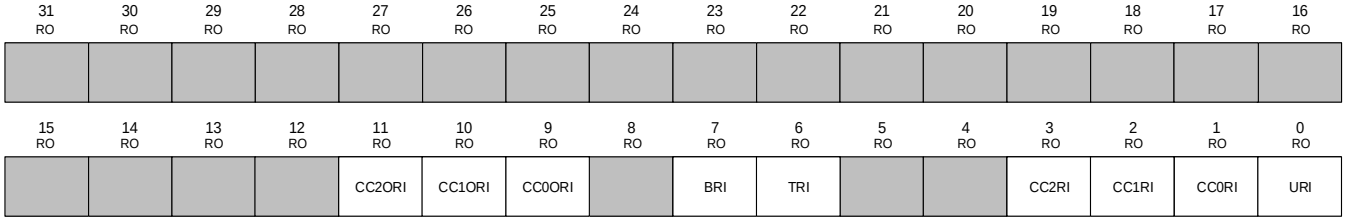
4.10.3.7 TIMX_IMR - TIM 0 / 1 / 2 / 3 INTERRUPT MASK REGISTER



Offset: 0x0018

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2OIM	RO	0	Capture/Compare 2 overcapture Interrupt Mask Status 0: CC2O interrupt is disable 1: CC2O interrupt is enable
10	CC1OIM	RO	0	Capture/Compare 1 overcapture Interrupt Mask Status 0: CC1O overcapture interrupt is disable 1: CC1O overcapture interrupt is enable
9	CC0OIM	RO	0	Capture/Compare 0 overcapture Interrupt Mask Status 0: CC0O interrupt is disable 1: CC0O interrupt is enable
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BIM	RO	0	Break Interrupt Mask Status 0: Break interrupt is disable 1: Break interrupt is enabled
5:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	CC2IM	RO	0	Capture/Compare 2 Interrupt Mask Status 0: CC2 interrupt is disable 1: CC2 interrupt is enabled
2	CC1IM	RO	0	Capture/Compare 1 Interrupt Mask Status 0: CC1 interrupt is disable 1: CC1 interrupt is enabled
1	CC0IM	RO	0	Capture/Compare 0 Interrupt Mask Status 0: CC0 interrupt is disable 1: CC0 interrupt is enabled.
0	UIM	RO	0	Update Interrupt Mask Status 0: Update interrupt is disable 1: Update interrupt is enabled

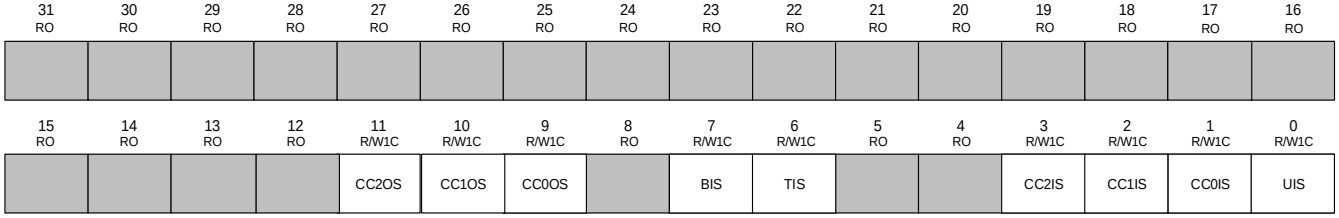
4.10.3.8 TIMX_ RIS - TIM 0 / 1 / 2 / 3 RAW INTERRUPT STATUS



Offset: 0x001C

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2ORI	RO	0	Capture/Compare 2 overcapture Raw Interrupt Status 0: No interrupt 1: CC2O interrupt is asserting.
10	CC1ORI	RO	0	Capture/Compare 1 overcapture Raw Interrupt Status 0: No interrupt 1: CC1O interrupt is asserting.
9	CC0ORI	RO	0	Capture/Compare 0 overcapture Raw Interrupt Status 0: No interrupt 1: CC0O interrupt is asserting.
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BRI	RO	0	Break Raw Interrupt Status 0: No interrupt 1: Break interrupt is asserting.
6	TRI	RO	0	Trigger Raw Interrupt Status 0: No interrupt 1: Trigger interrupt is asserting.
5:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	CC2RI	RO	0	Capture/Compare 2 Raw Interrupt Status 0: No interrupt 1: CC2 interrupt is asserting.
2	CC1RI	RO	0	Capture/Compare 1 Raw Interrupt Status 0: No interrupt 1: CC1 interrupt is asserting
1	CC0RI	RO	0	Capture/Compare 0 Raw Interrupt Status 0: No interrupt 1: CC0 interrupt is asserting
0	URI	RO	0	Update Raw Interrupt Status 0: No interrupt 1: Update interrupt is asserting

4.10.3.9 TIMX_ISC - TIM 0 / 1 / 2 / 3 INTERRUPT STATUS AND CLEAR



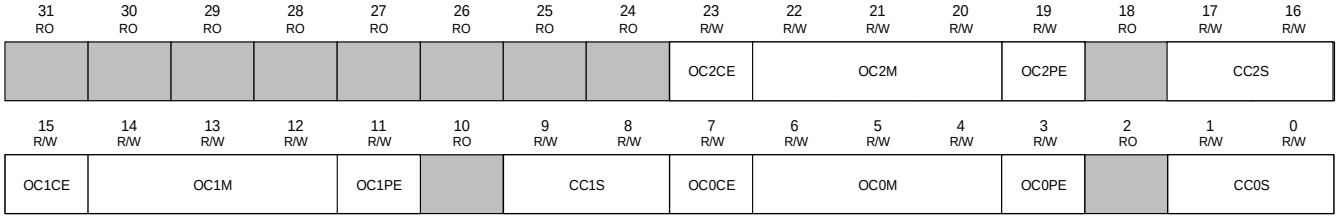
Offset: 0x0020

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2OS	R/W1C	0	Capture/Compare 2 overcapture Status and clear Refer to CC0OF
10	CC1OS	R/W1C	0	Capture/Compare 1 overcapture Status and clear Refer to CC0OF
9	CC0OS	R/W1C	0	Capture/Compare 0 overcapture Status and clear This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '1'. 0: No overcapture has been detected. 1: The counter value has been captured in **TIMx_CCR0** register while CC0RI in the TIMx_RIS was already set
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BIS	R/W1C	0	Break Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: Break interrupt has been signaled.
6	TIS	R/W1C	0	Trigger Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: Trigger interrupt has been signaled.
5:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	CC2IS	R/W1C	0	Capture/Compare 2 Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: CC2 interrupt has been signaled.
2	CC1IS	R/W1C	0	Capture/Compare 1 Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: CC1 interrupt has been signaled.
1	CC0IS	R/W1C	0	Capture/Compare 0 Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: CC0 interrupt has been signaled.
0	UIS	RO	0	Update Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: Update interrupt has been signaled.

4.10.3.10 TIMX_CCMR - TIM 0 / 1 / 2 / 3 CAPTURE/COMPARE MODE REGISTER

The channels can be used in input (capture mode) or in 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 in input and in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So you must take care that the same bit can have a different meaning for the input stage and for the output stage.

OUTPUT COMPARE MODE

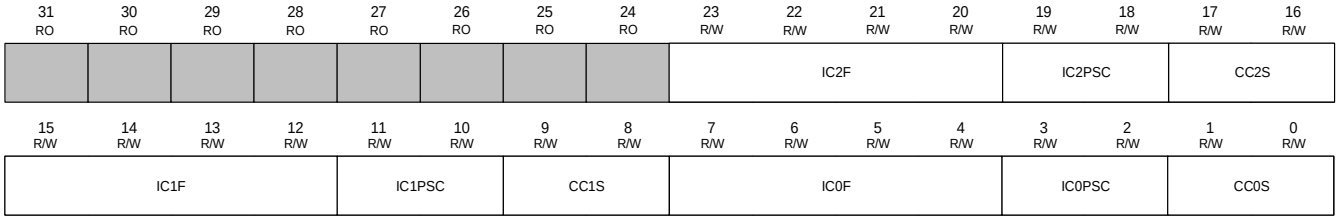


Offset: 0x0028

Bits	Name	Type	Reset	Description
31:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23	OC2CE	R/W	0	Output Compare 2 clear enable Refer to OC0CE
22:20	OC2M	R/W	0x0	Output Compare 2 mode Refer to OC0M
19	OC2PE	R/W	0	Output Compare 2 preload enable Refer to OCPE
18	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
17:16	CC2S	R/W	0x0	Capture/Compare 2 selection Refer to CC0S
15	OC1CE	R/W	0	Output Compare 1 clear enable Refer to OC0CE
14:12	OC1M	R/W	0x0	Output Compare 1 mode Refer to OC0M
11	OC1PE	R/W	0	Output Compare 1 preload enable. Refer to OCPE
10	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9:8	CC1S	R/W	0	Capture/Compare 1 selection Refer to CC0S
7	OC0CE	R/W	0	Output Compare 0 Clear Enable 0: OC0 is not affected by the OCREF_CLR_INPUT 1: OC0 is cleared as soon as a high level is detected on OCREF_CLR_INPUT
6:4	OC0M	R/W	0	Output Compare 0 mode 001: Set channel 0 to active level on match. OC0 signal is forced high when the counter **TIMx_CNTR** matches the capture/compare register 0 (**TIMx_CCR0**). 010: Set channel 0 to inactive level on match. OC0 signal is forced low

Bits	Name	Type	Reset	Description
				when the counter **TIMx_CNTR** matches the capture/compare register 0 (**TIMx_CCR0**). 011: Toggle - OC0 toggles when **TIMx_CNTR** matches **TIMx_CCR0**. 100: Force inactive level - OC0 is forced low. 101: Force active level - OC0 is forced high. 110: PWM mode 1 - In upcounting, channel 0 is active as long as **TIMx_CNTR** < **TIMx_CCR1** else inactive. In downcounting, channel 0 is inactive as long as **TIMx_CNTR** > **TIMx_CCR0** else active. 111: PWM mode 2 - In upcounting, channel 0 is inactive as long as **TIMx_CNTR** < **TIMx_CCR0** else active. In downcounting, channel 0 is active as long as **TIMx_CNTR** > **TIMx_CCR0** else inactive. Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in **TIMx_BDTR** register) and CC0S='00' (the channel is configured in output). 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from "frozen" mode to "PWM" mode. 3: On channels having a complementary output, this bit field is preloaded. If the CCPC bit is set in the **TIMx_CR2** register then the OC0M active bits take the new value from the preloaded bits only when a COM event is generated.
3	OC0PE	R/W	0	Output Compare 0 preload enable 0: Preload register on **TIMx_CCR0** disabled. **TIMx_CCR0** can be written at anytime, the new value is taken in account immediately. 1: Preload register on **TIMx_CCR0** enabled. Read/Write operations access the preload register. **TIMx_CCR0** preload value is loaded in the active register at each update event. Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in **TIMx_BDTR** register) and CC0S='00' (the channel is configured in output). 2: The PWM mode can be used without validating the preload register only in one shot mode (PSM bit set in **TIMx_CR1** register). Else the behavior is not guaranteed.
2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1:0	CC0S	R/W	0	Capture/Compare 0 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC0 channel is configured as output Note: CC0S bits are writable only when the channel is OFF (CC0E = '0' in **TIMx_CCER**).

INPUT CAPTURE MODE



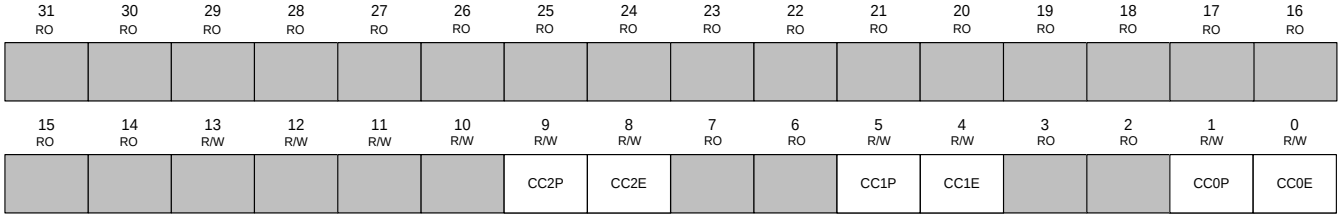
Offset: 0x0028

Bits	Name	Type	Reset	Description
31:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:20	IC2F	R/W	0x0	Input Capture 2 filter Refer to IC0F
19:18	IC2PSC	R/W	0x0	Input Capture 2 prescaler Refer to IC0PSC
17:16	CC2S	R/W	0x0	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: reserved 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (**TIMx_SMCR** register) Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in **TIMx_CCER**).
15:12	IC1F	R/W	0	Input Capture 1 Filter Refer to IC0F
11:10	IC1PSC	R/W	0x0	Input Capture 1 prescaler Refer to IC0PSC
9:8	CC1S	R/W	0x0	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 TI0 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 TS bit (**TIMx_SMCR** register) Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in **TIMx_CCER**).
7:4	IC0F	R/W	0	Input Capture 0 Filter This bit-field defines the frequency used to sample TI0 input and the length of the digital filter applied to TI0. The digital filter is made of an event counter in which N events are needed to validate a transition on the output: 000: No filter, sampling is done at f_{DTS} 0001: $f_{SAMPLING} = f_{CK_INT}$, N = 2

Bits	Name	Type	Reset	Description
				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: $f_{SAMPLING} = f_{DTS}$ / 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 Note: Care must be taken that f_{DTS} is replaced in the formula by CK_INT when ICxF[3:0] = 1, 2 or 3.
3:2	IC0PSC	R/W	0x0	Input Capture 0 prescaler This bit-field defines the ratio of the prescaler acting on CC0 input (IC0). The prescaler is reset as soon as CEN='0' (**TIMx_CR1** 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	CC0S	R/W	0x0	Capture/Compare 0 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC0 channel is configured as output 01: CC0 channel is configured as input, IC0 is mapped on TI0 10: CC0 channel is configured as input, IC0 is mapped on TI1 11: CC0 channel is configured as input, IC0 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (**TIMx_SMCR** register) Note: CC0S bits are writable only when the channel is OFF (CC0E = '0' in **TIMx_CCER**).

4.10.3.11 TIMX_CCER - TIM 0 / 1 / 2 / 3 CAPTURE/COMPARE ENABLE REGISTER

OUTPUT MODE



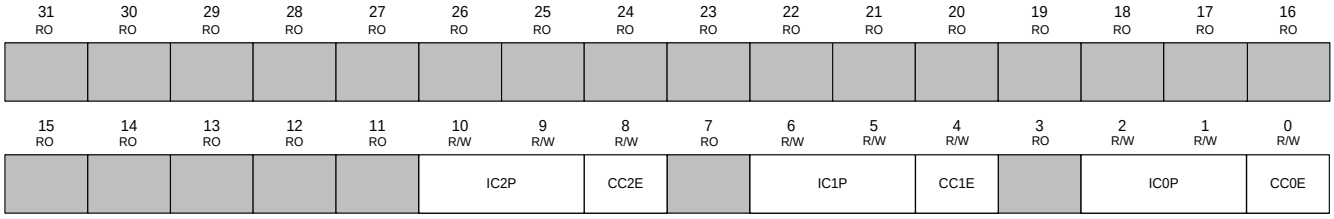
Offset: 0x002C

Bits	Name	Type	Reset	Description
31:10	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9	CC2P	R/W	0	Capture/Compare 2 output polarity Refer to CC0P
8	CC2E	R/W	0	Capture/Compare 2 output enable Refer to CC0E
7:6	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	CC1P	R/W	0	Capture/Compare 1 output polarity Refer to CC0P
4	CC1E	R/W	0	Capture/Compare 1 output enable Refer to CC0E
3:2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	CC0P	R/W	0	Capture/Compare 0 output polarity 0: OC0 active high 1: OC0 active low Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the **TIMx_CR2** register then the CC0P active bit takes the new value from the preloaded bits only when a Commutation event is generated. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in **TIMx_BDTR** register).
0	CC0E	R/W	0	Capture/Compare 0 output enable 0: Off – OC0 is not active. OC0 level is then function of MOE, OSSI, OSSR, OIS0, OIS0N and CC0NE bits. 1: On – OC0 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS0, OIS0N and CC0NE bits. Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the **TIMx_CR2** register then the CC0E active bit takes the new value from the preloaded bits only when a Commutation event is generated.

Table 4-3: Output control bits for complementary OCx and OCxN channels with break feature

Control bits					Output States(1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx output state	OCxN output state
1	X	0	0	0	Output Disabled (not driven by the timer) OCx=0, OCx_EN=1	Output Disabled (not driven by the timer) OCxN=0, OCxN_EN=1
		0	0	1	Output Disabled (not driven by the timer) OCx=0, OCx_EN=1	OCxREF + Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=0
		0	1	0	OCxREF + Polarity OCx=OCxREF xor CCxP, OCx_EN=0	Output Disabled (not driven by the timer) OCxN=0, OCxN_EN=1
		0	1	1	OCREF + Polarity + dead-time OCx_EN=0	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=0
		1	0	0	Output Disabled (not driven by the timer) OCx=CCxP, OCx_EN=1	Output Disabled (not driven by the timer) OCxN=CCxNP, OCxN_EN=1
		1	0	1	Off-State (output enabled with inactive state) OCx=CCxP, OCx_EN=0	OCxREF + Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=0
		1	1	0	OCxREF + Polarity OCx=OCxREF xor CCxP, OCx_EN=0	Off-State (output enabled with inactive state) OCxN=CCxNP, OCxN_EN=0
		1	1	1	OCREF + Polarity + dead-time OCx_EN=0	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=0
0	0	X	0	0	Output Disabled (not driven by the timer) OCx=CCxP, OCx_EN=1	Output Disabled (not driven by the timer) OCxN=CCxNP, OCxN_EN=1
	0		0	1	Output Disabled (not driven by the timer) Asynchronously: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1. Then if the clock is present: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	
	0		1	0		
	0		1	1		
	1		0	0	Output Disabled (not driven by the timer) OCx=CCxP, OCx_EN=1	Output Disabled (not driven by the timer) OCxN=CCxNP, OCxN_EN=1
	1		0	1	Off-State (output enabled with inactive state) Asynchronously: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0 Then if the clock is present: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	

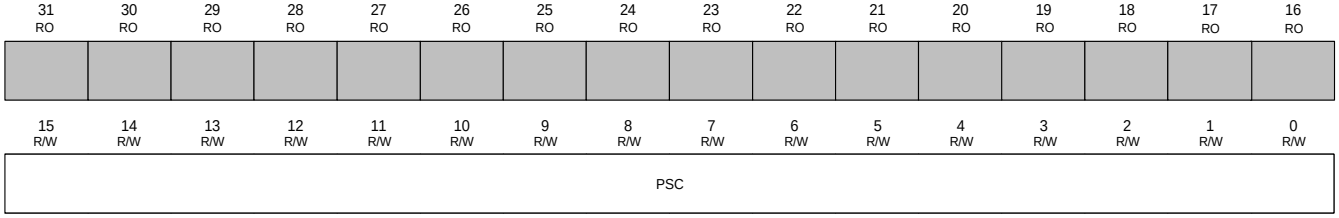
INPUT MODE



Offset: 0x0020

Bits	Name	Type	Reset	Description
31:11	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10:9	IC2P	R/W	0	Input Capture 2 input polarity Refer to IC0P
8	CC2E	R/W	0	Capture/Compare 2 input enable Refer to CC0E
7	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:5	IC1P	R/W	0	Input Capture 1 input polarity Refer to IC0P
4	CC1E	R/W	0	Capture/Compare 1 input enable Refer to CC0E
3	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:1	IC0P	R/W	0	Input Capture 0 input polarity IC0P bits select the active polarity of TI0FP0 and TI1FP0 for trigger or capture operations. 00: non-inverted/rising edge The circuit is sensitive to TIxFP0 rising edge (capture or trigger operations in reset, external clock or trigger mode), TIxFP0 is not inverted (trigger operation in gated mode or encoder mode). 01: inverted/falling edge The circuit is sensitive to TIxFP0 falling edge (capture or trigger operations in reset, external clock or trigger mode), TIxFP0 is inverted (trigger operation in gated mode or encoder mode). 10: reserved 11: non-inverted/both edges
0	CC0E	R/W	0	Capture/Compare 0 input enable 0: Input Capture is disabled 1: Input Capture is enabled

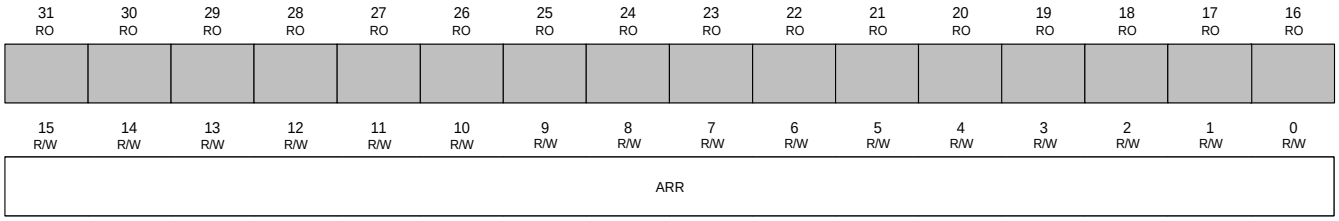
4.10.3.12 TIMX_PSC - TIM 0 / 1 / 2 / 3 PRESCALER



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	PSC	R/W	0x0	Prescaler valueThe counter clock frequency (CK_CNT) is equal to f_{CK_PSC} / (PSC [15:0] + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of **TIMx_EGR** register or through trigger controller when configured in “reset mode”).

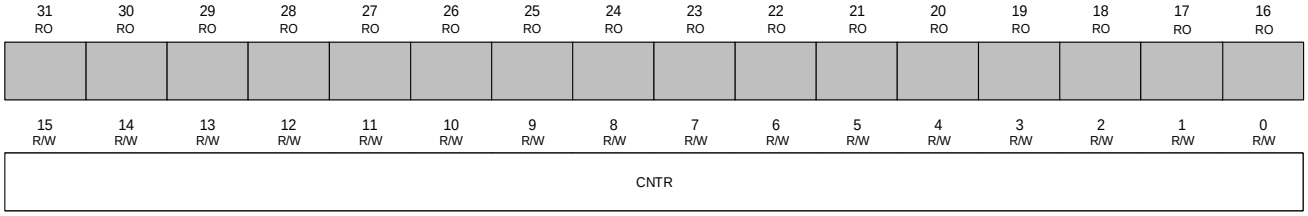
4.10.3.13 TIMX_ARR - TIM 0 / 1 / 2 / 3 AUTO-RELOAD REGISTER



Offset: 0x0024

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	ARR	R/W	0x0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to the Section 4.10.2.2 Timer-base unit for more details about ARR update and behavior The counter is blocked while the auto-reload value is null.

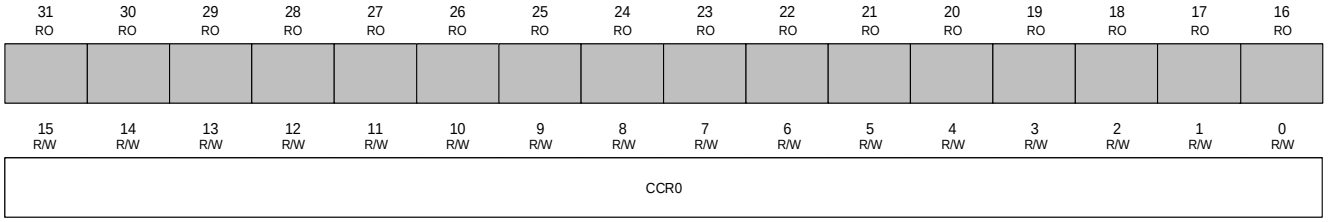
4.10.3.14 TIMX_CNTR - TIM 0 / 1 / 2 / 3 COUNTER



Offset: 0x0038

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	CNTR	R/W	0x0	Counter Register

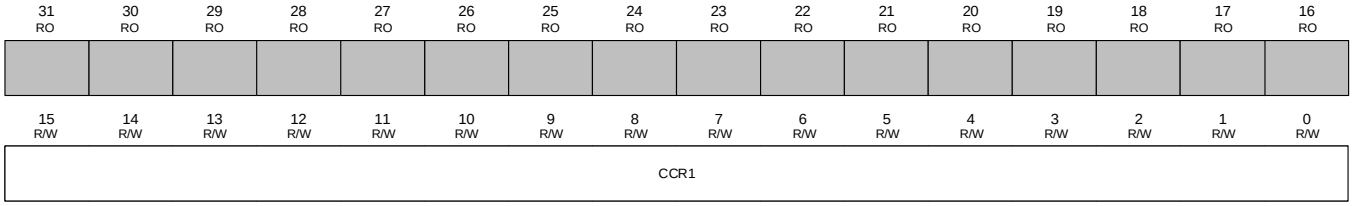
4.10.3.15 TIMX_CCR0 - TIM 0 / 1 / 2 / 3 CAPTURE/COMPARE REGISTER 0



Offset: 0x003C

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	CCR0	R/W	0x0	Capture/Compare Register 0 **If channel CC0 is configured as output:** CCR0 is the value to be loaded in the actual capture/compare 0 register (preload value). It is loaded permanently if the preload feature is not selected in the **TIMx_CCMR** register (bit OC0PE). Else the preload value is copied in the active capture/compare 0 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter **TIMx_CNTR** and signaled on OC0 output. **If channel CC0 is configured as input:** CCR0 is the counter value transferred by the last input capture 0 event (IC0)

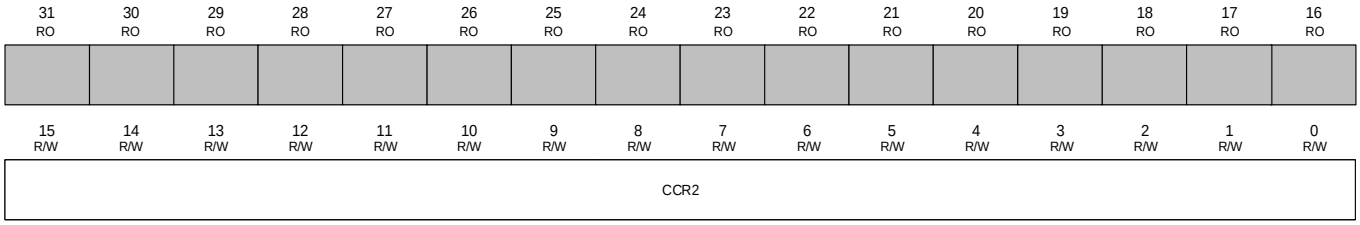
4.10.3.16 TIMX_CCR1 - IM 0 / 1 / 2 / 3 CAPTURE/COMPARE REGISTER 1



Offset: 0x0040

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	CCR1	R/W	0x0	Capture/Compare Register 1 Refer to CCR0 in the **TIMx_CCR0**

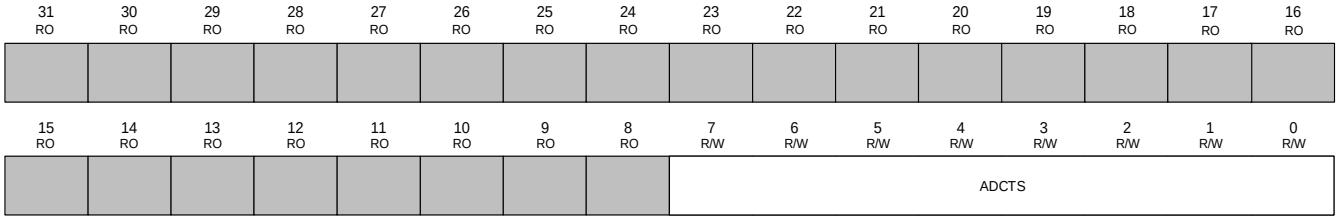
4.10.3.17 TIMX_CCR2 - TIM 0 / 1 / 2 / 3 CAPTURE/COMPARE REGISTER 2



Offset: 0x0044

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	CCR2	R/W	0x0	Capture/Compare Register 2 Refer to CCR0 in the **TIMx_CCR0**

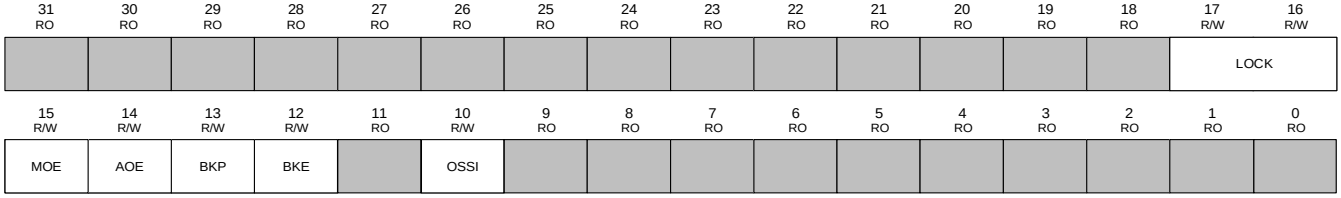
4.10.3.18 TIMX_ADCTS - TIM 0 / 1 / 2 / 3 ADC TRIGGER SELECT



Offset: 0x004C

Bits	Name	Type	Reset	Description
31:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	ADCTS	R/W	0	ADC Trigger Select Bit0 Select Update Event Raw Interrupt Bit1 Select CC0 Capture/Compare 0 Raw Interrupt Bit2 Select CC1 Capture/Compare 1 Raw Interrupt Bit3 Select CC2 Capture/Compare 2 Raw Interrupt Bit4 Reserved Bit5 Reserved Bit6 Select Slave Trigger Raw Interrupt Bit7 Select Break Event Raw Interrupt

4.10.3.19 TIMX_BDTR - TIM 0 / 1 / 2 / 3 BREAK AND DEAD-TIME REGISTER



Offset: 0x0050

Note: As the bits AOE, BKP, BKE, OSSI, OSSR and DTG[7:0] can be write-locked depending on the LOCK configuration, it can be necessary to configure all of them during the first write access to the TIMx_BDTR **register.**

Bits	Name	Type	Reset	Description
31:21	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
17:16	LOCK	R/W	0	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write protected. 01: LOCK Level 1 = DTG bits in **TIMx_BDTR** register, OISx and OISxN bits in **TIMx_CR2** register and BKE/BKP/AOE bits in **TIMx_BDTR** register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in **TIMx_CCER** register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 1 1: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in **TIMx_CCMR** 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.

Table 4.10-3: Lock table

Lock level	Protected register					
Lock level 1(LOCK bit 01)	DTG	OISx	OISxN	BKE	BKP	AOE
Lock level 2(LOCK bit 10)	DTG	OISx	OISxN	BKE	BKP	AOE
	CCxP	CCxNP	OSSR	OSSI		
Lock level 3(LOCK bit 11)	DTG	OISx	OISxN	BKE	BKP	AOE
	CCxP	CCxNP	OSSR	OSSI		
	OCxM	OCxPE				

15	MOE	R/W	0	Main output enable 0: OCN and OCxN outputs are disabled or forced to idle state. 1: OCN and OCxN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in **TIMx_CCER** register).
14	AOE	R/W	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 the break input is not be active) Note: This bit cannot be modified as long as LOCK level 1 has been programmed (LOCK bits in **TIMx_BDTR** register).

Bits	Name	Type	Reset	Description
13	BKP	R/W	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 programmed (LOCK bits in **TIMx_BDTR** register). Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.
12	BKE	R/W	0	Break enable 0: Break inputs BRK event disabled 1: Break inputs BRK event enabled *Note: This bit cannot be modified when LOCK level 1 has been programmed (LOCK bits in **TIMx_BDTR** register).* *Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.*
11	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10	OSSI	R/W	0	Off-state selection for Idle mode This bit is used when MOE=0 on channels configured as outputs. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal=0). 1: When inactive, OC/OCN outputs are forced first with their idle level as soon as CCxE=1 or CCxNE=1. OC/OCN enable output signal=1) Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in **TIMx_BDTR** register).
9:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.11 ADVANCED-CONTROL TIMERS (ADTM)

The advanced-control timers (TIM4 and TIM5) consist of a 16-bit auto-reload counter driven by a programmable prescaler.

They 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 APB clock controller prescalers.

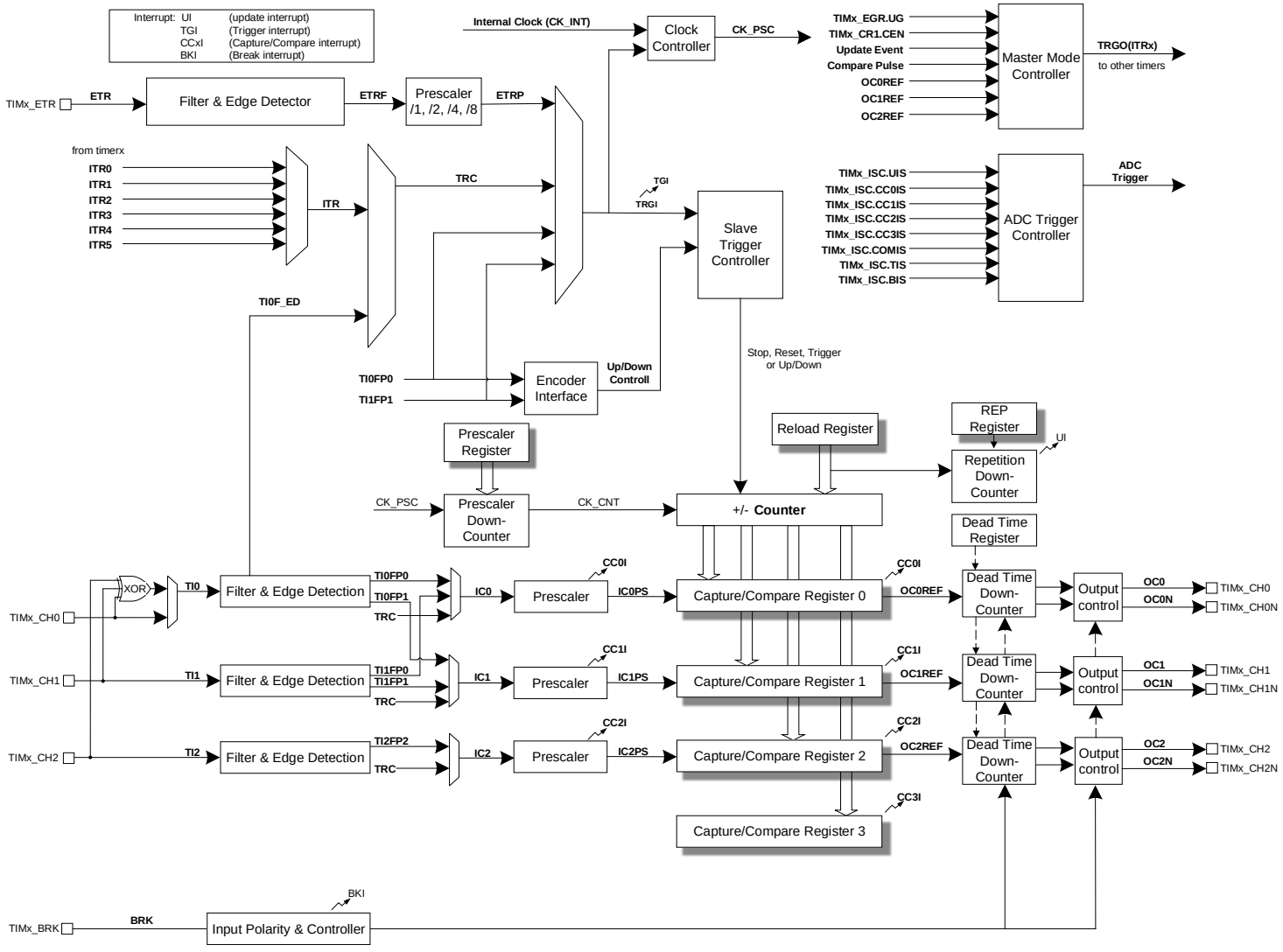
The advanced-control (TIM4, TIM5) and general-purpose (TIM0, TIM1, TIM2, TIM3) timers are completely independent, and do not share any resources.

Advanced-control Timers features include:

- 16-bit up, down, up/down auto-reload counter.
- 16-bit programmable prescaler allowing dividing (also “on the fly”) the counter clock frequency either by any factor between 1 and 65535.
- Up to 3 independent channels for:
  - Input Capture
  - Output Compare
  - PWM generation (Edge- and Center-aligned modes)
  - One-shot mode output
- Complementary outputs with programmable dead-time
- Synchronization circuit to control the timer with external signals and to interconnect several timers together.
- Repetition counter to update the timer registers only after a given number of cycles of the counter.
- Break input to put the timer’s output signals in reset state or in a known state.
- Interrupt 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 (capture register)
  - Output compare (counter register match compare register)
  - Break event
  - Overcapture event
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management.

4.11.1 BLOCK DIAGRAM

Figure 4.11-1: Advance timer block diagram



4.11.2 FUNCTIONAL DESCRIPTION

4.11.2.1 TIMER INITIALIZATION

The following bits are enable-protected, meaning that it can only be written when the Timer is disabled (**TIM4/5_CR1**.*CEN* is zero);

- Main output enable bit in the Break and dead-time register (**TIMx_BDTR.***MOE*). For timers equipped with deadtime generators, the *MOE* bit controls all outputs and acts a circuit breaker in case of fault detection on the break input (global disable of all PWM outputs)
- Capture/Compare x output enable bits in the Capture/Compare enable register (**TIM4/5_CCER**.*CCnE*)

4.11.2.2 TIMER-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 timer-base unit includes:

- Counter register (**TIMx_CNTR**)
- Prescaler register (**TIMx_PSC**)
- Auto-reload register (**TIMx_ARR**)
- Repetition counter register (**TIMx_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 **TIMx_CR1** register. The update event is sent when the counter reaches the overflow when upcounting (or underflow when downcounting) and if the *UDIS* bit equals 0 in the **TIMx_CR1** register. It can also be generated by software. The generation of the update event is described in detailed for each configuration.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (*CEN*) in **TIMx_CR1** register is set (refer also to the slave mode controller description to get more details on counter enabling).

Note that the actual counter enable signal CNT_EN is set 1 clock cycle after *CEN* bit in the TIMx_CR1 register.

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. The prescaler counter is down-counting only. When the prescaler is set, the value of the counter will counting up or down as soon as the prescaler counter reaches 0, otherwise, the counter updates along with the APB clock.
Figure 4.11-2 and

give some examples of the counter behavior when the prescaler ratio is changed on the fly:

Figure 4.11-2: Counter timing diagram with prescaler division change from 1 to 2

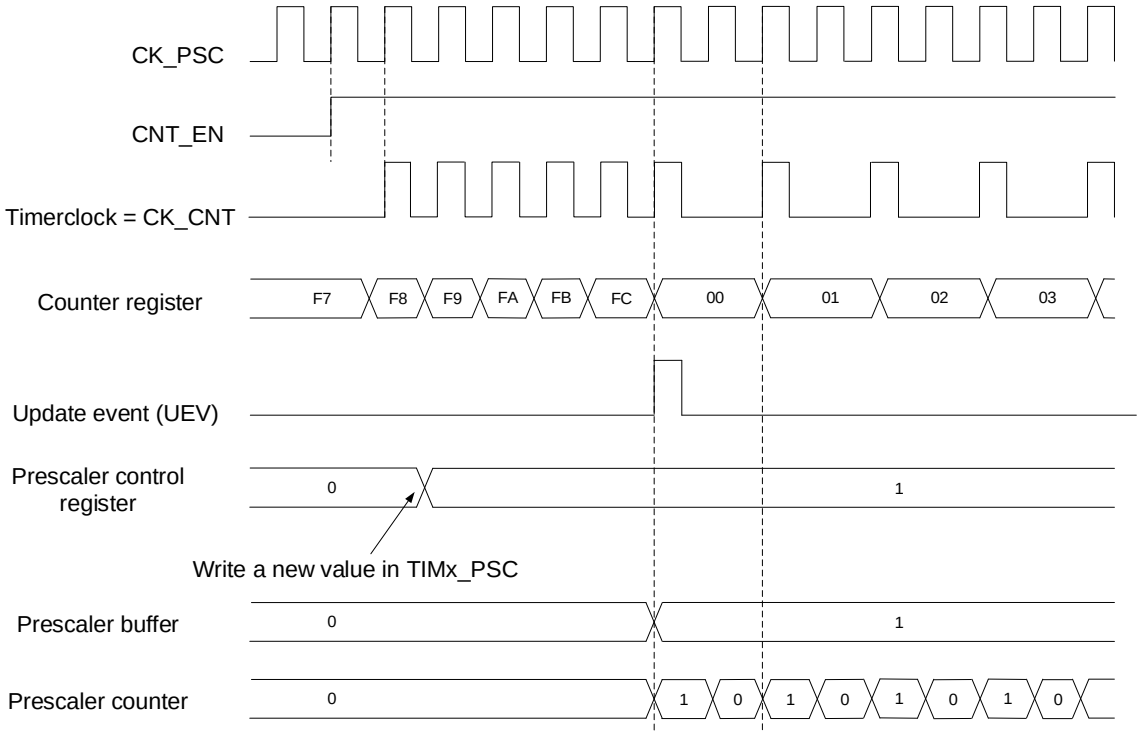
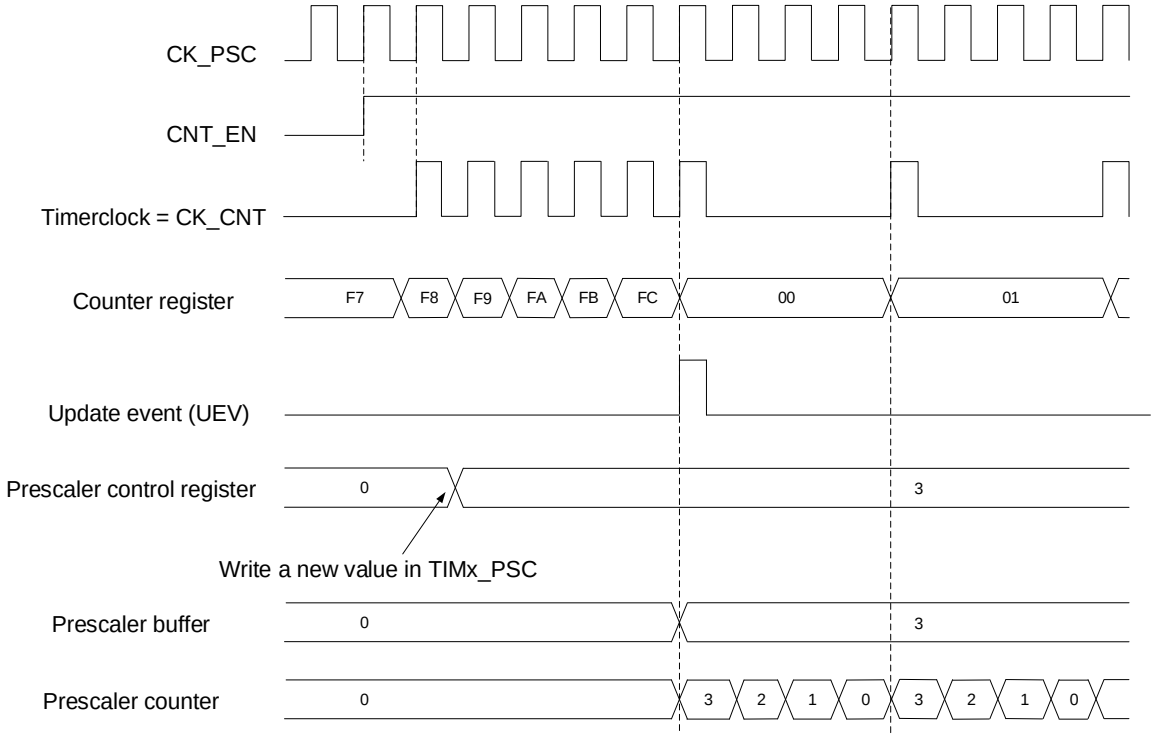


Figure 4.11-3: Counter timing diagram with prescaler division change from 1 to 4



4.11.2.3 COUNTER MODES

UPCOUNTING MODE

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the **TIMx_ARR** register), then restarts from 0 and generates a counter overflow event.

If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times programmed in the repetition counter register (**TIMx_RCR**). Else the update event is generated at each counter overflow.

An Update event can be generated at each counter overflow or by setting the UG bit in the **TIMx_EGR** register (by software or by using the slave mode controller).

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 the *UDIS* bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change).

When an update event occurs, all the registers are updated and the update status (*UIS* bit in **TIMx_ISC** register) is set:

- The repetition counter is reloaded with the content of **TIMx_RCR** register,
- The auto-reload 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 show some examples of the counter behavior for different clock frequencies when **TIMx_ARR**=0x36.

Figure 4.11-4: Counter timing diagram, internal clock divided by 1

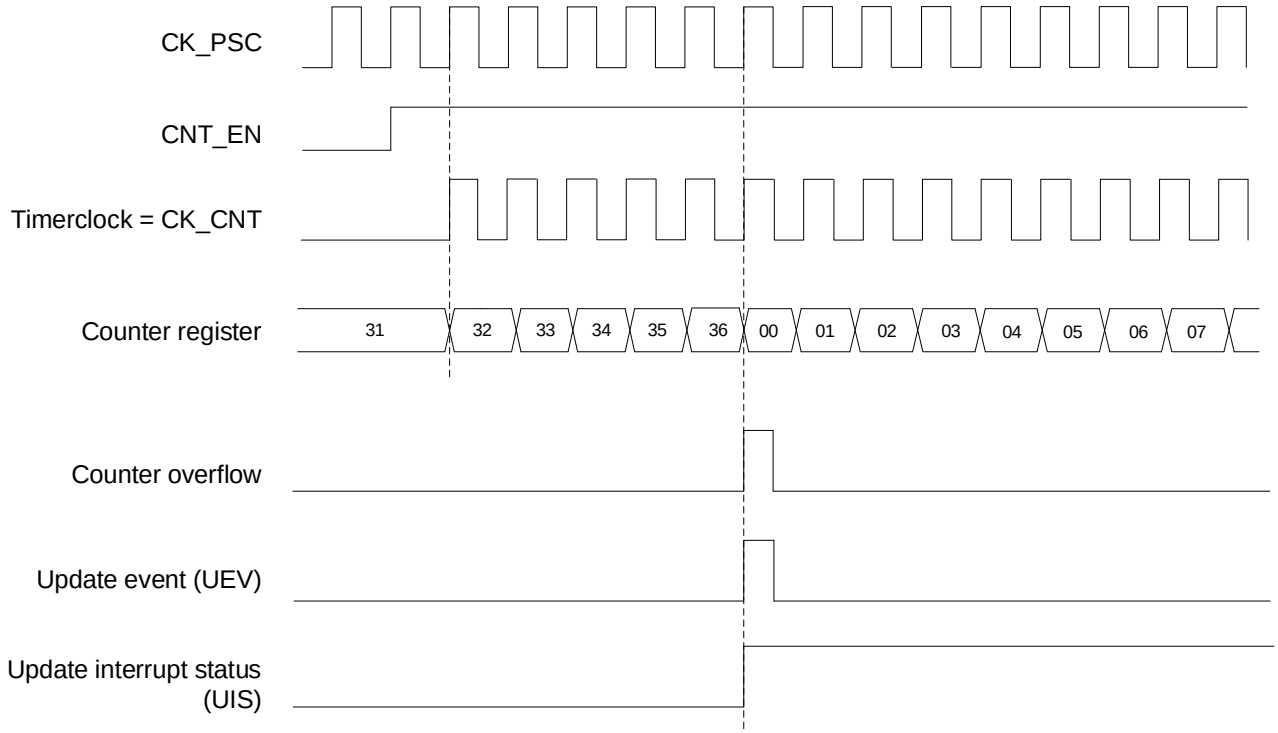


Figure 4.11-5: Counter timing diagram, internal clock divided by 2

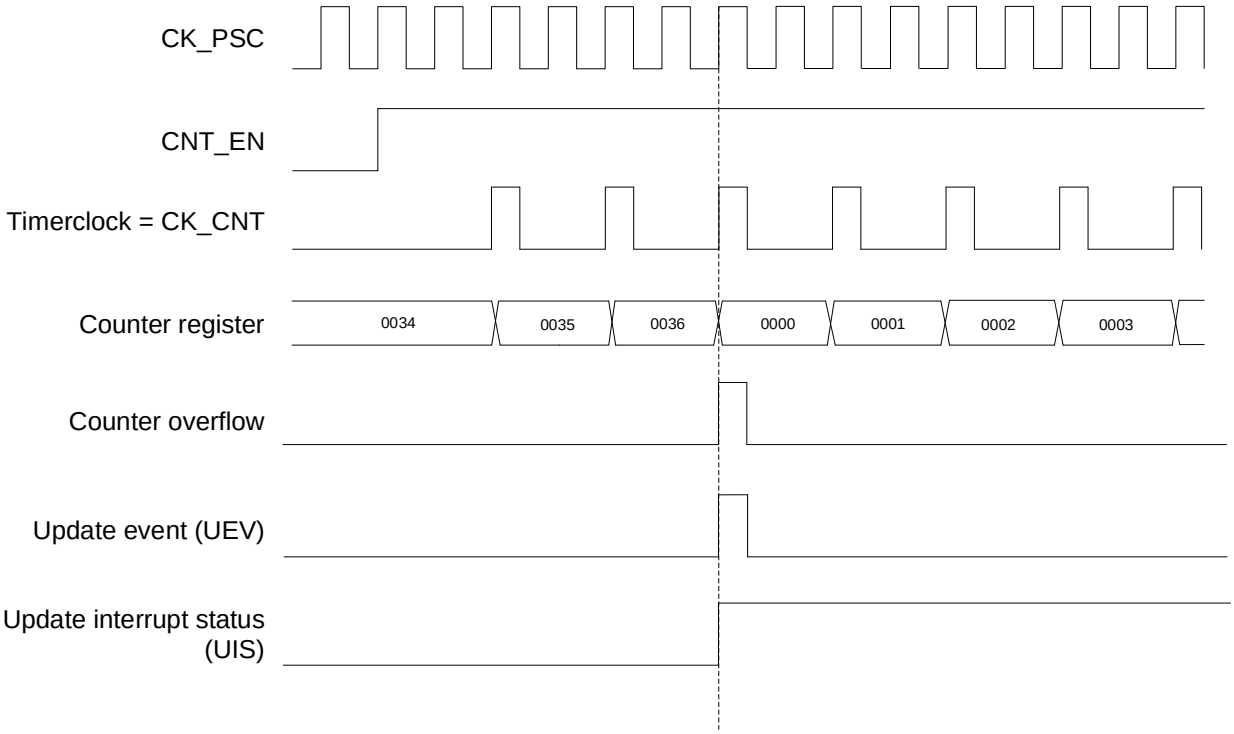


Figure 4.11-6: Counter timing diagram, internal clock divided by 4

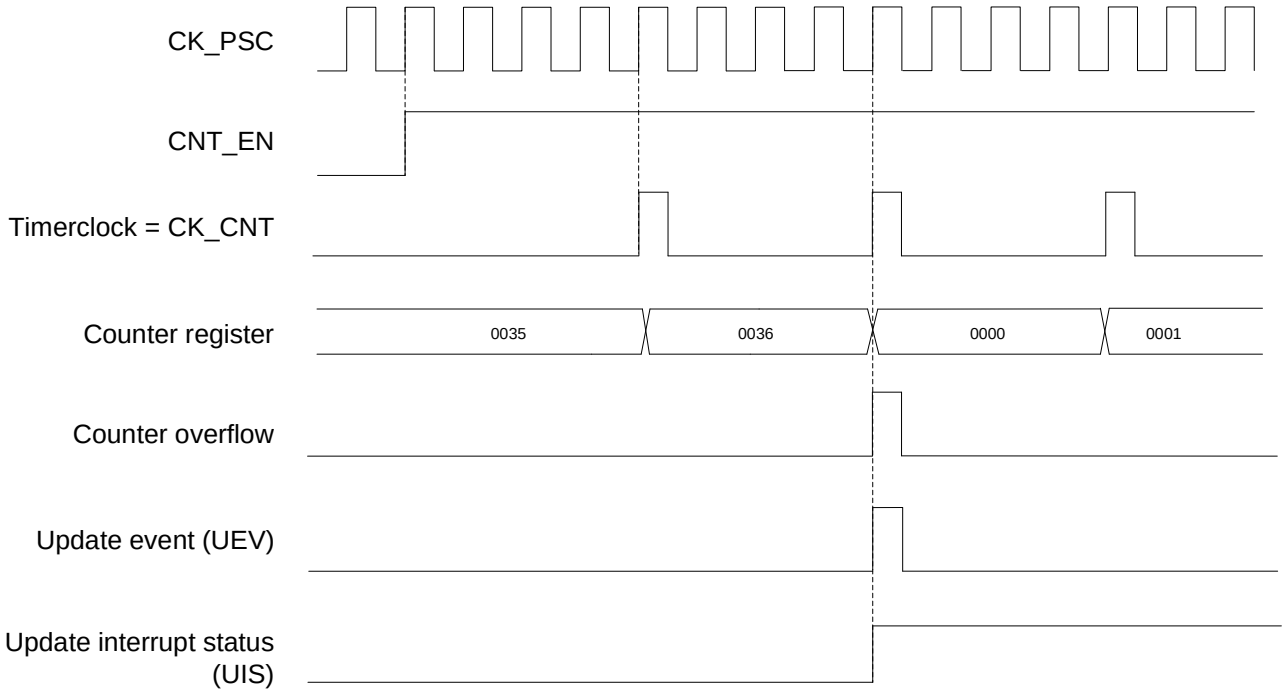


Figure 4.11-7: Counter timing diagram, internal clock divided by N

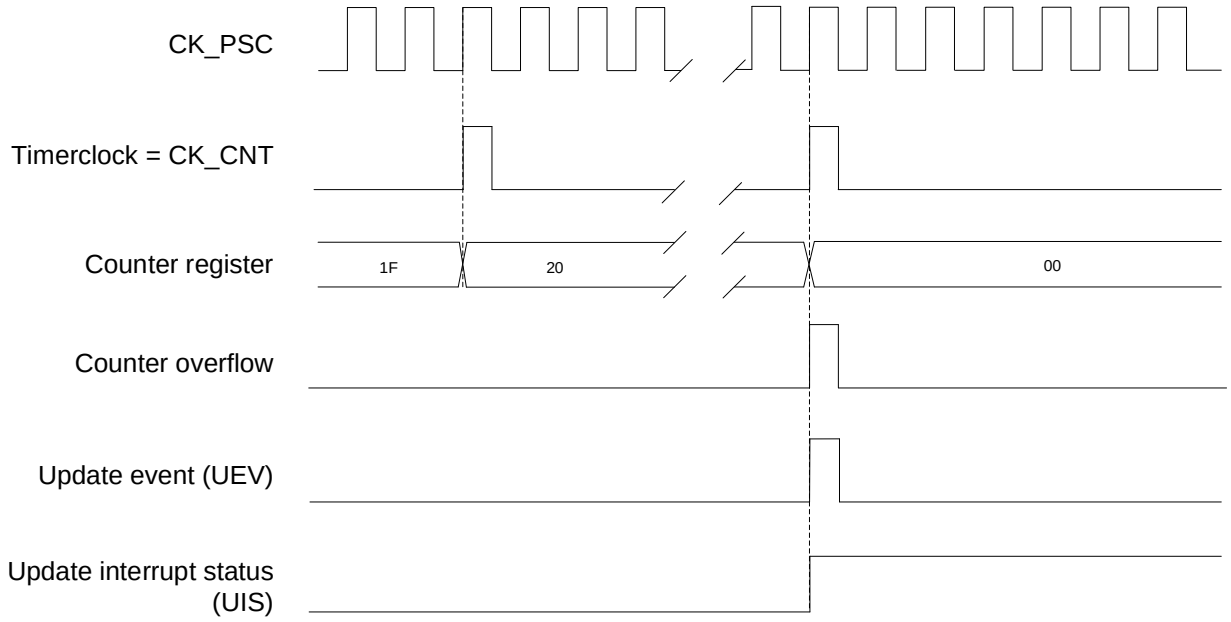


Figure 4.11-8: Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

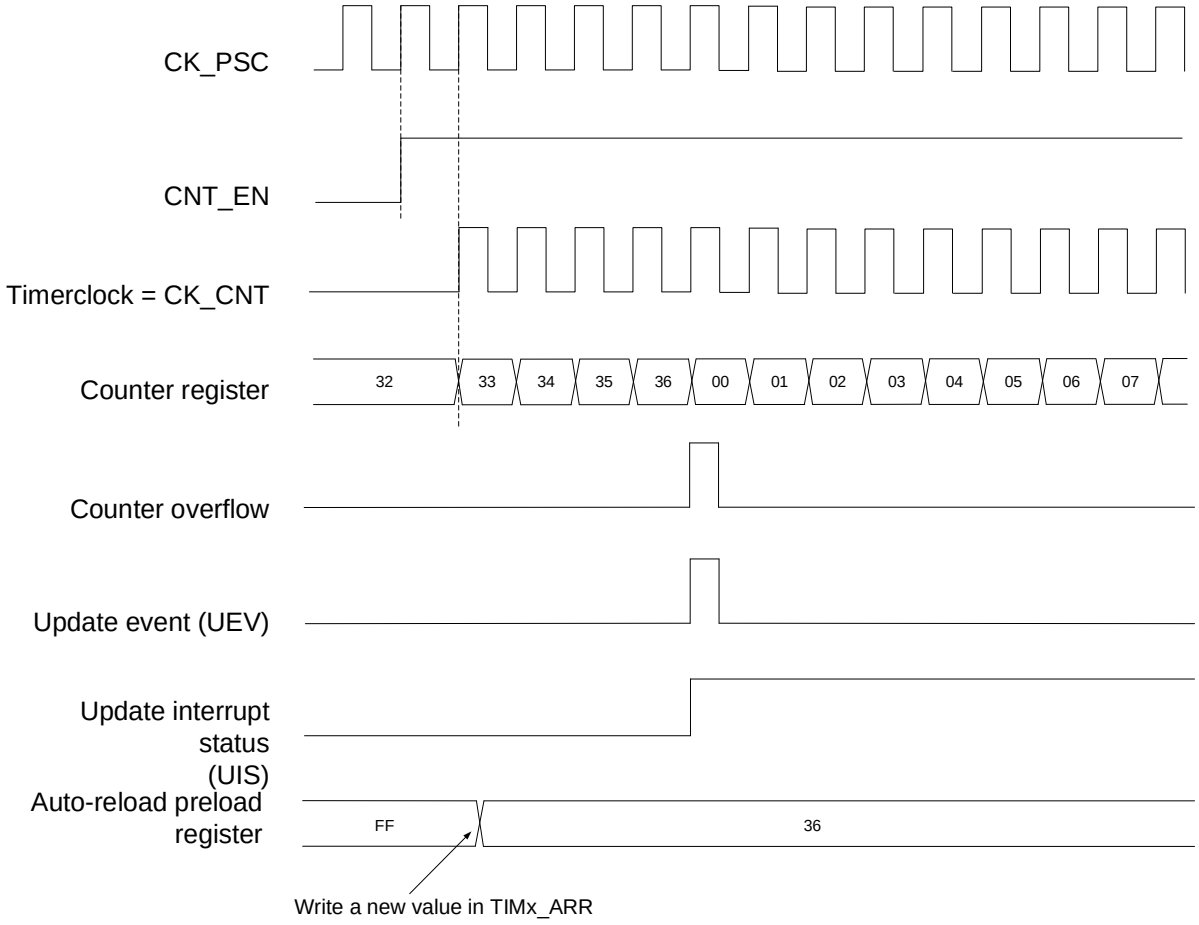
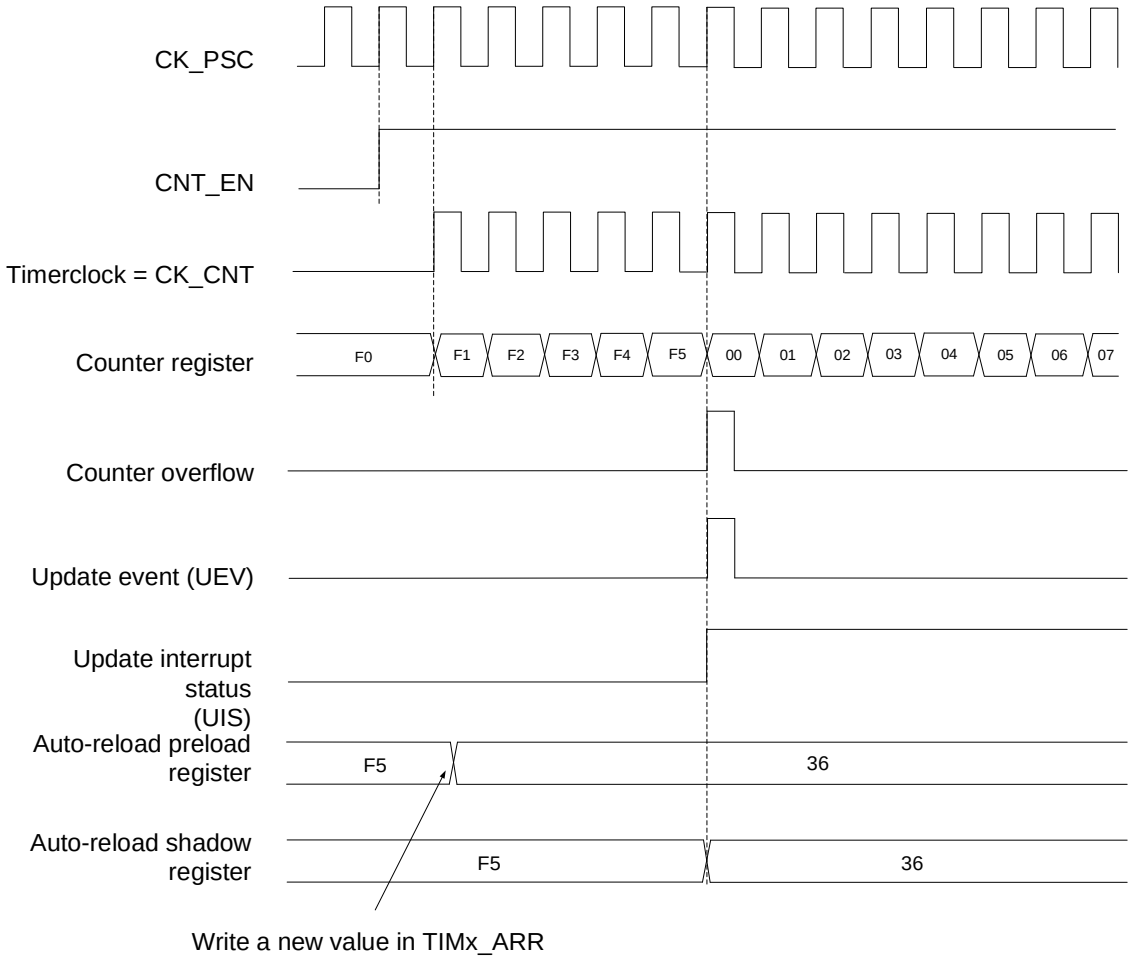


Figure 4.11-9: Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)



DOWNCOUNTING MODE

In downcounting mode, the counter counts from the auto-reload value (content of the **TIMx_ARR** register) down to 0, then restarts from the auto-reload value and generates a counter underflow event.
If the repetition counter is used, the update event (UEV) is generated after downcounting is repeated for the number of times programmed in the repetition counter register (**TIMx_RCR**). Else the update event is generated at each counter underflow.

Setting the UG bit in the **TIMx_EGR** register (by software or by using the slave mode controller) also generates an update event.

The UEV update event can be disabled by software by setting the UDIS bit in **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 restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).

When an update event occurs, all the registers are updated and the update status (UIS bit in TIMx_ISC register) is set:

- The repetition counter is reloaded with the content of **TIMx_RCR** register
- 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

The following figures show some examples of the counter behavior for different clock frequencies when **TIMx_ARR**=0x36.

Figure 4.11-10: Counter timing diagram, internal clock divided by 1

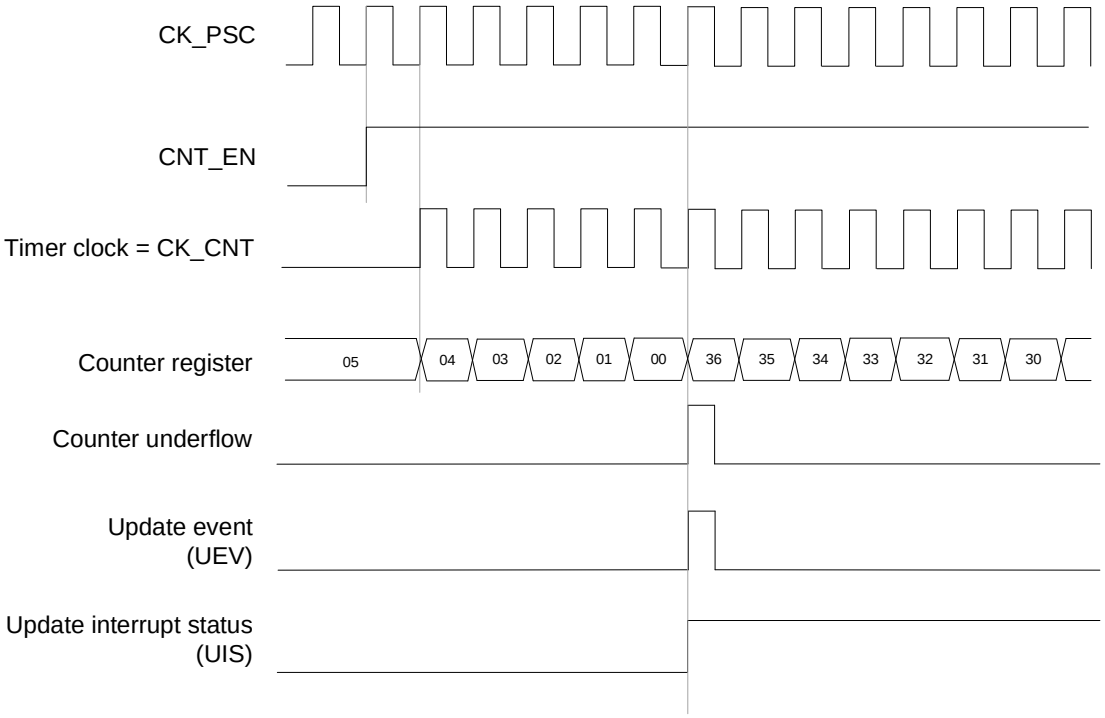


Figure 4.11-11: Counter timing diagram, internal clock divided by 2

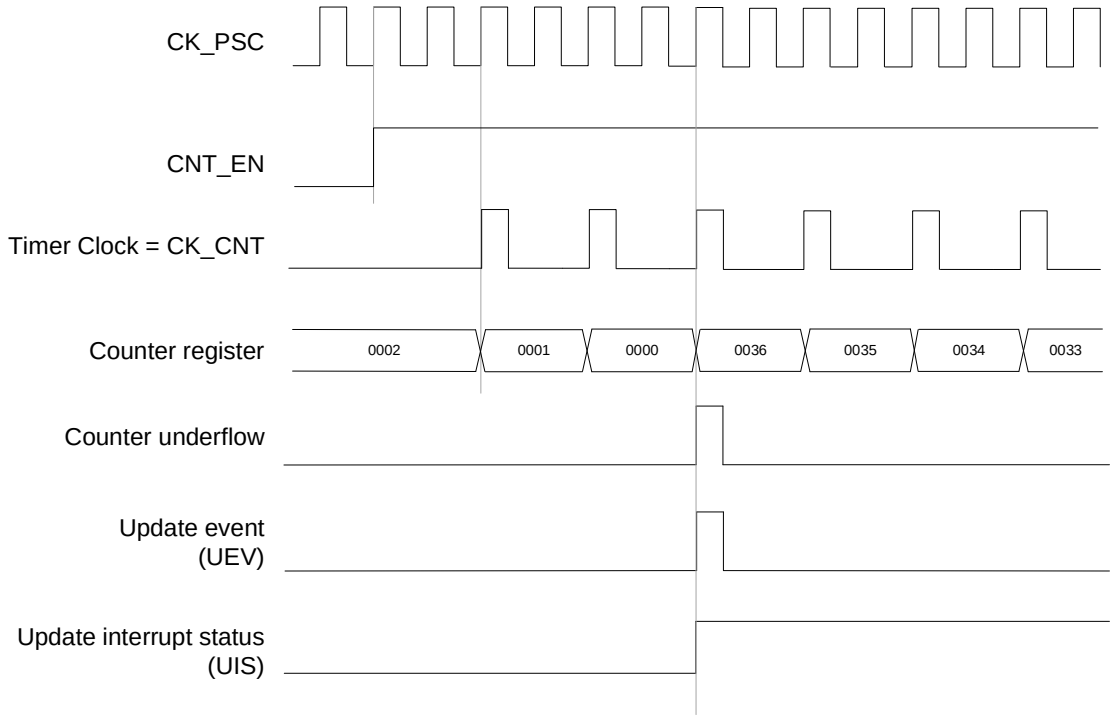


Figure 4.11-12: Counter timing diagram, internal clock divided by 4

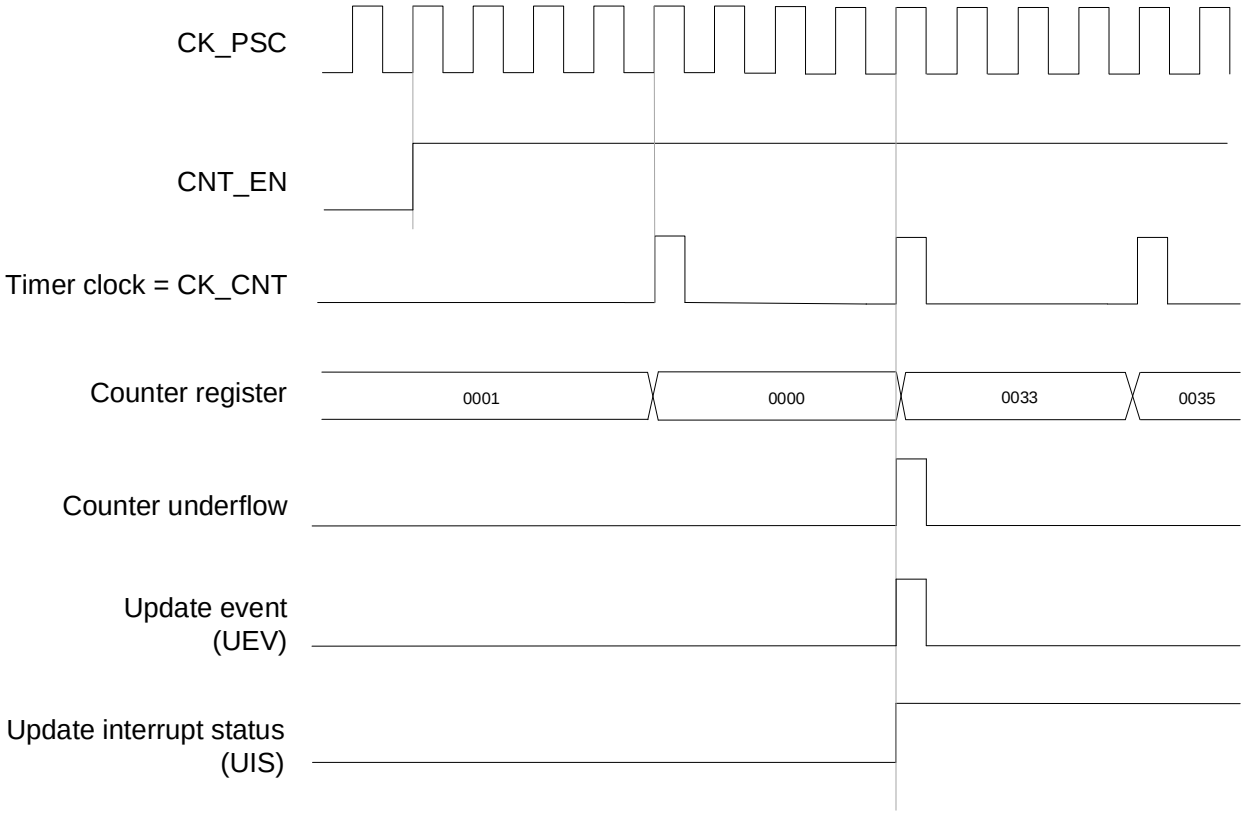


Figure 4.11-13: Counter timing diagram, internal clock divided by N

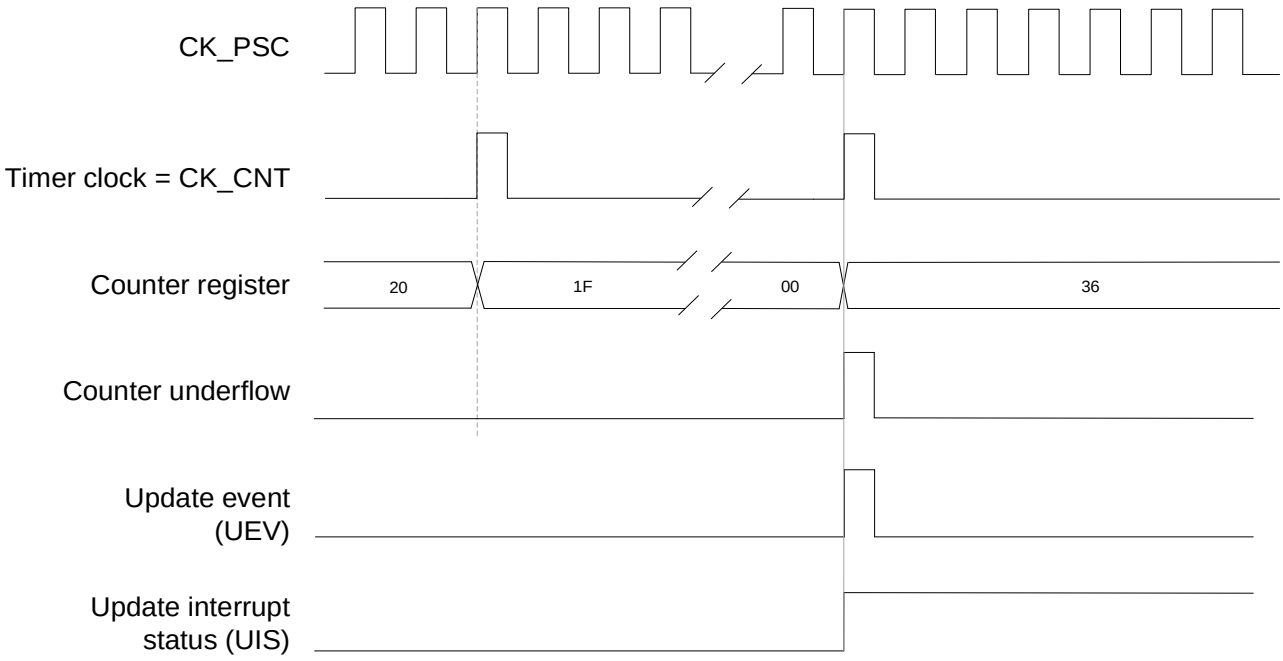
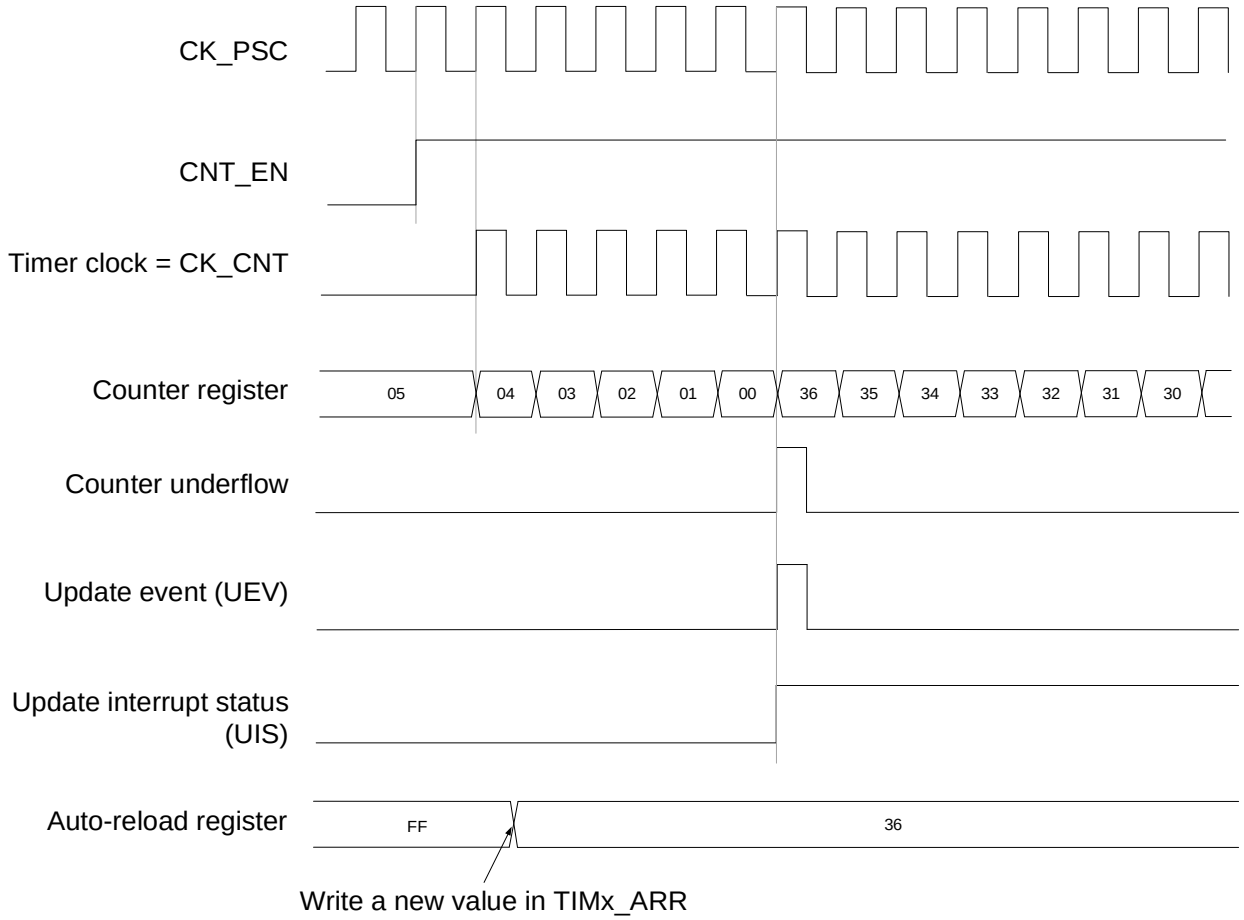


Figure 4.11-14: Counter timing diagram, update event when repetition counter is not used



CENTER-ALIGNED MODE (UP/DOWN COUNTING)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the **TIMx_ARR** register) - 1, generates a counter overflow event, then counts from the auto-reload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

Center-aligned mode is active when the CMS bits in **TIMx_CR1** register are not equal to '00'. The Output compare interrupt status of channels configured in output is set when: the counter counts down (Center aligned mode 1, CMS = “01”), the counter counts up (Center aligned mode 2, CMS = “10”) the counter counts up and 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 update 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 counting up and down, based on the current auto-reload value. 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 registers are updated and the update status (UIS bit in TIMx_ISC register) is set:

- The repetition counter is reloaded with the content of **TIMx_RCR** register
- 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 if the update source is a counter overflow, the auto-reload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value).

Figure 4.11-15: Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

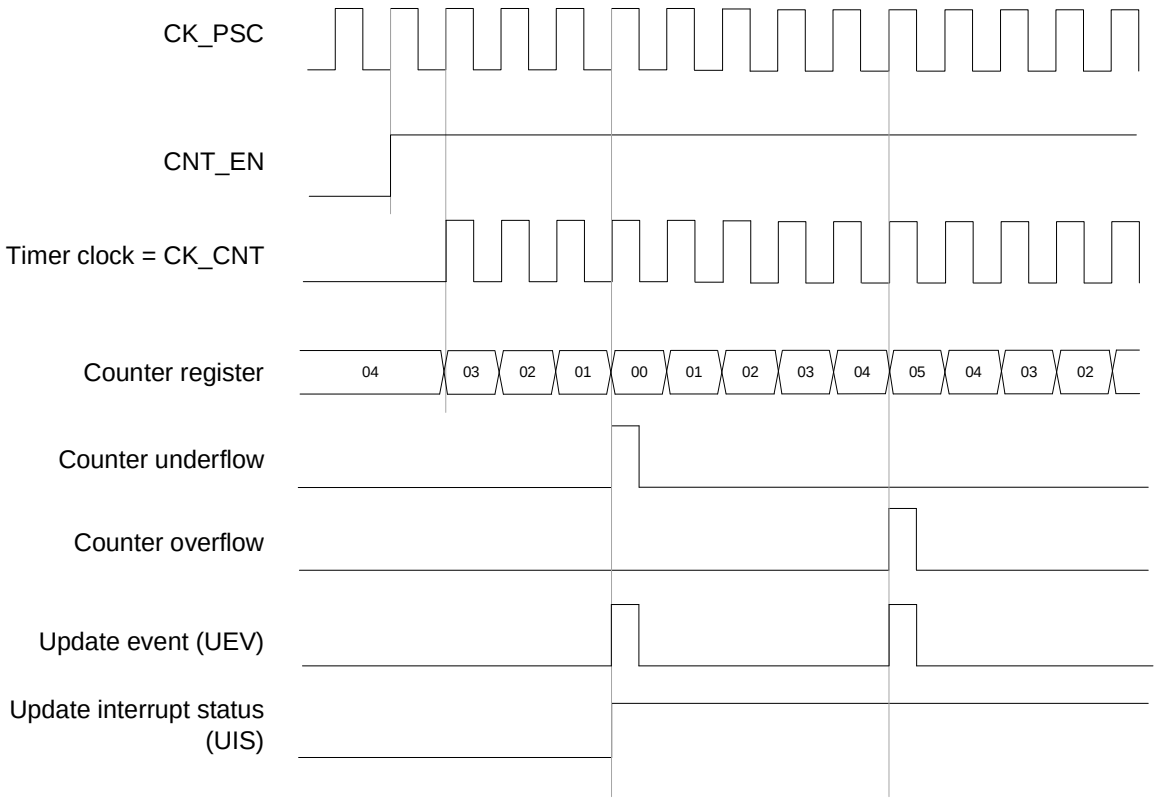


Figure 4.11-16: Counter timing diagram, internal clock divided by 2

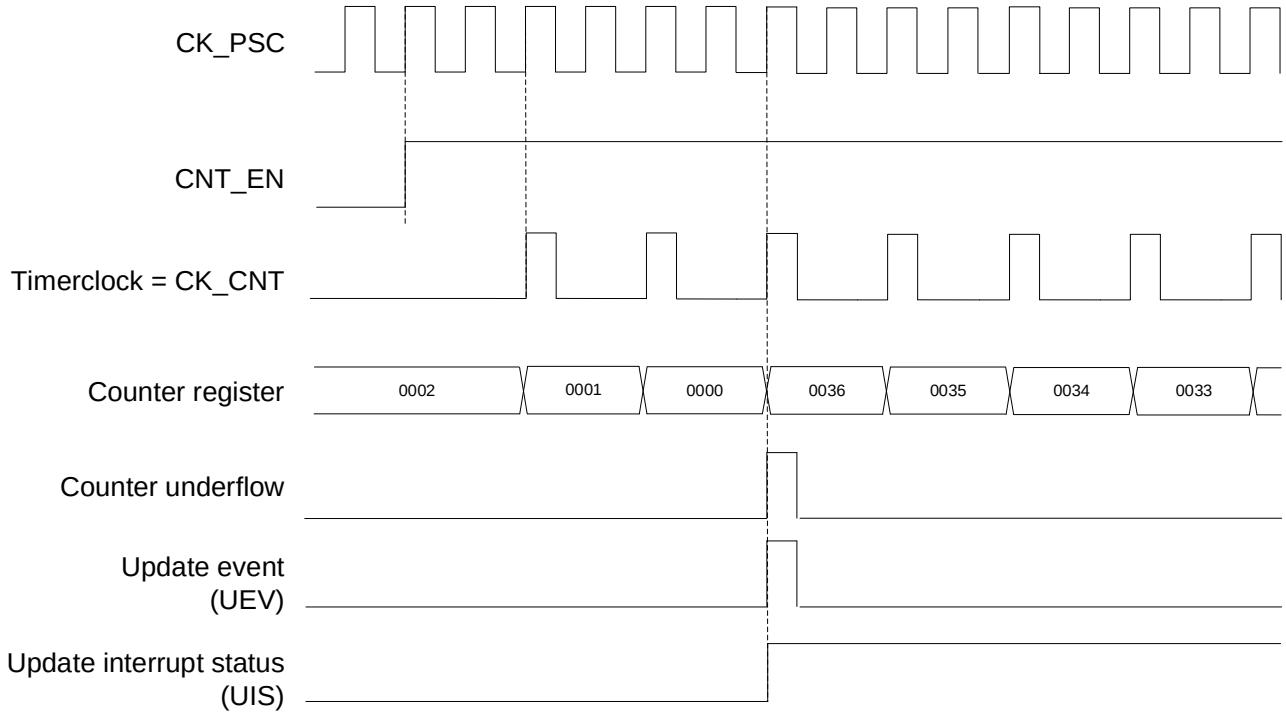
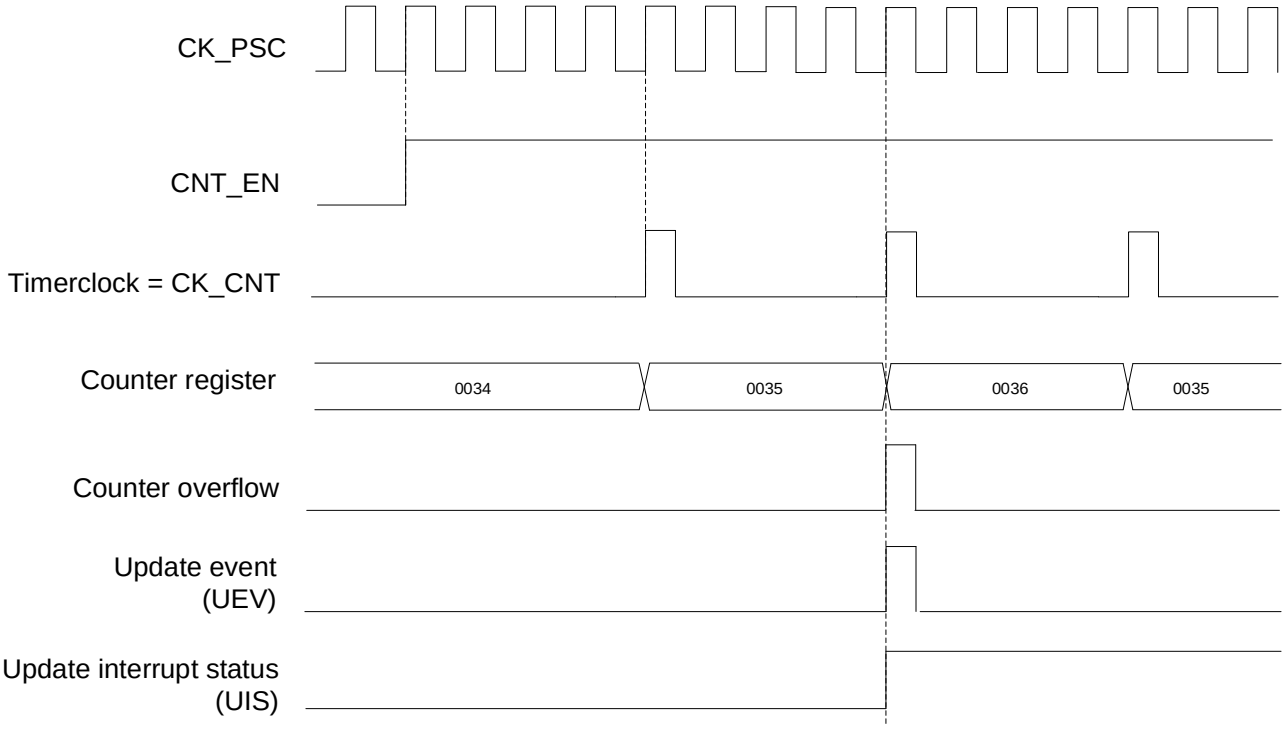


Figure 4.11-17: Counter timing diagram, internal clock divided by 4, TIMx_ARR = 0x36



Note: Here, center_aligned mode 2 or 3 is update with an UIS on overflow.
1. Center-aligned mode 2 or 3 is used with an UIS on overflow.

Figure 4.11-18: Counter timing diagram, internal clock divided by N

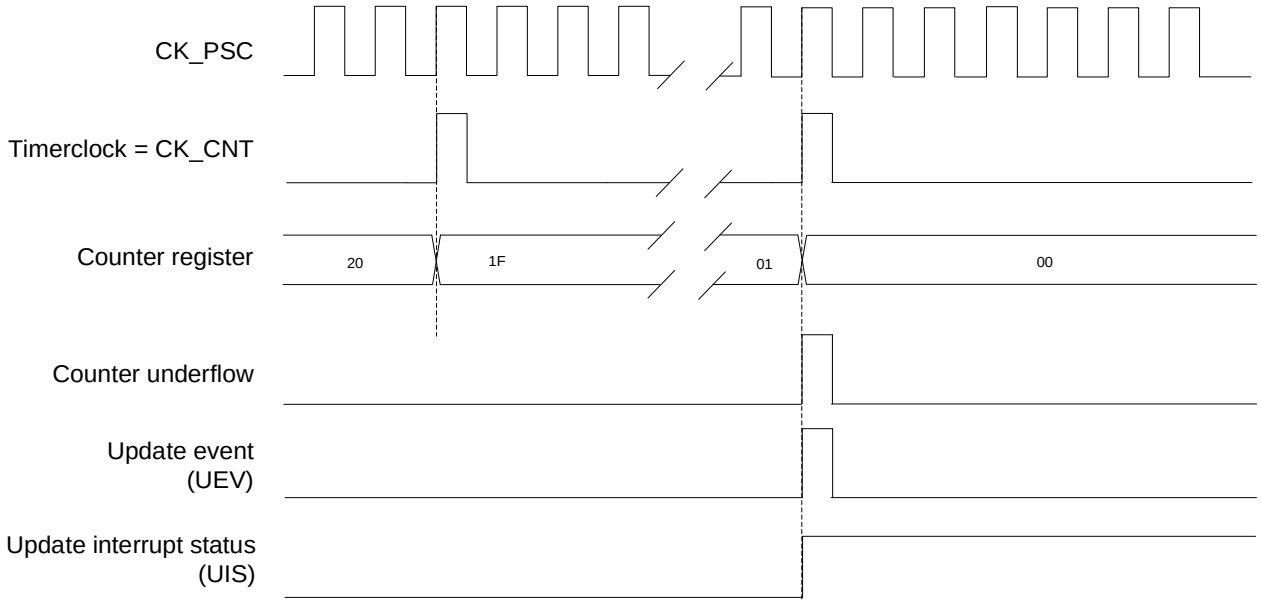


Figure 4.11-19: Counter timing diagram, update event with ARPE=1 (counter underflow)

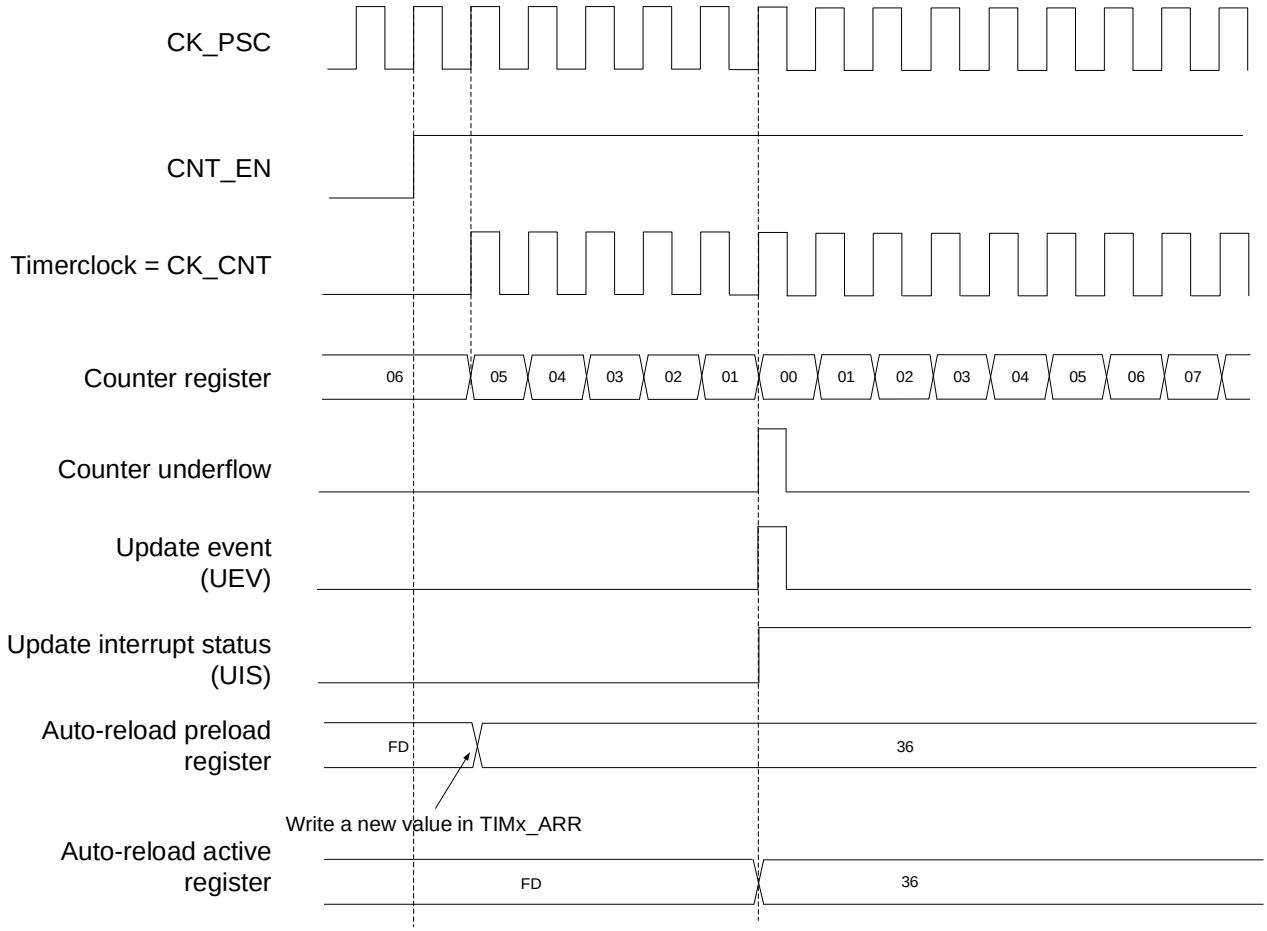
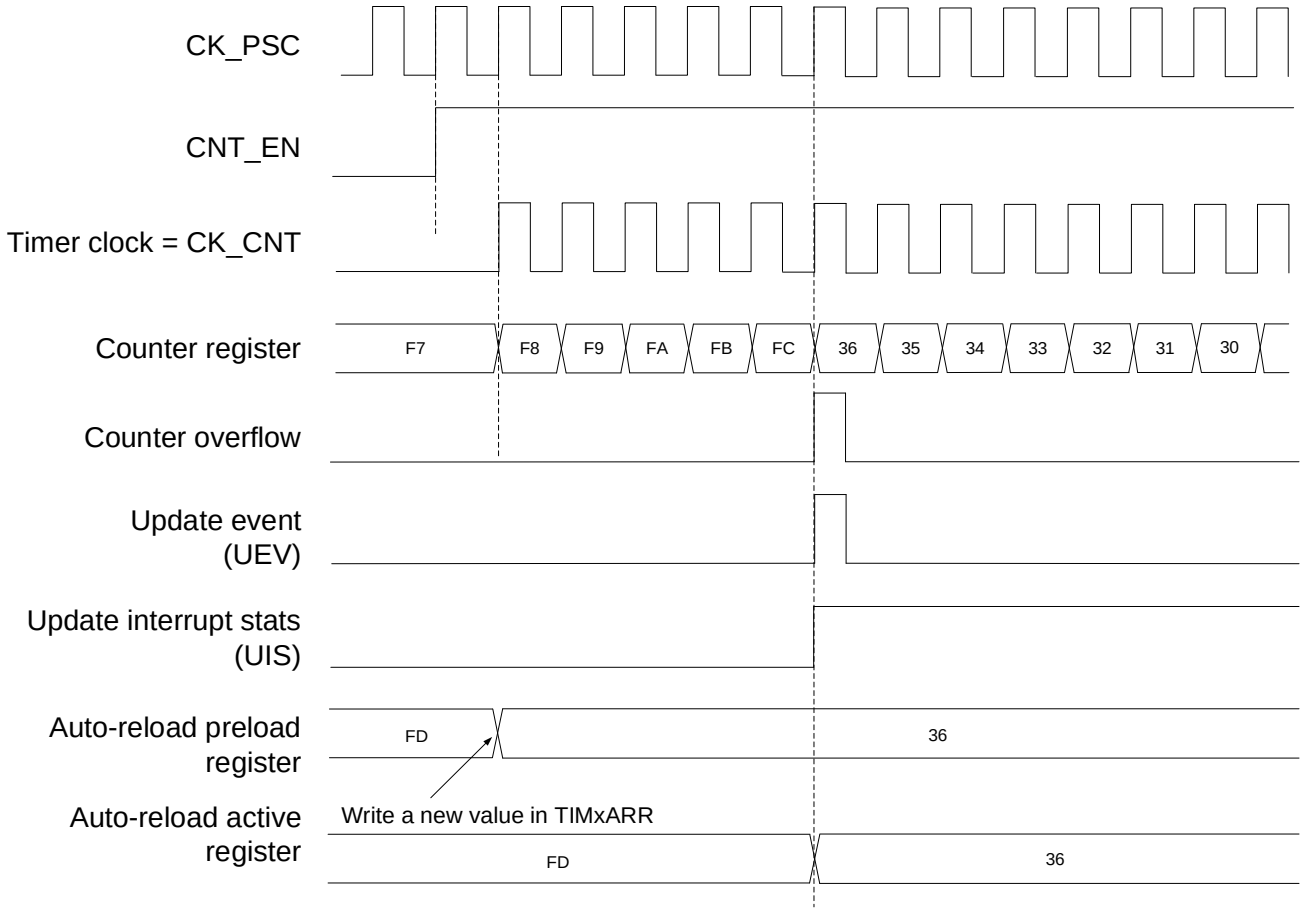


Figure 4.11-20: Counter timing diagram, Update event with ARPE=1 (counter overflow)



4.11.2.4 REPETITION COUNTER

Timer-base Unit describes how the update event (UEV) is generated with respect to the counter overflows/underflows. It is actually generated only when the repetition counter has reached zero. This can be useful when generating PWM signals.

This means that data are transferred from the preload registers to the shadow registers (**TIMx_ARR** auto-reload register, **TIMx_PSC** prescaler register, but also **TIMx_CCRx** capture/compare registers in compare mode) every N counter overflows or underflows, where N is the value in the **TIMx_RCR** repetition counter register.

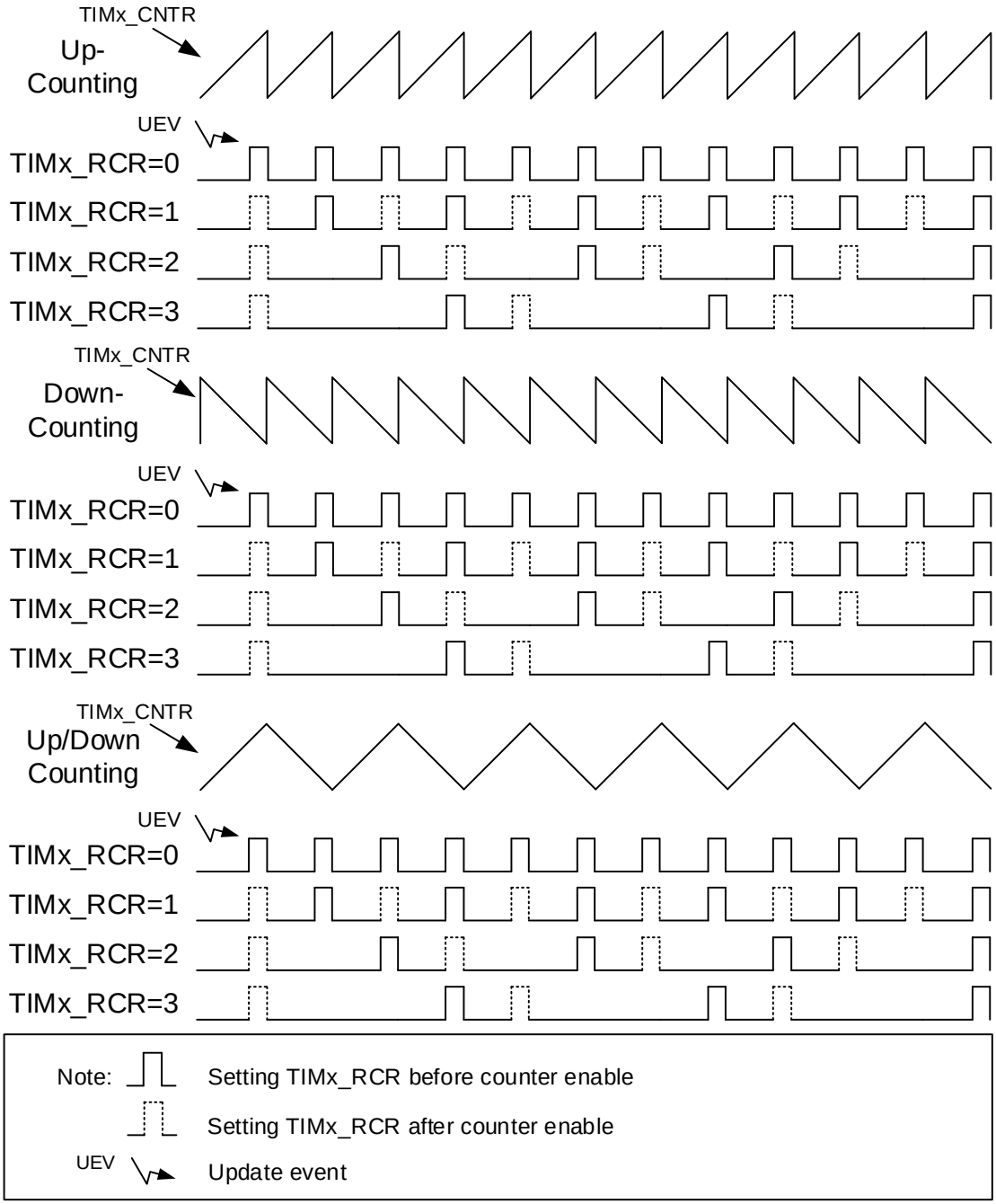
The repetition counter is decremented:

- At each counter overflow in upcounting mode,
- At each counter underflow in downcounting mode,
- At each counter overflow and at each counter underflow in center-aligned mode.Although this limits the maximum number of repetition to 128 PWM cycles, it makes it possible to update the duty cycle twice per PWM period. When refreshing compare registers only once per PWM period in center-aligned mode, maximum resolution is 2xTck, due to the symmetry of the pattern.

The repetition counter is an auto-reload type; the repetition rate is maintained as defined by the **TIMx_RCR** register value (refer to Figure 4.11-21). When the update event is generated by software (by setting the UG bit in **TIMx_EGR** register) or by hardware through the slave mode controller, it occurs immediately whatever the value of the repetition counter is and the repetition counter is reloaded with the content of the **TIMx_RCR** register.

In center-aligned mode, for odd values of RCR, the update event occurs either on the overflow or on the underflow depending on when the RCR register was written and when the counter was started. If the RCR was written before starting the counter, the UEV occurs on the underflow. If the RCR was written after starting the counter, the UEV occurs on the overflow. For example for RCR = 3, the UEV is generated on each 4th overflow or underflow event depending on when RCR was written.

Figure 4.11-21: Update rate examples depending on mode and TIMx_RCR register settings



4.11.2.5 CLOCK SOURCES

The counter clock can be provided by the following clock sources:

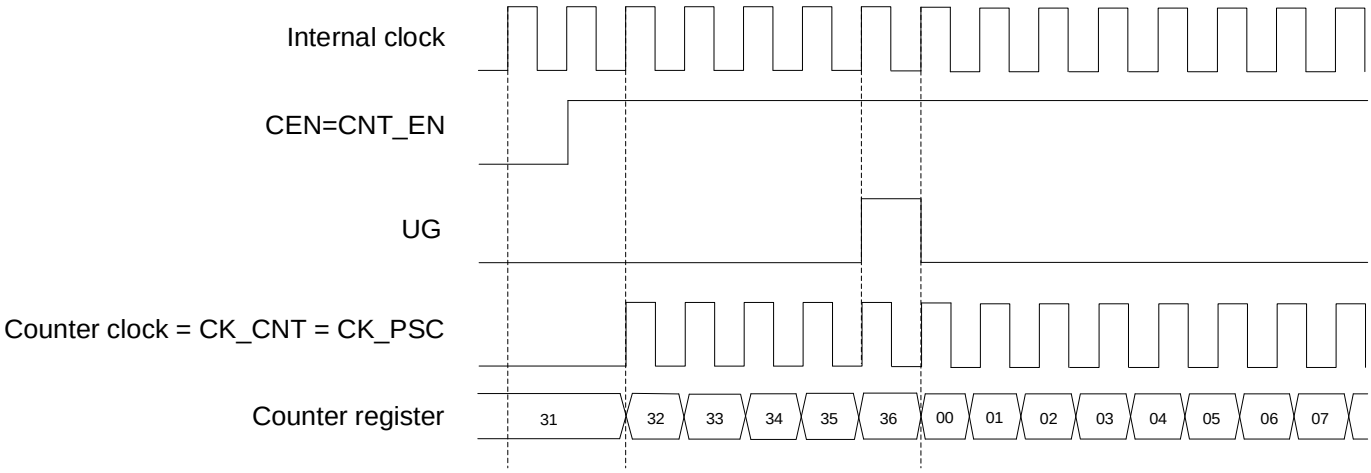
- Internal clock (CK_INT)
- External clock mode1: external input pin
- External clock mode2: external trigger input ETR
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, you can configure Timer 1 to act as a prescaler for Timer 2.

INTERNAL CLOCK SOURCE (CK_INT)

If the slave mode controller is disabled (SMS=000 in the **TIMx_SMCR** register), then the *CEN*, *DIR* (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 *UG* which remains cleared automatically). As soon as the *CEN* bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

Figure 4.11-22 shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

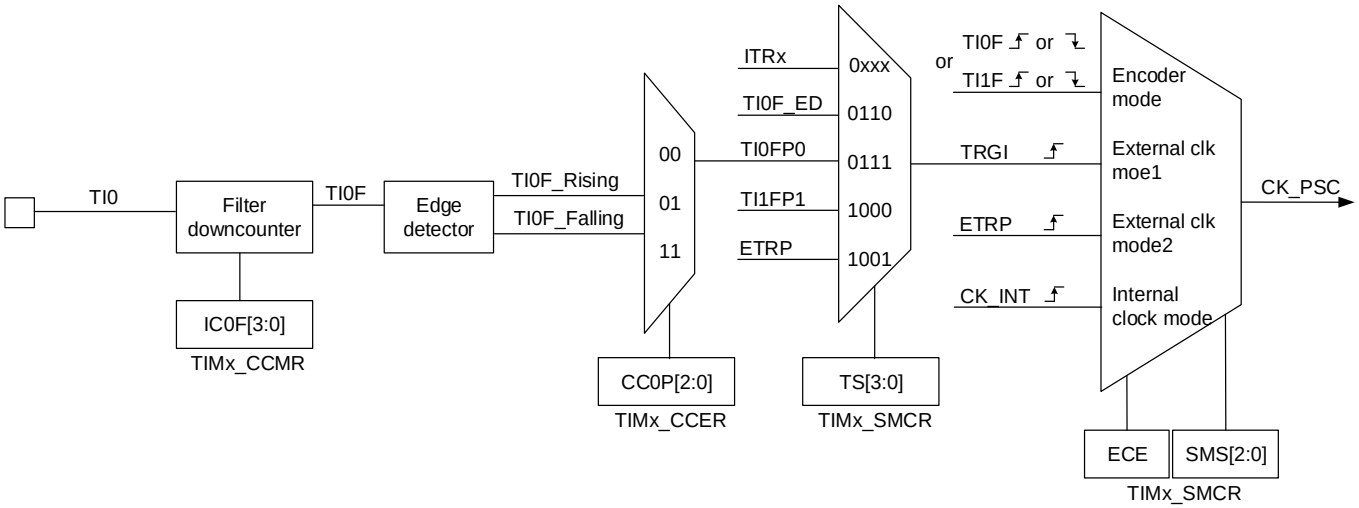
Figure 4.11-22: Control circuit in normal mode, internal clock divided by 1



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 4.11-23: TI0 external clock connection example



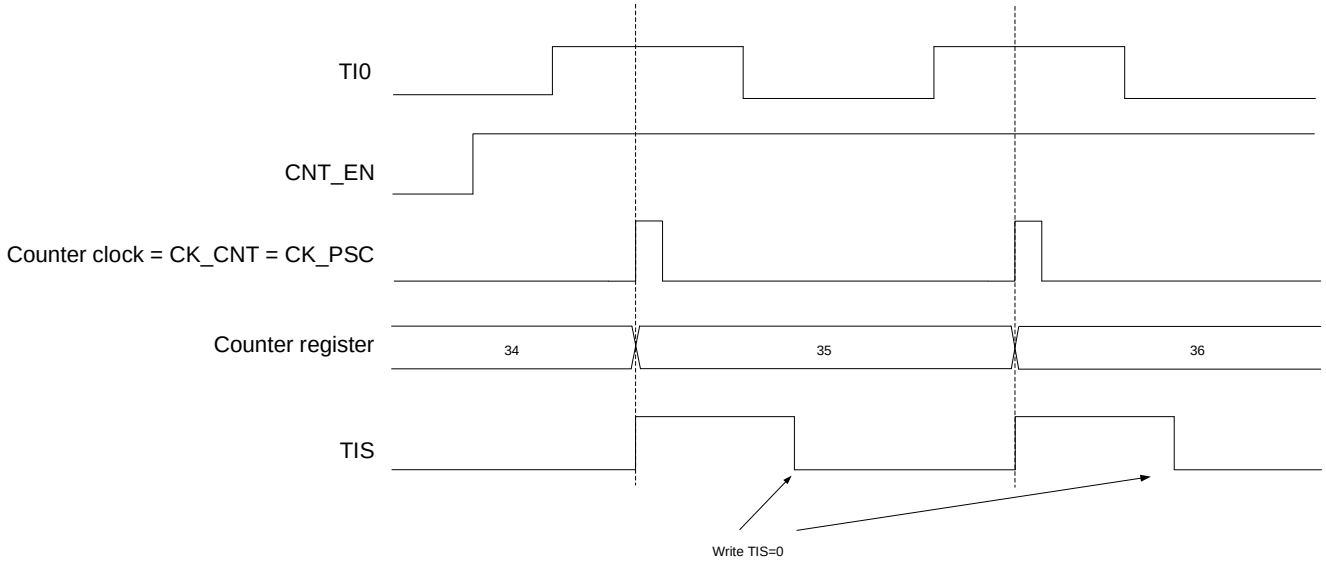
For example, to configure the upcounter to count in response to a rising edge on the TI0 input, use the following procedure:

1. Configure channel 0 to detect rising edges on the TI0 input by writing CC0S = '01' in the TIMx_CCMR register.
2. Configure the input filter duration by writing the IC0F[3:0] bits in the TIMx_CCMR register (if no filter is needed, keep IC0F=0000)
3. Select rising edge polarity by writing CC0P=00 in the TIMx_CCER register.
4. Configure the timer in external clock mode 1 by writing SMS=111 in the TIMx_SMCR register.
5. Select TI0 as the trigger input source by writing TS=0111 in the TIMx_SMCR register.
6. Enable the counter by writing CEN=1 in the TIMx_CR1 register.

When a rising edge occurs on TI0, the counter counts once and the TIS status is set.

The delay between the rising edge on TI0 and the actual clock of the counter is due to the resynchronization circuit on TI0 input.

Figure 4.11-24: Control circuit in external clock mode 1

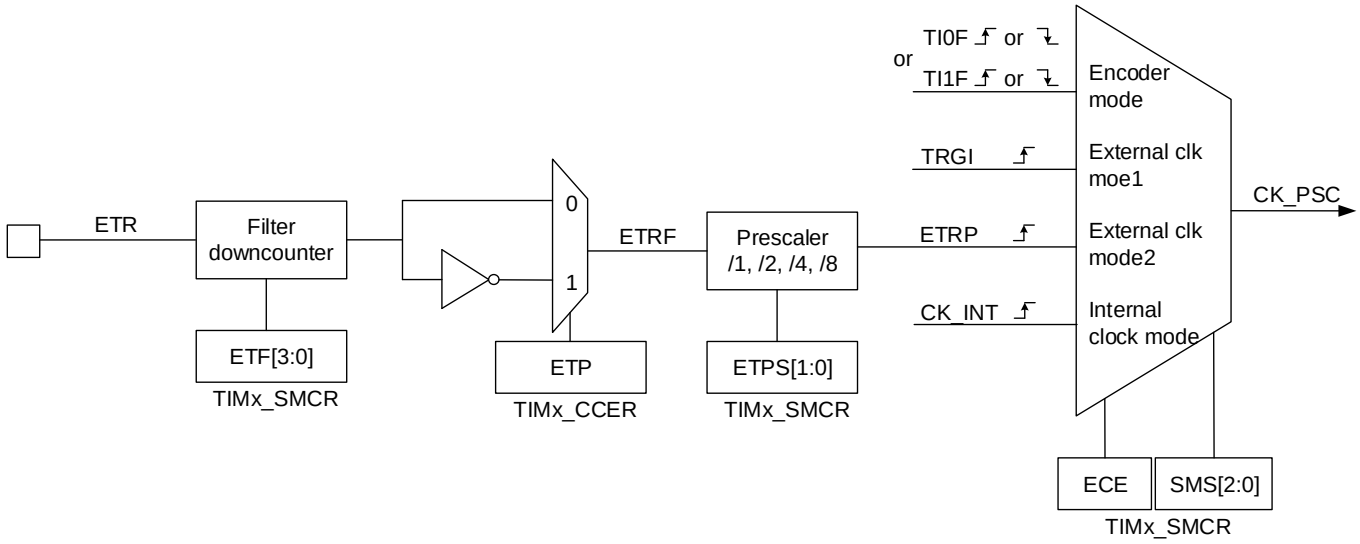


EXTERNAL CLOCK SOURCE MODE 2

This mode is selected by writing ECE=1 in the **TIMx_SMCR** register.

The counter can count at each rising or falling edge on the external trigger input ETR.

Figure 4.11-25: TI0 gives an overview of the external trigger input block

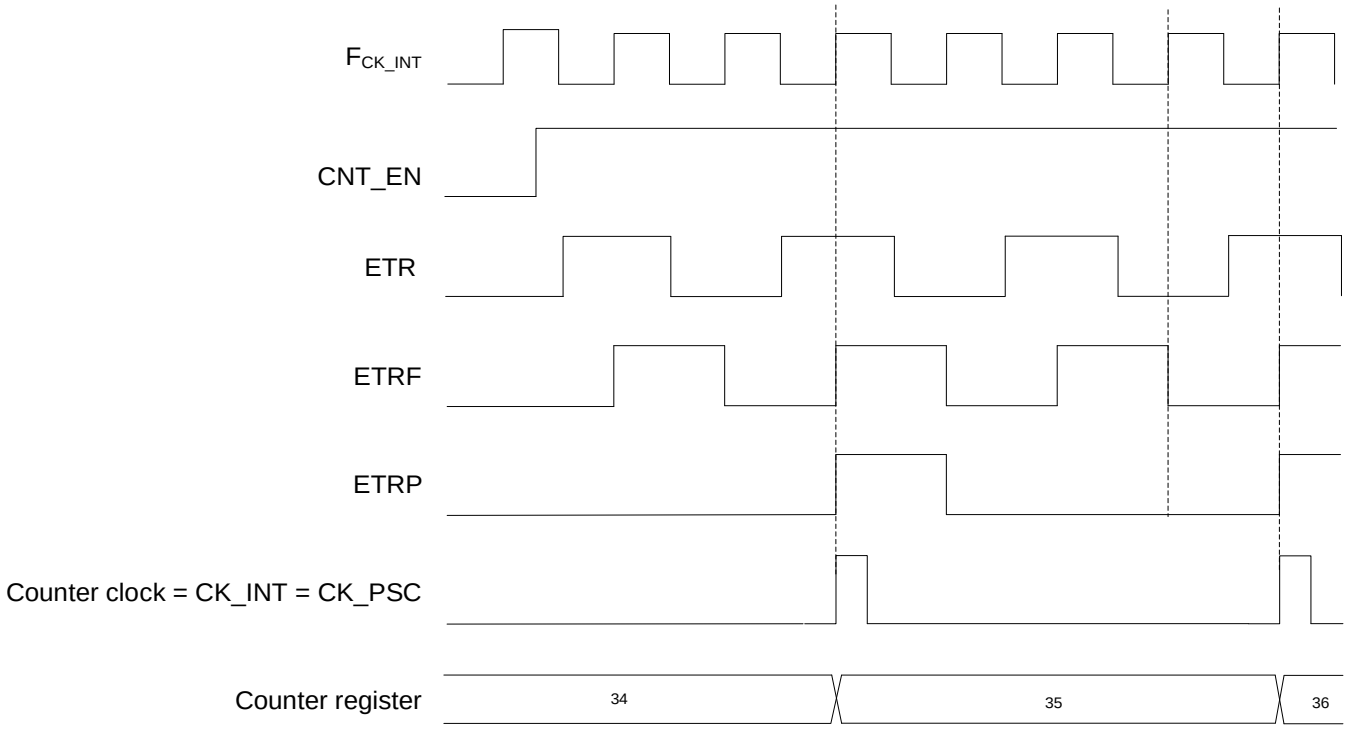


For example, to configure the upcounter to count each 2 rising edges on ETR, use the following procedure:

1. As no filter is needed in this example, write ETF[3:0]=0000 in the TIMx_SMCR register.
2. Set the prescaler by writing ETPS[1:0]=01 in the TIMx_SMCR register
3. Select rising edge detection on 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 TIMx_CR1 register.

The counter counts once each 2 ETR rising edges

Figure 4.11-26: Control circuit in external clock mode 2

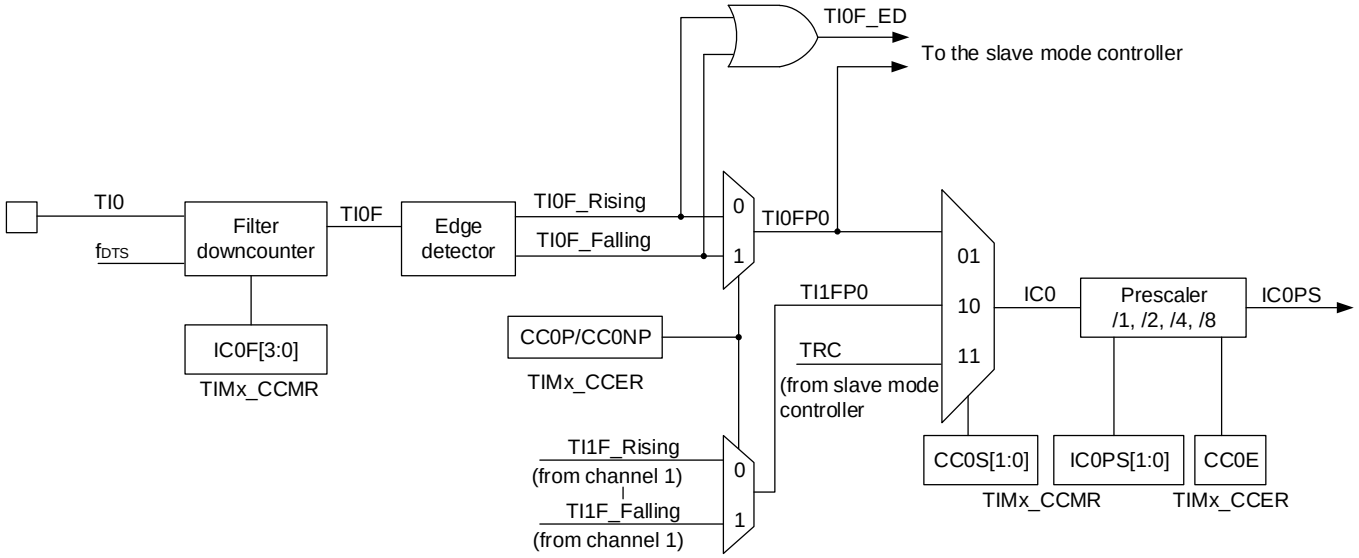


4.11.2.6 CAPTURE/COMPARE CHANNELS

Each Capture/Compare channel, is built around a capture/compare register (including a shadow register), an input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The following Figure 4.11-27 gives an overview of one Capture/Compare channel
The input stage samples the corresponding TIx input to generate a filter 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 4.11-27: Capture/compare channel (example: channel 0 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 4.11-28: Capture/compare channel 0 main circuit

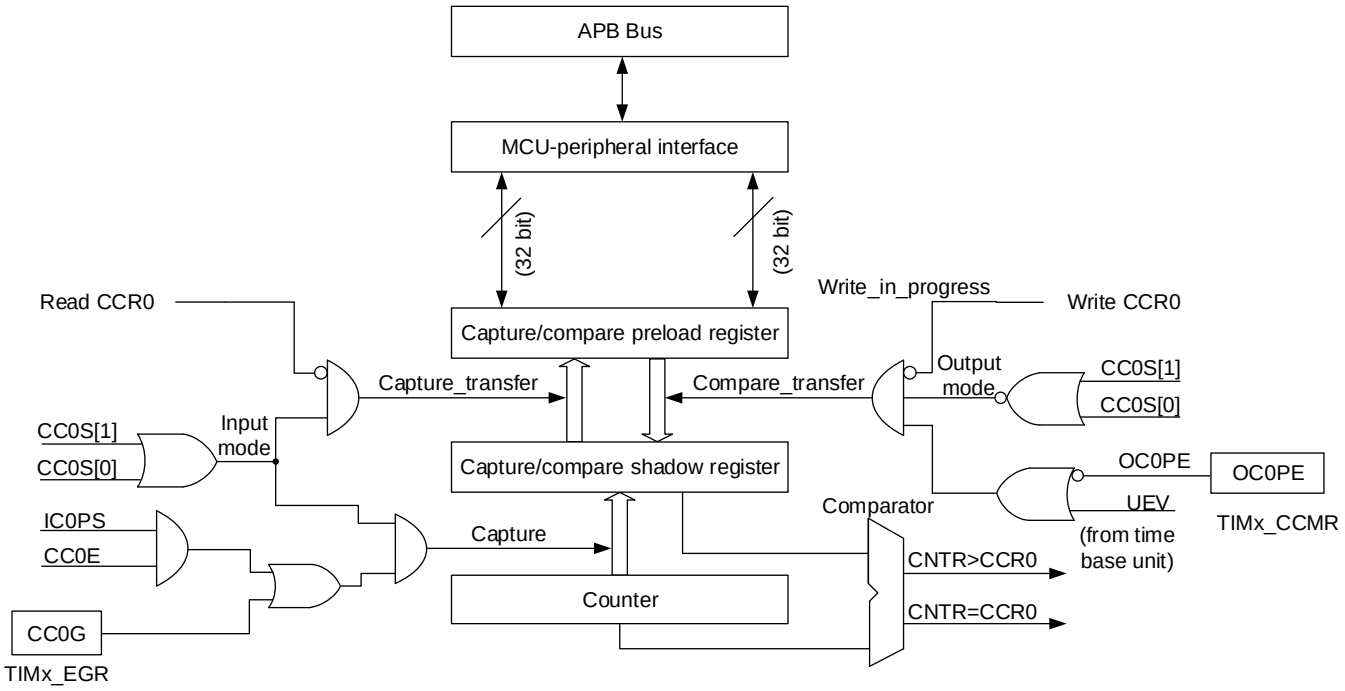
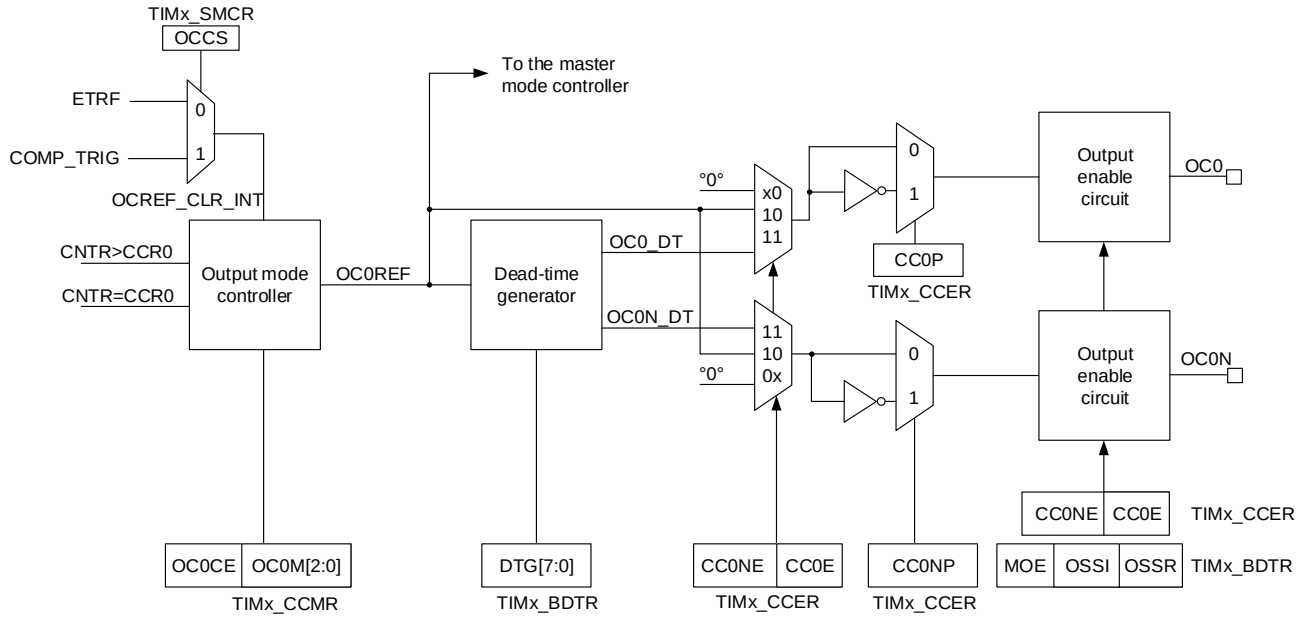


Figure 4.11-29: Output stage of capture/compare channel (channel 0 to 2)



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, 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.

4.11.2.7 INPUT CAPTURE MODE

In input capture mode, the Capture/Compare Registers (**TIMx_CCRx**) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding interrupt can be sent if they are enabled.

The following example shows how to capture the counter value in **TIMx_CCR0** when TI0 input rises. To do this, use the following procedure:

- Select the active input: **TIMx_CCR0** must be linked to the TI0 input, so write the CC0S bits to 01 in the **TIMx_CCMR** register. As soon as CC0S becomes different from 00, the channel is configured in input and the **TIMx_CCR0** 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_CCMR** register). Let's imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI0 when 8 consecutive samples with the new level have been detected (sampled at f_{DTS} frequency). Then write IC0F bits to 0011 in the **TIMx_CCMR** register.
- Select the edge of the active transition on the TI0 channel by writing *CC0P* and *CC0NP* bits to 0 in the **TIMx_CCER** register (rising edge in this case).
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC0PS bits to '00' in the **TIMx_CCMR** register).
- Enable capture from the counter into the capture register by setting the *CC0E* bit in the **TIMx_CCER** register.
- If needed, enable the related interrupt request by setting the CC0IE bit in the **TIMx_IER** register.

When an input capture occurs:

- The **TIMx_CCR0** register gets the value of the counter on the active transition.
- CC0IS status is set (interrupt status). CC0OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC0IE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture status. This is to avoid missing an overcapture which could happen after reading the status and before reading the data.
NOTE: ICx interrupt requests can be generated by software by setting the corresponding CCxG bit in the **TIMx_EGR** register.

4.11.2.8 PWM INPUT MODE

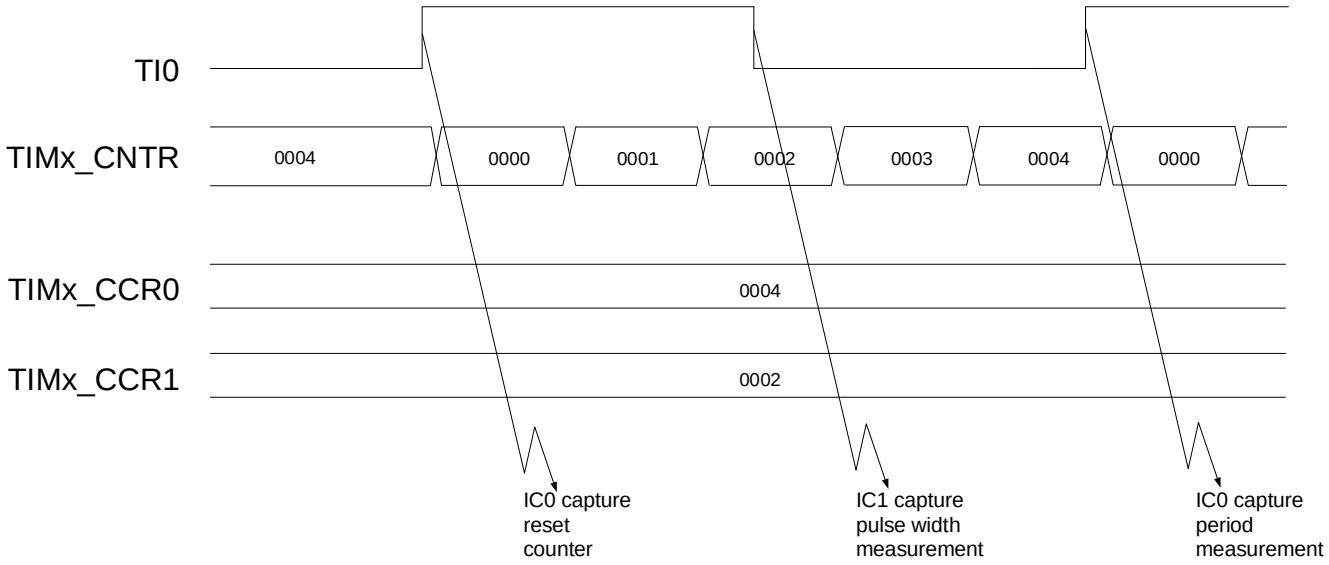
This mode is a particular case of input capture mode. The procedure is the same except:

- 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 **TIMx_CCR0** register) and the duty cycle (in **TIMx_CCR1** register) of the PWM applied on TI0 using the following procedure (depending on CK_INT frequency and prescaler value):

- Select the active input for **TIMx_CCR0**: write the CC0S bits to 01 in the **TIMx_CCMR** register (TI0 selected).
- Select the active polarity for TI0FP0 (used both for capture in **TIMx_CCR0** and counter clear): write the *CC0P* bit to '0' (active on rising edge).
- Select the active input for **TIMx_CCR1**: write the *CC1S* bits to 10 in the **TIMx_CCMR** register (TI0 selected).
- Select the active polarity for TI0FP1 (used for capture in **TIMx_CCR1**): write the *CC1P* bit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 111 in the **TIMx_SMCR** register (TI0FP0 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 CC0E and CC1E bits to '1' in the **TIMx_CCER** register.

Figure 4.11-30: PWM input mode timing



4.11.2.9 FORCED OUTPUT MODE

In output mode (CCxS bits = 00 in the TIMx_CCMR register), each output compare signal (OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison 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_CCMR 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 100 in the TIMx_CCMR register.

Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the status to be set. Interrupt can be sent accordingly. This is described in the output compare mode section below.

4.11.2.10 OUTPUT COMPARE MODE

In output mode (CCxS bits = 00 in the TIMx_CCMR register), each output compare signal (OCxREF and then
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_CCMR** register). 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.
- Generates an interrupt if the corresponding interrupt is enable (CCXIE bit in the **TIMx_IER** register).

The **TIMx_CCRx** registers can be programmed with or without preload registers using the *OCxPE* bit in the **TIMx_CCMR** register.

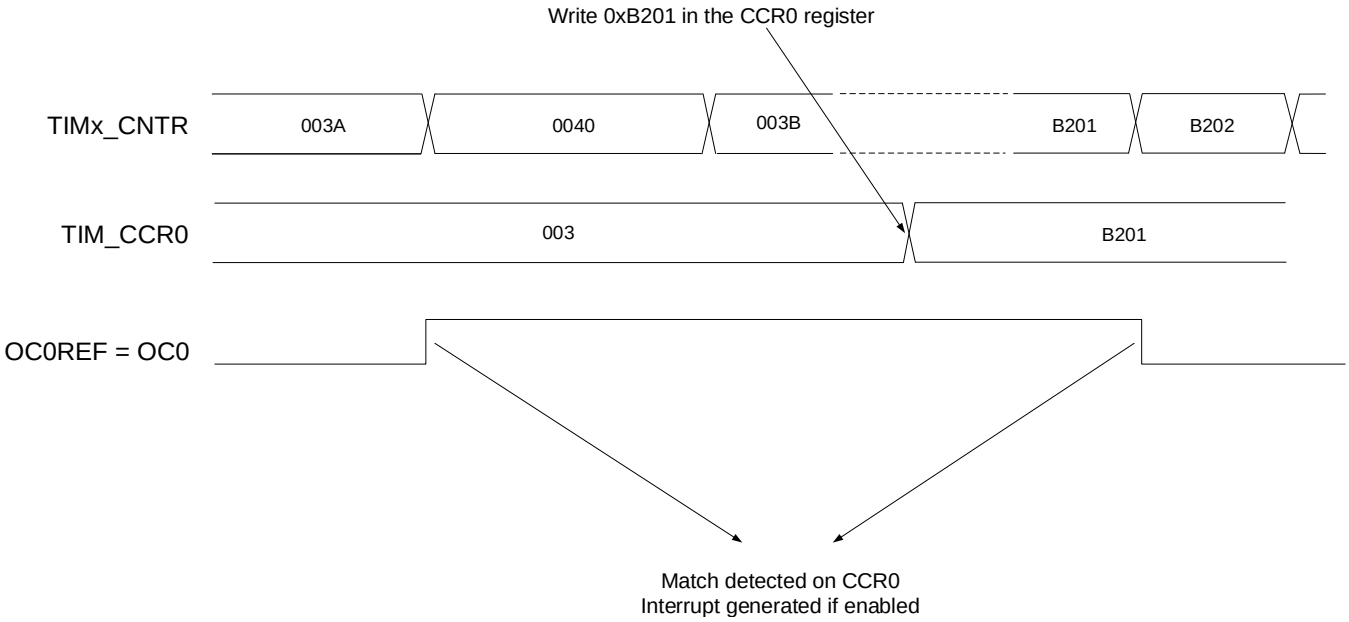
In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Shot mode).

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 in the TIMx_IER if an interrupt request is to be generated.
4. Select the output mode. For example:
   - Write OCxM = 011 to toggle OCx output pin when CNTR 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 by software to control the output waveform, provided that the preload register is not enabled (OCxPE='0', else **TIMx_CCRx** shadow register is updated only at the next update event UEV). An example is given in

Figure 4.11-31: Output compare mode, toggle on OC0



4.11.2.11 PWM MODE

Pulse Width Modulation mode allows you to generate a signal with a frequency determined by the value of the **TIMx_ARR** register and a duty cycle determined by the value of 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_CCMR** register. You must enable the corresponding preload register by setting the OCxPE bit in the **TIMx_CCMR** register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the *ARPE* bit in the **TIMx_CR1** register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, you have to initialize all the registers by setting the UG bit in the **TIMx_EGR** register.

OCx polarity is software programmable using the *CCxP* bit in the **TIMx_CCER** register. It can be programmed as active high or active low. OCx output is enabled by a combination of the *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 (1 or 2), **TIMx_CNTR** and **TIMx_CCRx** are always compared to determine whether **TIMx_CCRx ≤TIMx_CNTR** or **TIMx_CNTR ≤TIMx_CCRx** (depending on the direction of the counter).

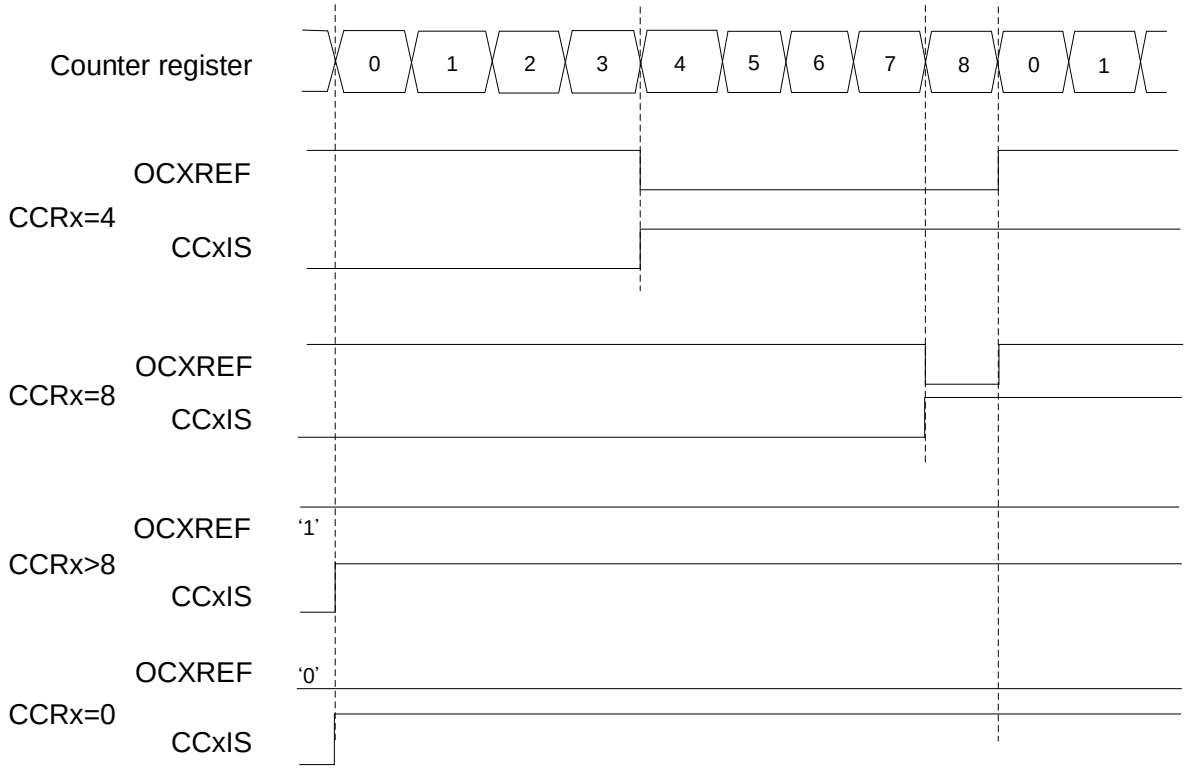
The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the *CMS* bits in the **TIMx_CR1** register.

PWM EDGE-ALIGNED MODE

- Upcounting configuration

Upcounting is active 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_CNTR < 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 4.11-32 shows some edge-aligned PWM waveforms in an example where **TIMx_ARR**=8.

Figure 4.11-32: Edge-aligned PWM waveforms (ARR=8)



- Downcounting configuration

Downcounting is active when DIR bit in **TIMx_CR1** register is high. In PWM mode 1, the reference signal OCxRef is low as long as **TIMx_CNTR** > **TIMx_CCRx** else it becomes 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.

PWM CENTER-ALIGNED MODE

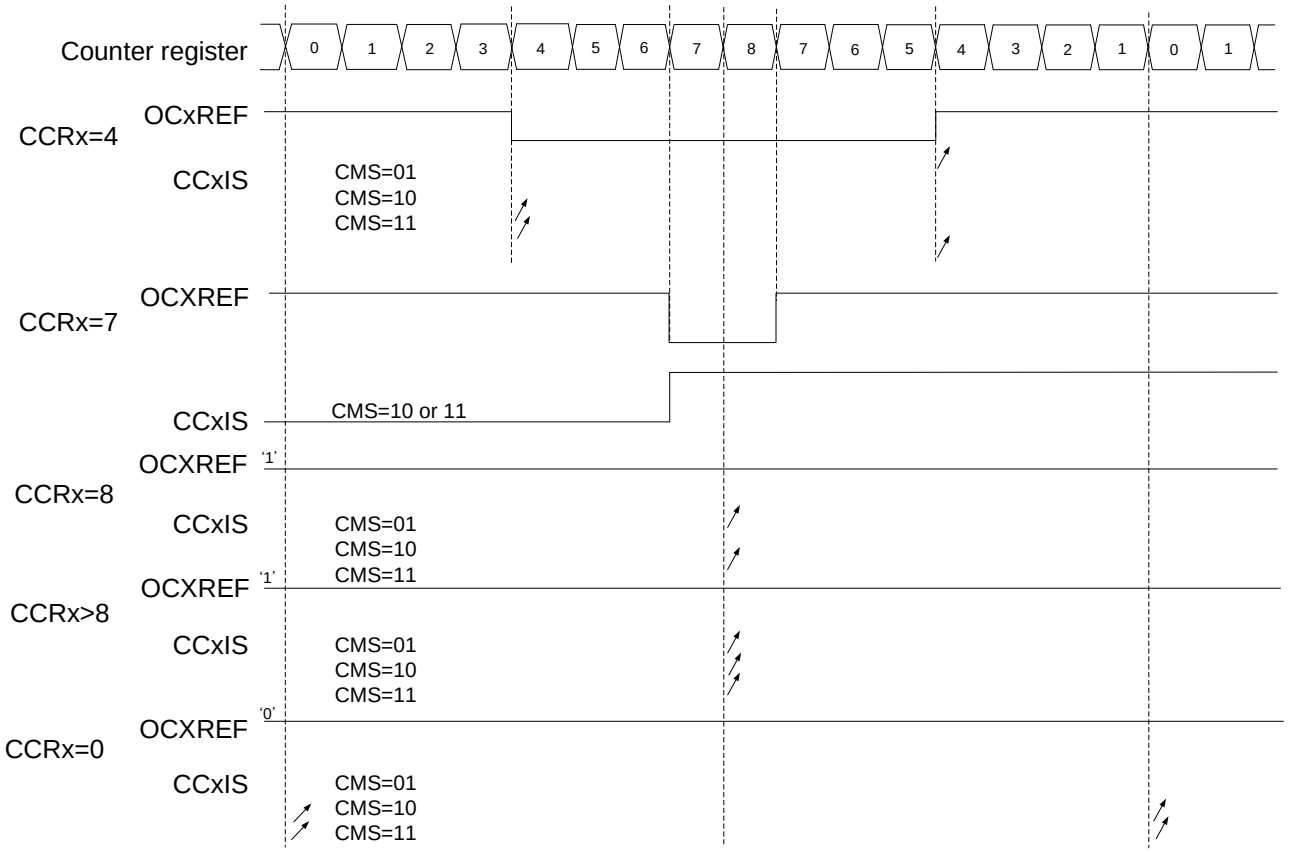
Center-aligned mode is active when the CMS bits in **TIMx_CR1** register are different from '00' (all the remaining configurations having the same effect on the OCxRef/OCx signals). The compare status is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the **TIMx_CR1** register is updated by hardware and must not be changed by software.

Figure 4.11-33 shows some center-aligned PWM waveforms in an example where:

- TIMx_ARR=8,
- PWM mode is the PWM mode 1,

The status is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS=01 in **TIMx_CR1** register..

Figure 4.11-33: Center-aligned PWM waveforms (ARR=8)



HINTS ON USING CENTER-ALIGNED MODE

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in 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.

- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:

- The direction is not updated if you write a value in the counter that is greater than the auto-reload value (TIMx_CNTR>TIMx_ARR). For example, if the counter was counting up, it continues to count up.
- The direction is updated if you write 0 or write the TIMx_ARR value in the counter but no Update Event UEV is generated.

- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the **TIMx_EGR** register) just before starting the counter and not to write the counter while it is running.

4.11.2.12 ONE-SHOT MODE

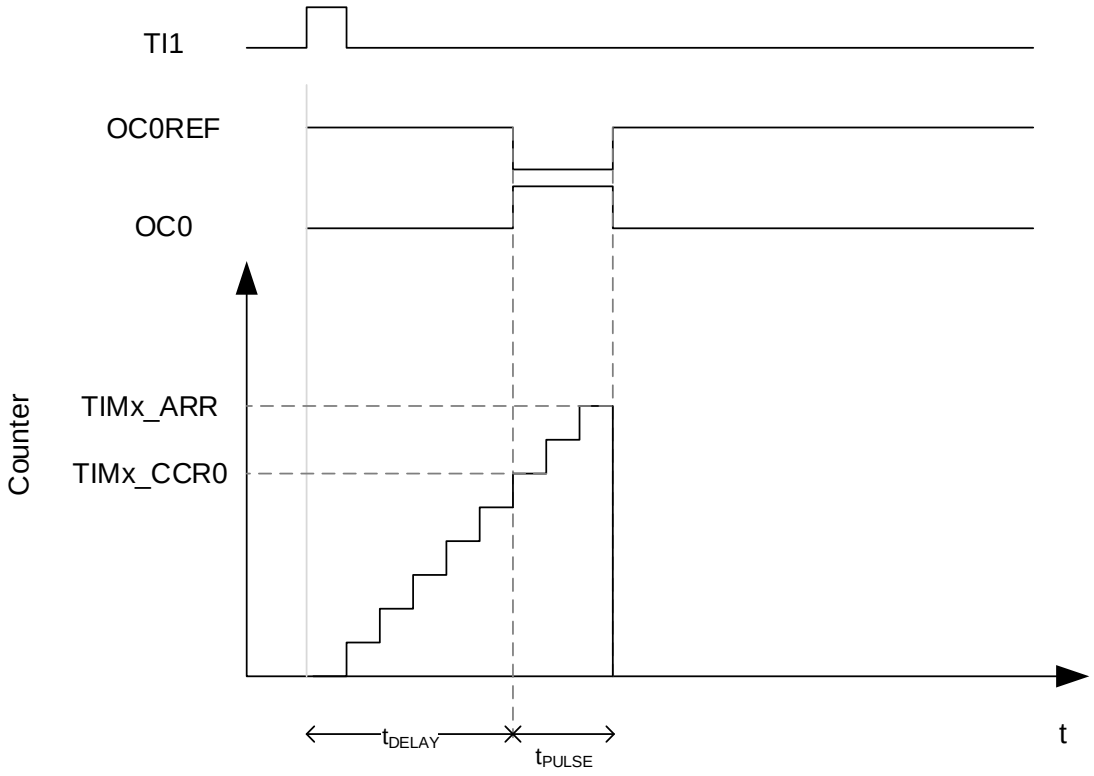
One-shot mode (OSM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. You select one-shot mode by setting the OSM 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:

- In upcounting: CNTR < CCRx ≤ ARR (in particular, 0 < CCRx)
- In downcounting: CNTR > CCRx

Figure 4.11-34: Example of one shot mode



For example you may want to generate a positive pulse on OC0 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI1 input pin.

Let's use TI1FP1 as trigger:

- Map TI1FP1 to TI1 by writing CC1S='01' in the **TIMx_CCMR** register.
- TI1FP1 must detect a rising edge, write CC1P='0' and CC1NP='0' in the **TIMx_CCER** register.
- Configure TI1FP1 as trigger for the slave mode controller (TRGI) by writing TS='1000' in the **TIMx_SMCR** register.
- TI1FP1 is used to start the counter by writing SMS to '110' in the **TIMx_SMCR** register (trigger mode).

The OSM 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 **TIMx_CCR0** register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (**TIMx_ARR** - **TIMx_CCR1**+1).
- Let's say you 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 you enable PWM mode 2 by writing OC0M=111 in the **TIMx_CCMR** register. You can optionally enable the preload registers by writing OC0PE='1' in the **TIMx_CCMR** register and ARPE in the **TIMx_CR1** register. In this case you have to write the compare value in the **TIMx_CCR0** 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 TI1. CC0P is written to '0' in this example.

In our example, the *DIR* and *CMS* bits in the **TIMx_CR1** register should be low.

You only want 1 shot (Single mode), so you write '1 in the OSM 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). When OSM bit in the **TIMx_CR1** register is set to '0', so the Repetitive Mode is selected.

4.11.2.13 COMPLEMENTARY OUTPUTS AND DEAD-TIME INSERTION

The advanced-control timers can output two complementary signals and manage the switching-off and the switching-on instants 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 activated by a combination of several control bits: the *CCxE* and *CCxNE* bits in the **TIMx_CCER** register and the *MOE*, *OISx*, *OISxN*, *OSSI* and *OSSR* bits in the **TIMx_BDTR** and **TIMx_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 (users can determine rising or falling delay as 8/4-bit by DTSEL in the **TIMx_BDTR**). 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=1, CCxNP=0, MOE=1, CCxE=1 and CCxNE=1 in these examples)

Figure 4.11-35: Complementary output with dead-time insertion

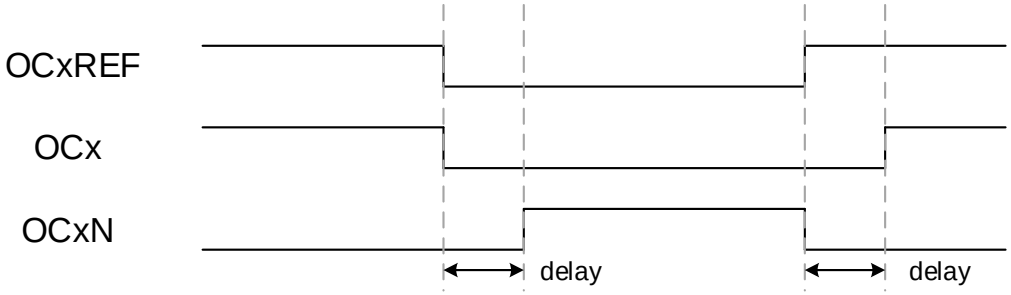


Figure 4.11-36: Dead-time waveforms with delay greater than the negative pulse

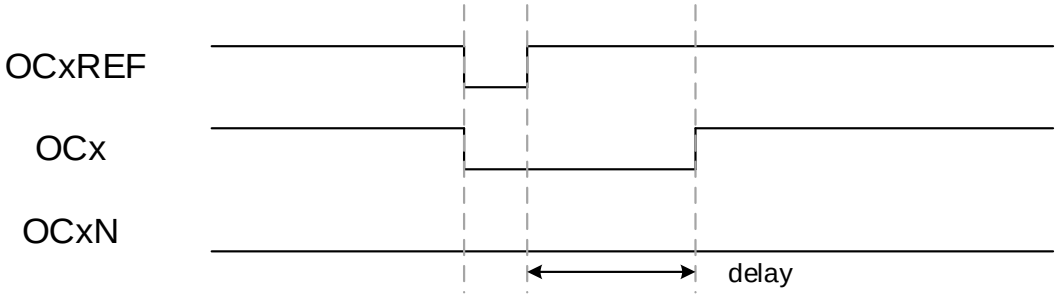
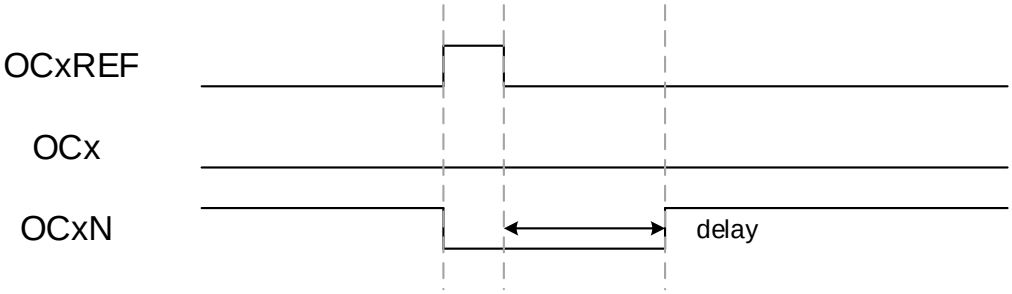


Figure 4.11-37: Dead-time waveforms with delay greater than the positive pulse



Re-directing OCxREF to OCx or OCxN

In output mode (forced, output compare or PWM), OCxREF can be re-directed to the OCx output or to OCxN output by configuring the *CCxE* and *CCxNE* bits in the **TIMx_CCER** register.

This allows you to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other alternative possibilities are to have both outputs at inactive level or both outputs active and complementary with dead-time.

NOTE: When only OCxN is enabled (CCxE=0, CCxNE=1), it is not complemented and becomes active as soon as OCxREF is high. For example, if CCxNP=1 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.

4.11.2.14 USING THE BREAK FUNCTION

When using the break function, the output enable signals and inactive levels are modified according to additional control bits (*MOE, OSSI* and *OSSR* bits in the **TIMx_BDTR** register, *OISx* and *OISxN* bits in the **TIMx_CR2** register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time. Refer to Table 4.11-3: Output control bits for complementary OCx and OCxN channels with break feature more details.

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 1 APB clock period to correctly read back the bit after the write operation.

Because MOE falling edge can be asynchronous, 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 you write the asynchronous signal and read 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 OSSI 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 OSSI=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. This is done asynchronously so that it works even if no clock is provided to the timer.
  - If OSSI=1, MOE=0 and 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 that because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 ck_tim clock cycles).
  - If OSSI=0, MOE=0 then the timer releases the enable outputs else the enable outputs become low level.
- The break status (BIS bit in the TIMx_ISC register) is set. An interrupt can be generated if the BIE bit in the TIMx_IER register is set.
- 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 you write it to '1' again. In this case, it can be used for security and you can connect the break input to an alarm from power drivers, thermal sensors or any security components.

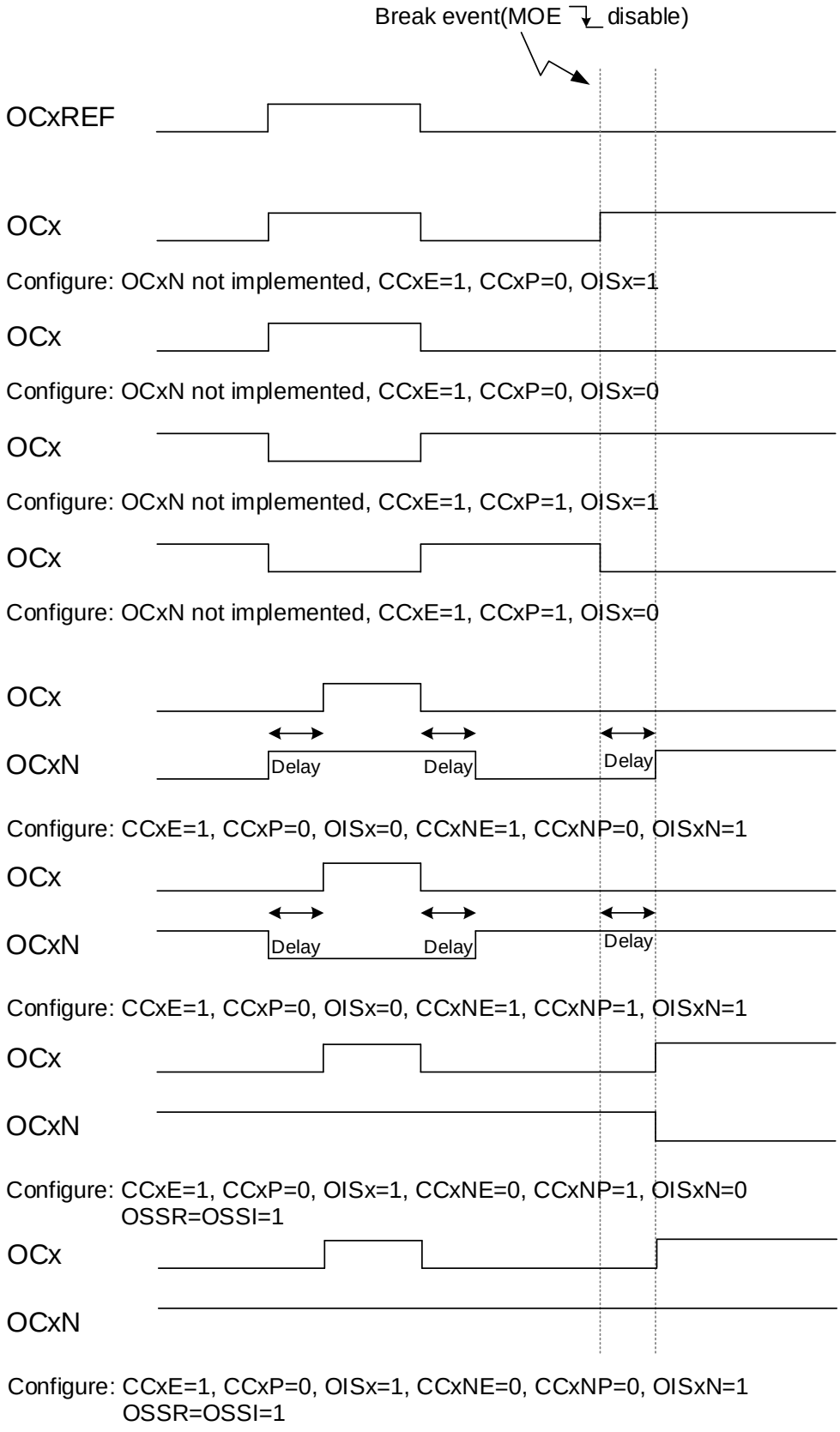
NOTE: The break inputs is acting on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status BIS cannot be cleared.

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the **TIMx_BDTR** Register.

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 you to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). You can choose from 3 levels of protection selected by the LOCK bits in the **TIMx_BDTR** register. Refer to Section 16.4.18: TIM1 break and dead-time register (**TIMx_BDTR**). The LOCK bits can be written only once after an MCU reset.

The Figure 4.11-38 shows an example of behavior of the outputs in response to a break.

Figure 4.11-38: Output behavior in response to a break



4.11.2.15 CLEARING THE OCXREF SIGNAL ON AN EXTERNAL EVENT

The OCxREF signal of a given channel can be cleared when a high level is applied on the OCREF_CLR_INPUT (OCxCE enable bit in the corresponding TIMx_CCMR register set to 1). OCxREF remains low until the next update event (UEV) occurs. This function can only be used in Output compare and PWM modes. It does not work in Forced mode.

OCREF_CLR_INPUT can be selected between the COMP_TRIG input and ETRF (ETR after the filter) by configuring the OCCS bit in the TIMx_SMCR register.

When ETRF is chosen, ETR must be configured as follows:

The OCxREF signal for a given channel can be driven low by applying a high level to the ETRF input (OCxCE enable bit of the corresponding TIMx_CCMR register set to '1'). The OCxREF signal remains low until the next update event, UEV, occurs.

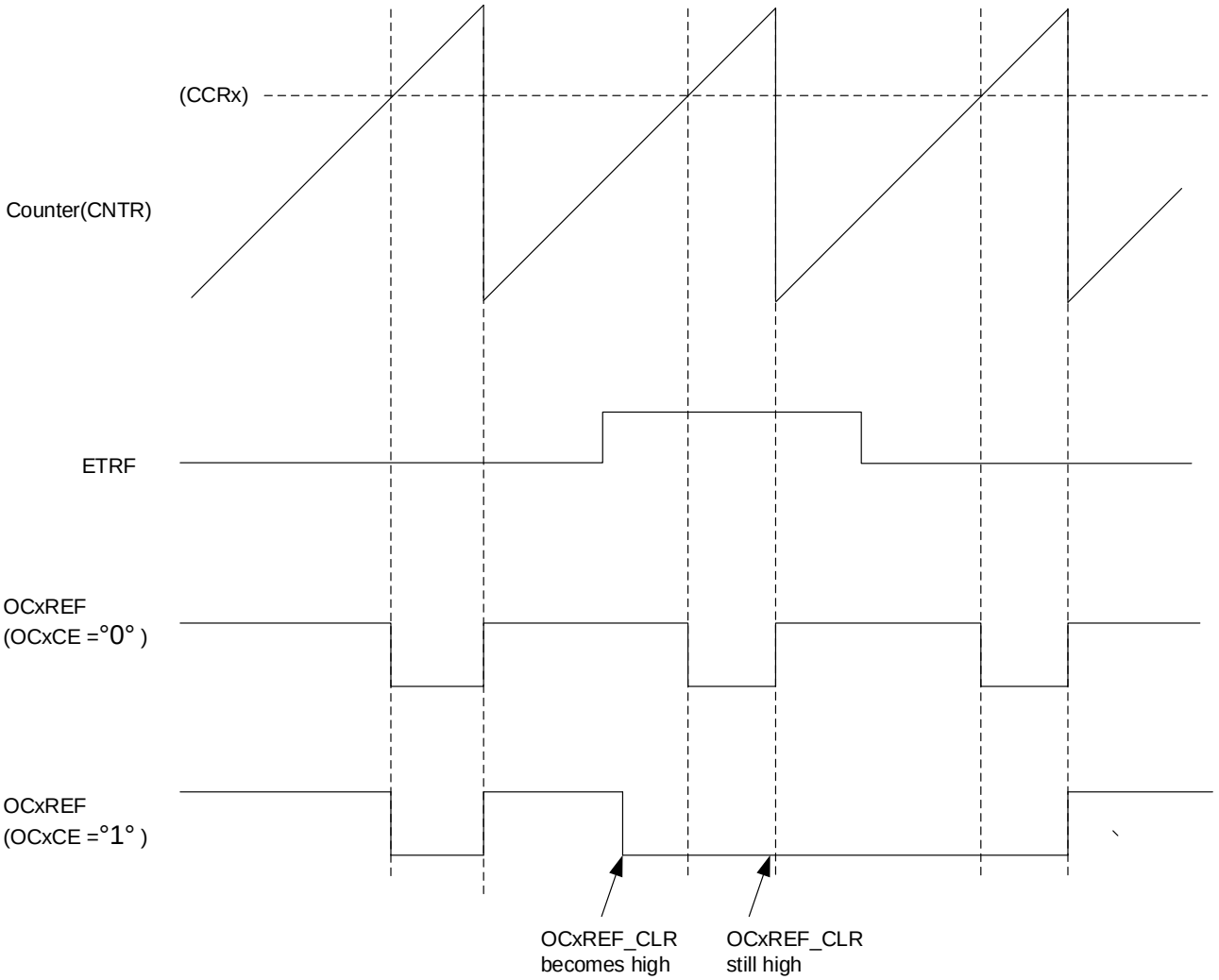
This function can only be used in output compare and PWM modes, and does not work in forced mode.

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follow:

1. The External Trigger Prescaler should be kept off: bits ETPS[1:0] of the TIMx_SMCR register set to '00'.
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 according to the user needs.

Figure 11-39 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 4.11-39: Clearing TIMx OCxREF



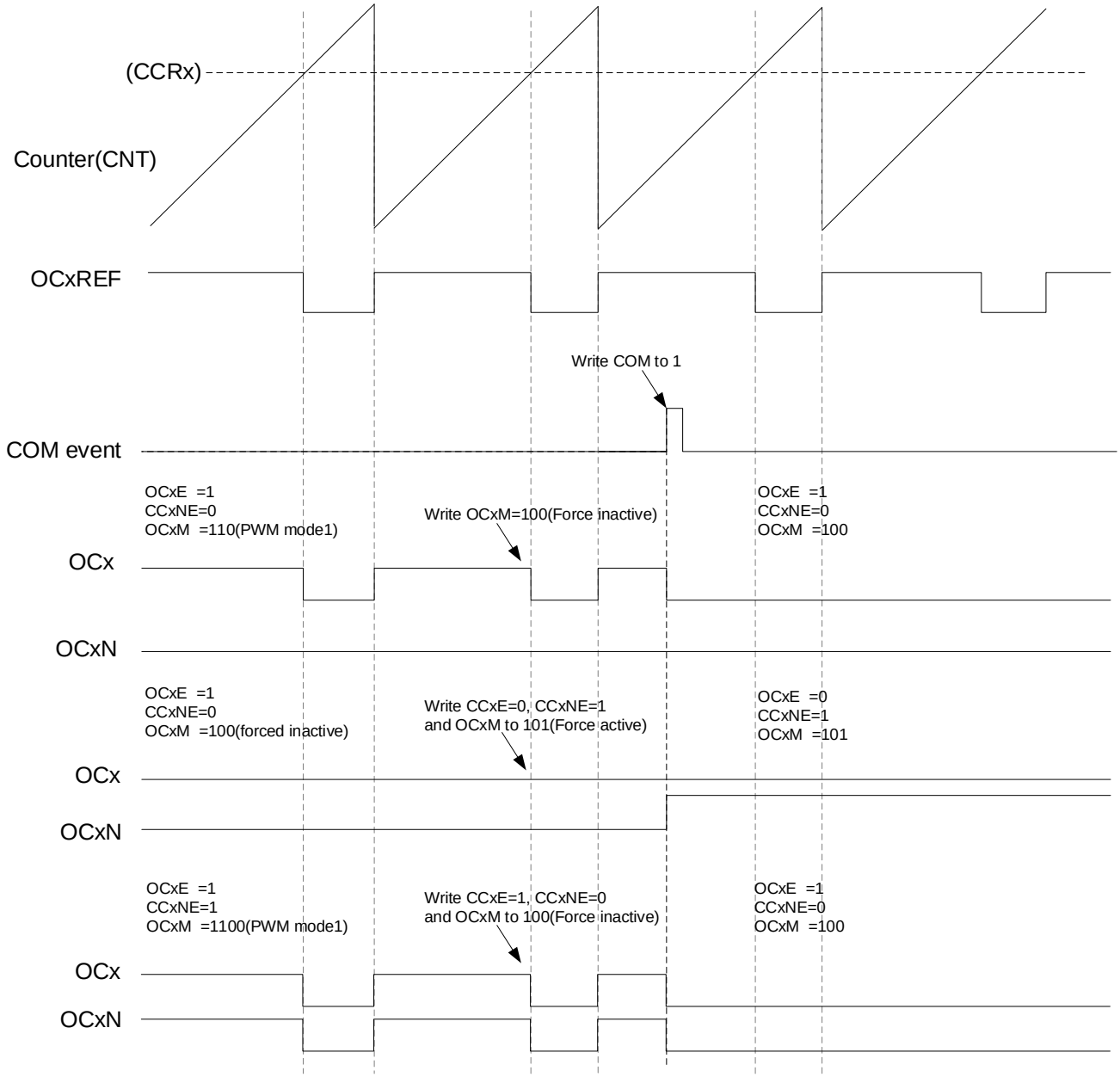
Note: In case of a PWM with a 100% duty cycle (if CCRx>ARR), then OCxREF is enabled again at the next counter overflow.

4.11.2.16 6-STEP PWM GENERATION

OCxM, *CCxE* and *CCxNE* bits. The preload bits are transferred to the shadow bits at the COM commutation event. Thus you can program in advance the configuration for the next step and change the configuration of all the channels at the same time. COM can be generated by software by setting the *COM* bit in the **TIMx_EGR** register or by hardware (on TRGI rising edge).

Figure 4.11-40 describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations.

Figure 4.11-40: 6-step generation, COM example



4.11.2.17 ENCODER INTERFACE MODE

To select Encoder Interface mode write SMS='001' in the **TIMx_SMCR** register if the counter is counting on TI0 edges only, SMS='010' if it is counting on TI1 edges only and SMS='011' if it is counting on both TI10 and TI1 edges.

The two inputs TI0 and TI1 are used to interface to an incremental encoder. The counter is clocked by each valid transition on TI0FP0 or TI1FP1 (TI0 and TI1 after input filter and polarity selection, TI0FP0=TI0 if not filtered and not inverted, TI1FP1=TI1 if not filtered and not inverted) assuming that it is enabled (CEN bit in **TIMx_CR1** register written to '1'). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the **TIMx_CR1** register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI0 or TI1), whatever the counter is counting on TI0 only, TI1 only or both TI0 and TI1.

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 you must configure **TIMx_ARR** before starting. In the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal.

In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor. The table summarizes the possible combinations, assuming TI0 and TI1 don't switch at the same time.

Table 4.11-1: Counting direction versus encoder signals

Active edge	Level on opposite signal (TI0FP0 for TI2, TI1FP1 for TI1)	TI0FP0 signal		TI1FP1 signal	
		Rising	Falling	Rising	Falling
Counting on TI0 only	High	Down	Up	No Count	No Count
	Low	Up	Down	No Count	No Count
Counting on TI1 only	High	No Count	No Count	Up	Down
	Low	No Count	No Count	Down	Up
Counting on TI0 and TI1	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be used to convert the encoder's differential outputs to digital signals. This greatly increases 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.

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:

- CC0S=’01’ (**TIMx_CCMR** register, TI0FP0 mapped on TI0).
- CC1S=’01’ (**TIMx_CCMR** register, TI1FP1 mapped on TI1).
- SMS=’011’ (**TIMx_SMCR** register, both inputs are active on both rising and falling edges).
- DIR’0’ (**TIMx_CR1** register, set upcountung)
- CEN=’1’ (**TIMx_CR1** register, Counter enabled).

Figure 4.11-41: Example of counter operation in encoder interface mode (set DIR='0', before CEN=1)

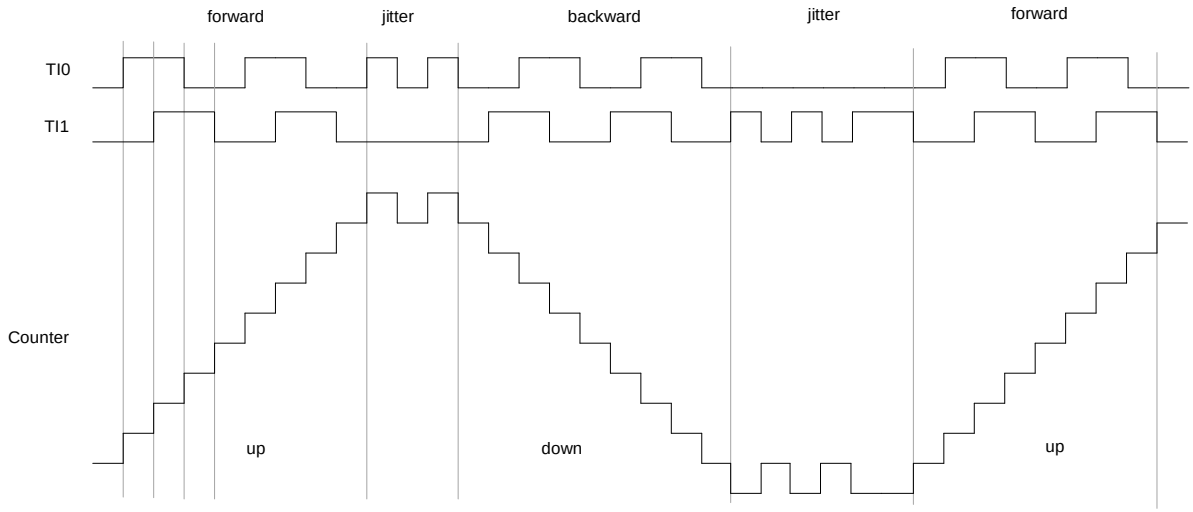
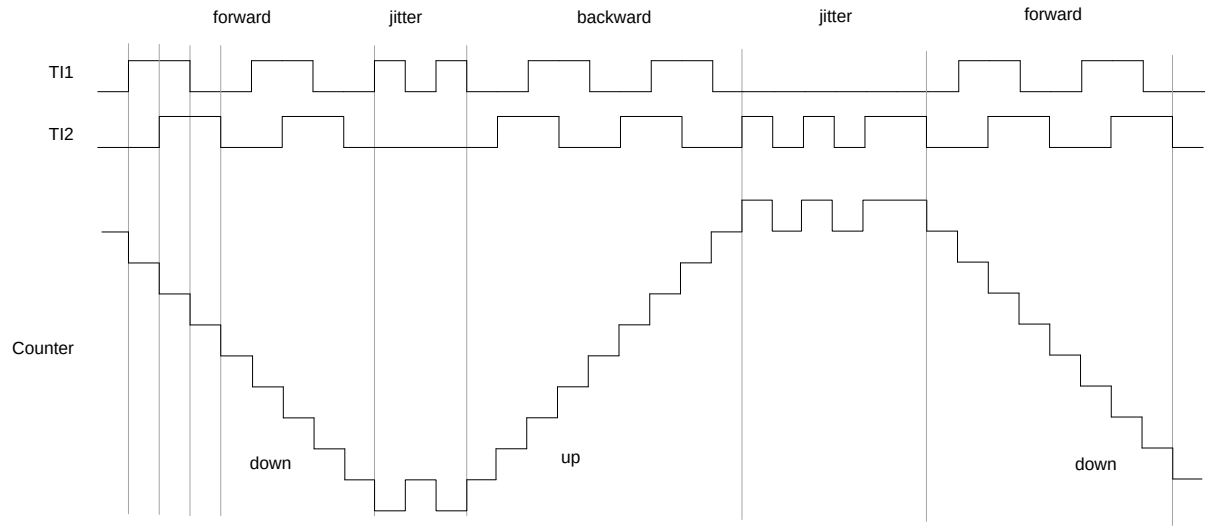


Figure 4.11-42: Example of encoder interface mode with (set DIR='1', before CEN=1)



The timer, when configured in Encoder Interface mode provides information on the sensor's current position. You can obtain dynamic information (speed, acceleration, and 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. You can do this 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).

4.11.2.18 TIMER INPUT XOR FUNCTION

The TI0S bit in the **TIMx_CR2** register, allows the input filter of channel 0 to be connected to the output of a XOR gate, combining the three input pins TIMx_CH0, TIMx_CH1 and TIMx_CH2.

The XOR output can be used with all the timer input functions such as trigger or input capture. An example of this feature used to interface Hall sensors is given in below section.

4.11.2.19 INTERFACING WITH HALL SENSORS

This is done using the advanced-control timers (TIM4/5) to generate PWM signals to drive the motor and another timer TIMx (TIM0/1/2/3) referred to as “interfacing timer” in

The “interfacing timer” captures the 3 timer input pins (CC0, CC1, CC2) connected through a XOR to the TI0 input channel (selected by setting the TI0S bit in the **TIMx_CR2** register).

The slave mode controller is configured in reset mode; the slave input is TI0F_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.

On the “interfacing timer”, capture/compare channel 0 is configured in capture mode, capture signal is TRC. The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

The “interfacing timer” can select compare pulse which changes the configuration of the channels of the advanced-control timer (TIM4/5) (by triggering a COM event). The TIM0 timer is used to output compare pulse to drive the motor. To do this, the interfacing timer channel must be programmed so that a positive pulse is generated after a programmed delay (in output compare or PWM mode). This pulse is sent to the advanced-control timer (TIM4/5) through the TRGO output.

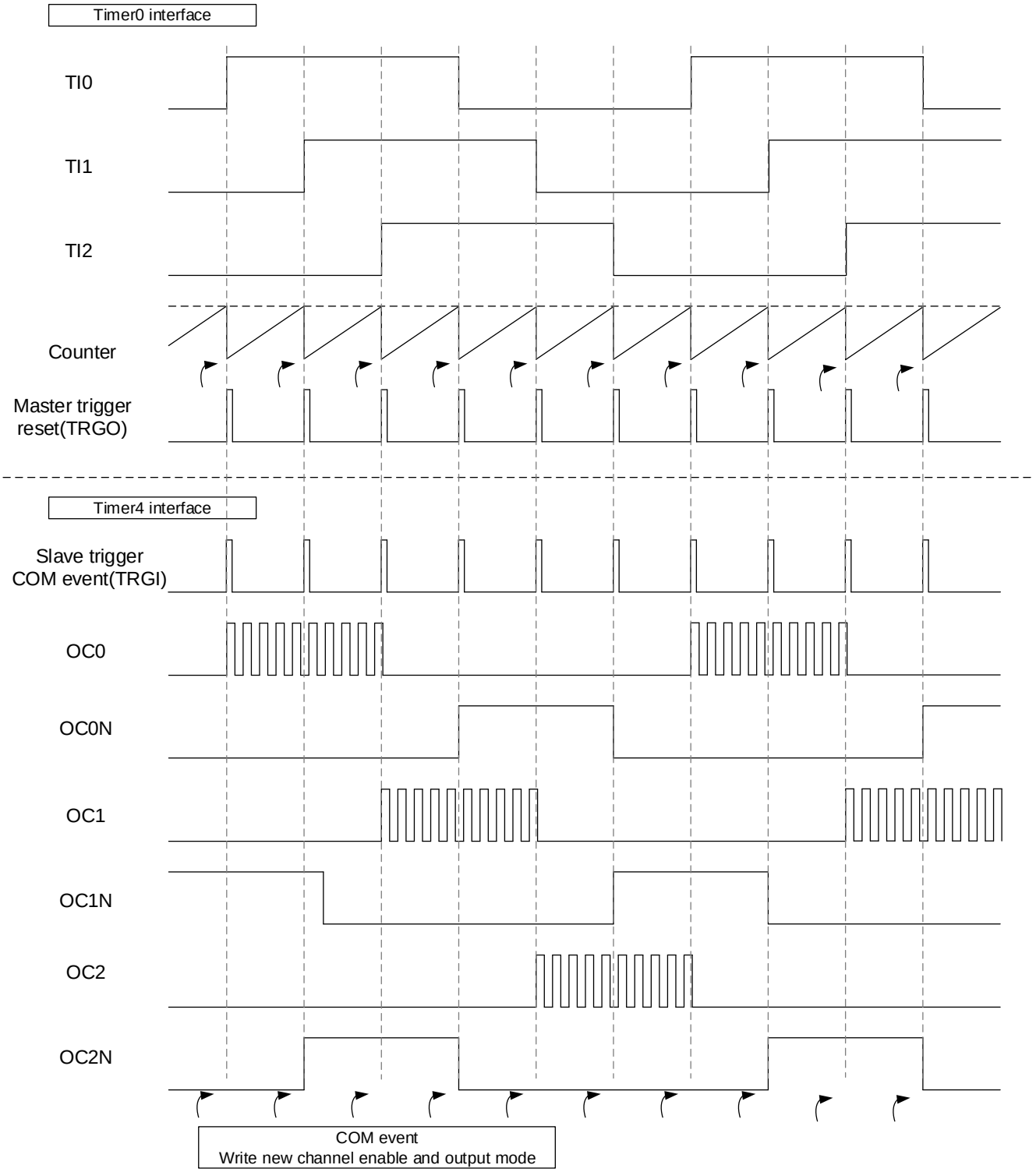
Example: you want to change the PWM configuration of your advanced-control timer TIM4 after a programmed delay each time a change occurs on the Hall inputs connected to one of the TIMx timers.

Master timer setting:

- Configure 3 timer inputs XORed to the TI0 input channel by writing the TI0S bit in the **TIMx_CR2** register to ‘1’
- Program the time base: write the **TIMx_ARR** to the max value (the counter must be cleared by the TI0 change. Set the prescaler to get a maximum counter period longer than the time between 2 changes on the sensors
- Configure the counter to be in the slave reset mode: write the SMS bit in the **TIMx_SMCR** register to 100
- Configure TI0FP0 as slave trigger: write the TS bit in the **TIMx_SMCR** register to 0111
- Configure TI0 compare pulse as master trigger(TRGO): write the MMS bit in the **TIMx_CR2** register to 011
- Slave timer setting:
- Program slave trigger (TRGI) select timer0: write the TS bits in the **TIMx_SMCR** register to ‘0000.
- Program channel 0 in PWM 1 mode with the desired delay: write the OC0M bits to ‘110’ and the CC0S bits to ‘00’ in the **TIMx_CCMR**1 register,
- Enable the CCxE, CCxNE and OCxM preload function: write the CCPC bit in the **TIMx_CR2** register to 1
- Select the preload function are updated by setting the COMG bit or when an rising edge occurs on TRGI: write the CCUS bit in the **TIMx_CR2** register to 1

In the advanced-control timer TIM4, the right ITR input must be selected as trigger input, the timer is programmed to generate PWM signals, the capture/compare control signals are preloaded (CCPC=1 in the **TIMx_CR2** register) and the COM event is controlled by the trigger input (CCUS=1 in the **TIMx_CR2** register). The PWM control bits (CCxE, OCxM) are written after a COM event for the next step (this can be done in an interrupt subroutine generated by the rising edge of TRGI).

Figure 4.11-43: Example of hall sensor interface



4.11.2.20 TIMER AND EXTERNAL TRIGGER SYNCHRONIZATION

The TIMx timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

SLAVE MODE: RESET MODE

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the UDIS bit in the TIMx_CR1 register is low, an update event UEV is generatedThen all the preloaded registers (**TIMx_ARR**, **TIMx_CCRx**) are updated.

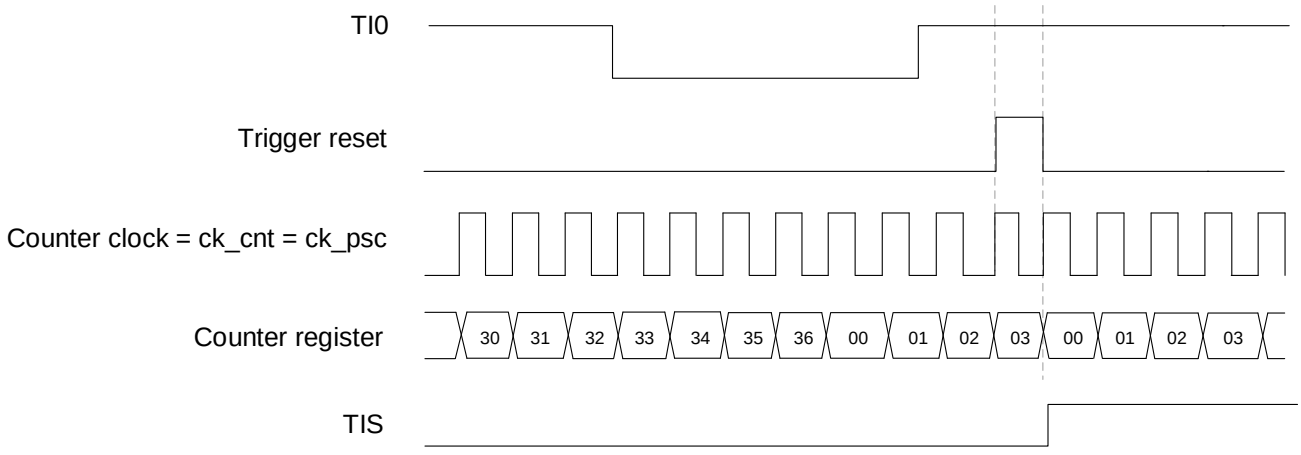
In the following example, the upcounter is cleared in response to a rising edge on TI0 input:

- Configure the channel 0 to detect rising edges on TI0. Configure the input filter duration (in this example, we don't need any filter, so we keep IC0F=0000). The capture prescaler is not used for triggering, so you don't need to configure it. The CC0S bits select the input capture source only, CC0S = 01 in the **TIMx_CCMR** register. Write CC0P=0 and CC0NP='0' in **TIMx_CCER** register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=100 in **TIMx_SMCR** register. Select TI0 as the input source by writing TS=0111 in **TIMx_SMCR** register.
- Start the counter by writing CEN=1 in the **TIMx_CR1** register

The counter starts counting on the internal clock, then behaves normally until TI0 rising edge. When TI0 rises, the counter is cleared and restarts from 0. In the meantime, the trigger status is set (TIS bit in the TIMx_ISC register) if enabled (depending on the TIE bits in TIMx_IER register).

The following figure shows this behavior when the auto-reload register **TIMx_ARR**=0x36. The delay between the rising edge on TI0 and the actual reset of the counter is due to the resynchronization circuit on TI0 input.

Figure 4.11-44: Control circuit in reset mode



SLAVE MODE: GATED MODE

The counter can be enabled depending on the level of a selected input.

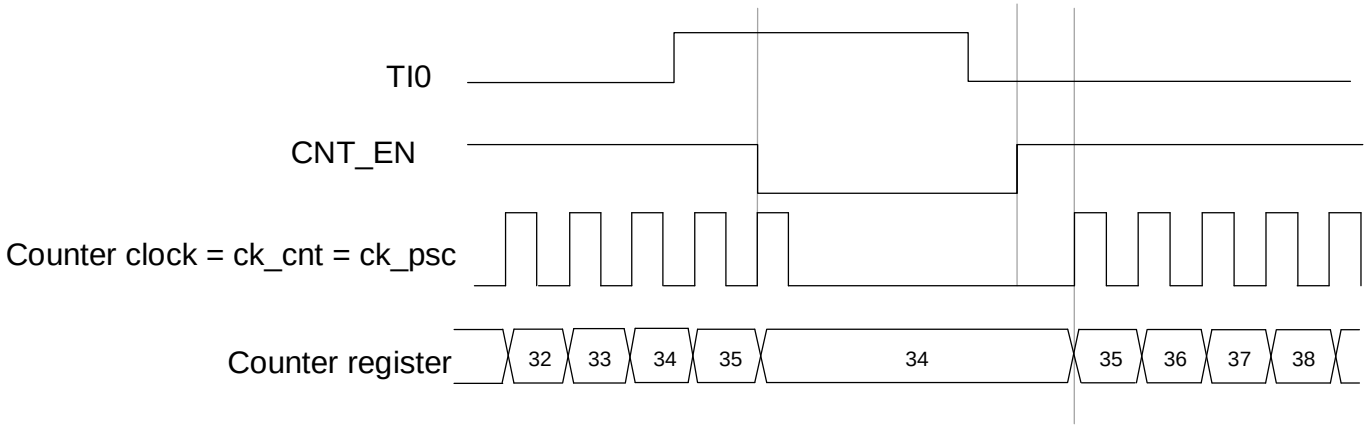
In the following example, the upcounter counts only when TI0 input is low:

- Configure the channel 0 to detect low levels on TI0. Configure the input filter duration (in this example, we don't need any filter, so we keep IC0F=0000). The capture prescaler is not used for triggering, so you don't need to configure it. The CC0S bits select the input capture source only, CC0S=01 in **TIMx_CCMR** register. Write CC0P=1 and CC0NP='0' in **TIMx_CCER** register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS=101 in **TIMx_SMCR** register. Select TI0 as the input source by writing TS=0111 in **TIMx_SMCR** register.
- Enable the counter by writing CEN=1 in the **TIMx_CR1** register (in gated mode, the counter doesn't start if CEN=0, whatever is the trigger input level).

The counter starts counting on the internal clock as long as TI0 is low and stops as soon as TI0 becomes high. The TIS status in the TIMx_ISC register is not set in this mode.

The delay between the rising edge on TI0 and the actual stop of the counter is due to the resynchronization circuit on TI0 input.

Figure 4.11-45: Control circuit in gated mode



SLAVE MODE: TRIGGER MODE

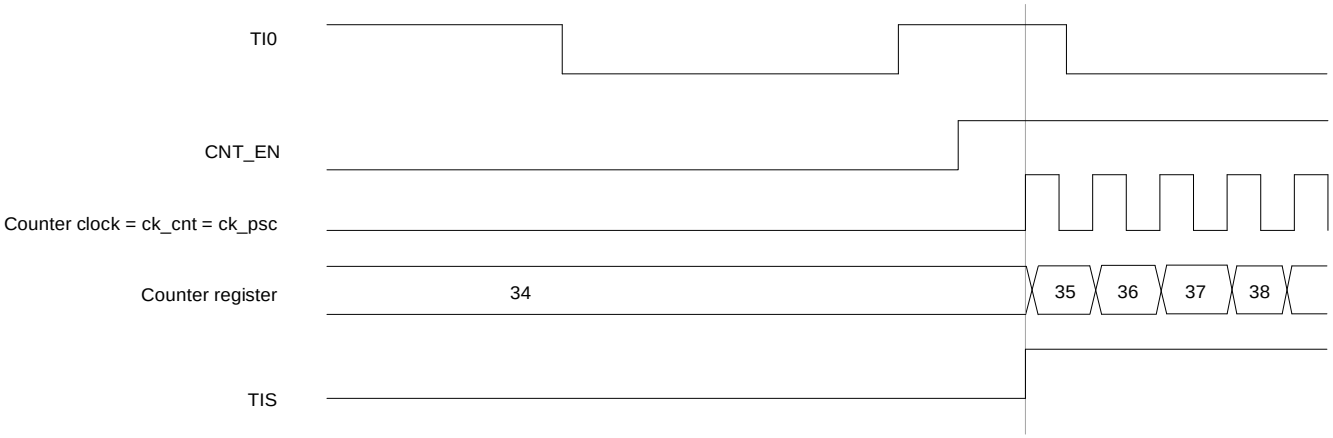
The counter can start in response to an event on a selected input.

In the following example, the upcounter starts in response to a rising edge on TI0 input:

- Configure the channel 0 to detect rising edges on TI0. Configure the input filter duration (in this example, we don't need any filter, so we keep IC0F=0000). The capture prescaler is not used for triggering, so you don't need to configure it. The CC0S bits are configured to select the input capture source only, CC0S=01 in **TIMx_CCMR** register. Write CC0P=0 and CC0NP=0 in **TIMx_CCER** register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in **TIMx_SMCR** register. Select TI0 as the input source by writing TS=0111 in **TIMx_SMCR** register.
- Enable the counter by writing CEN=1 in the **TIMx_CR1** register (in trigger mode, the counter doesn't start if CEN=0, whatever is the trigger input level)

When a rising edge occurs on TI0, the counter starts counting on the internal clock and the TIS status is set.

Figure 4.11-46: Control circuit in trigger mode



4.11.2.21 TIMER SYNCHRONIZATION

The timers can be internally connected together for timer chaining or synchronization. This can be implemented by configuring one timer to operate in the master mode while configuring another timer to be in the slave mode. The following figures present several examples of trigger selection for the master and slave modes.

Figure 4.11-47: Master/Slave timer example

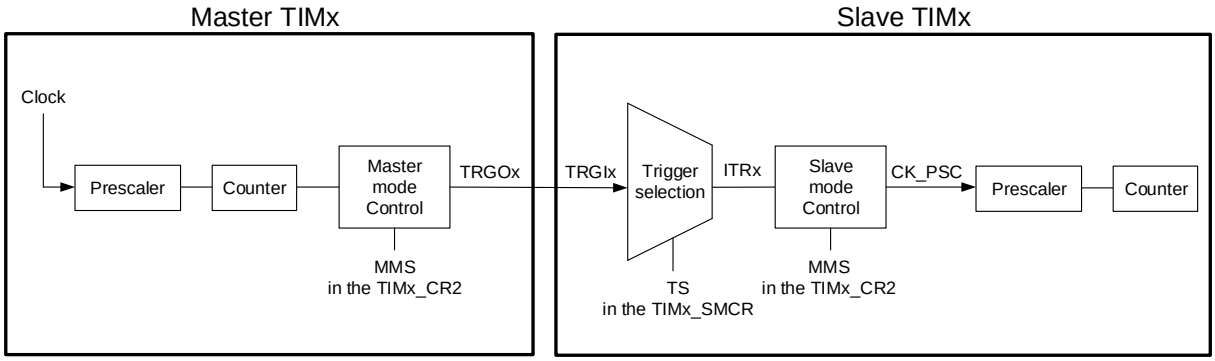
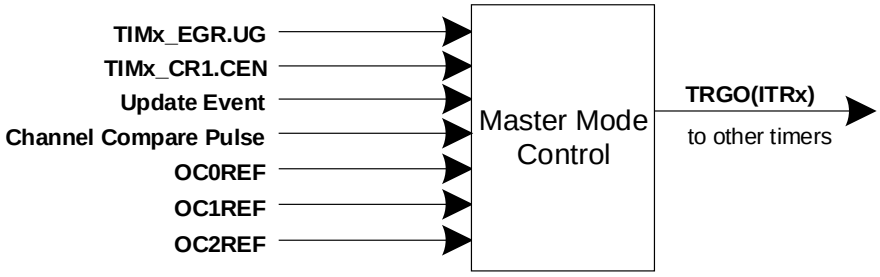


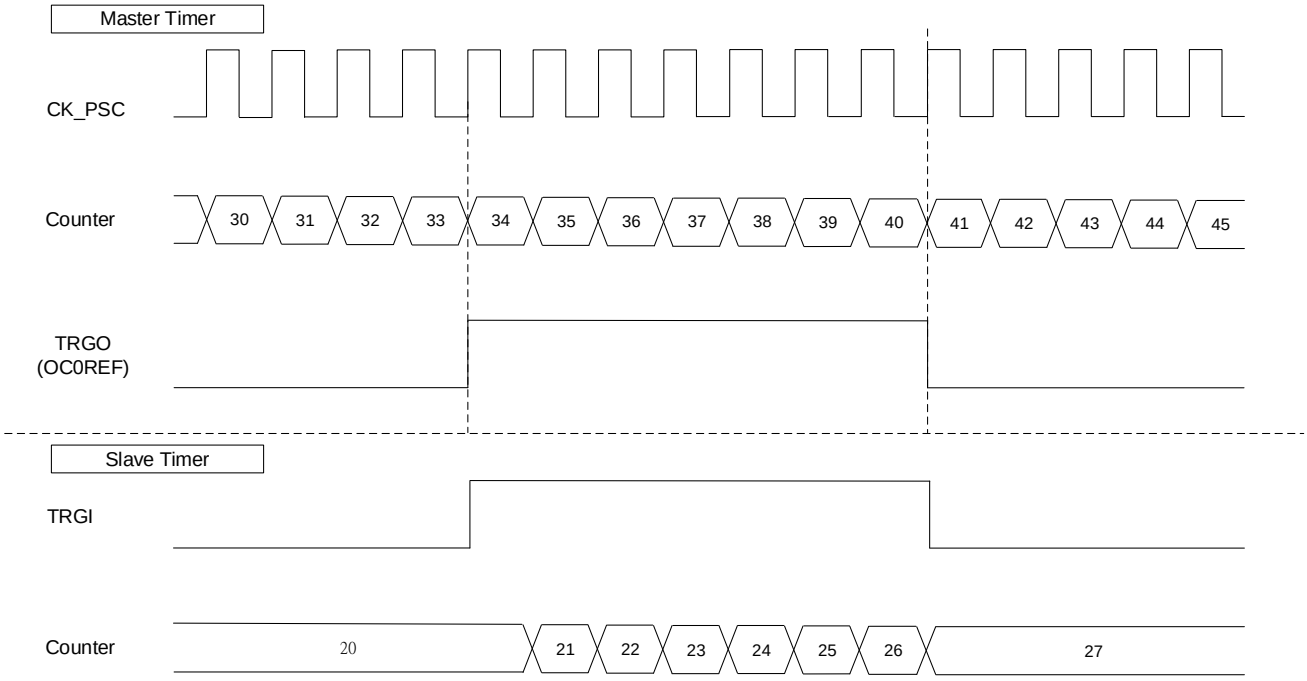
Figure 4.11-48: Master Selection



Using one timer to trigger another timer to start or stop counting

- Configure TIM0 to be in the master mode and to send its channel 0 Output Reference signal OC0REF as a trigger output: write MMS bit in the TIM0_CR2 register to 100
- Configure TIM0 OC0REF waveform
- Configure the TIM4 to receive its input trigger source from the TIM0 trigger output: write TS bit in the TIM4_SMCR register to 0000
- Configure TIM4 to operate in the slave gated mode: write SMS bit in the TIM4_SMCR register to 101
- Enable TIM4: write CEN bit in the TIM0_CR1 register to 1
- Enable TIM0: write CEN bit in the TIM4_CR1 register to 1

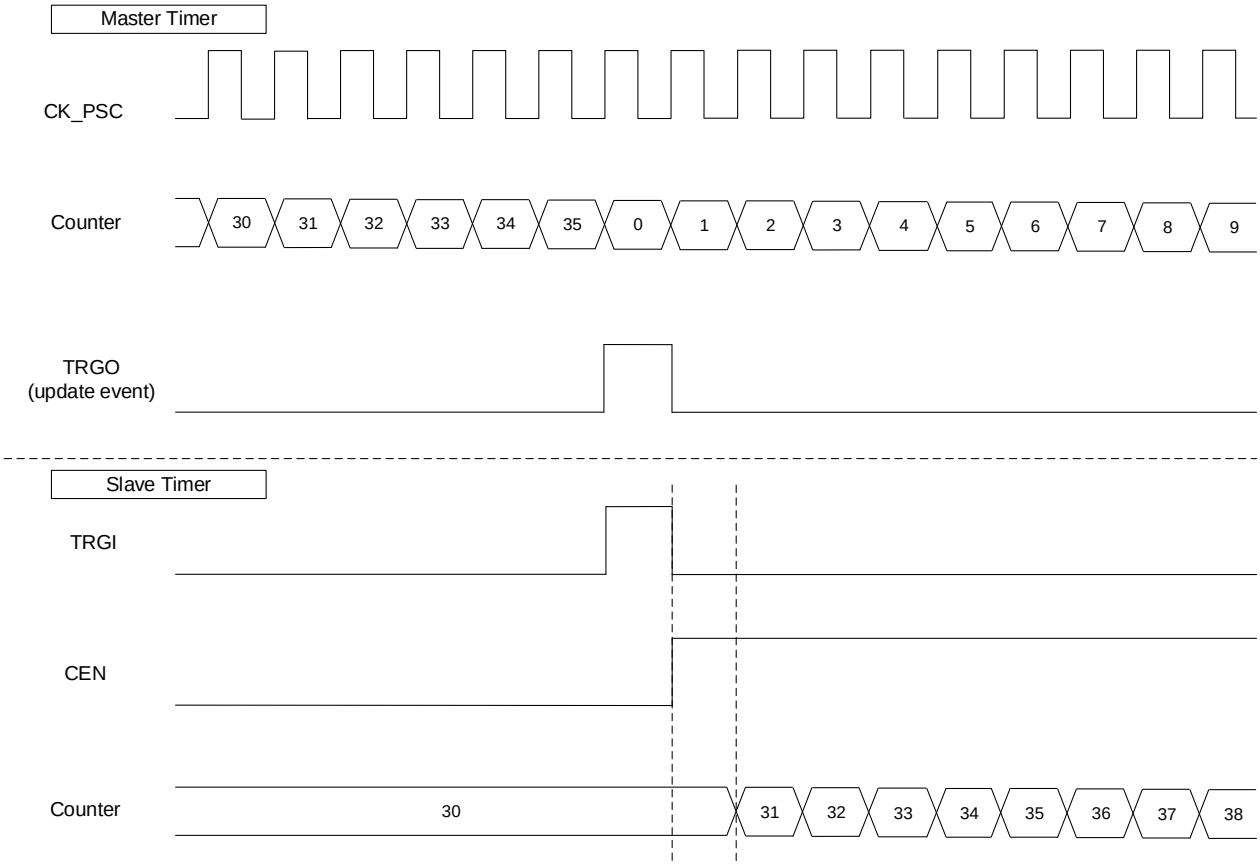
Figure 4.11-49: Gating Slave timer using Master Timer OC0REF



Using one timer update event to trigger another timer to start or stop counting

- Configure TIM0 to be in the master mode and to send its update event as a trigger output: write MMS bit in the TIM0_CR2 register to 010
- Configure the TIM1 to receive its input trigger source from the TIM0 trigger output: write TS bit in the TIM4_SMCR register to 0000
- Configure TIM4 to operate in the slave trigger mode: write SMS bit in the TIM4_SMCR register to 110
- Configure TIM0 and TIM4 counter set
- Only enable TIM0: write CEN bit in the TIM0_CR1 register to 1

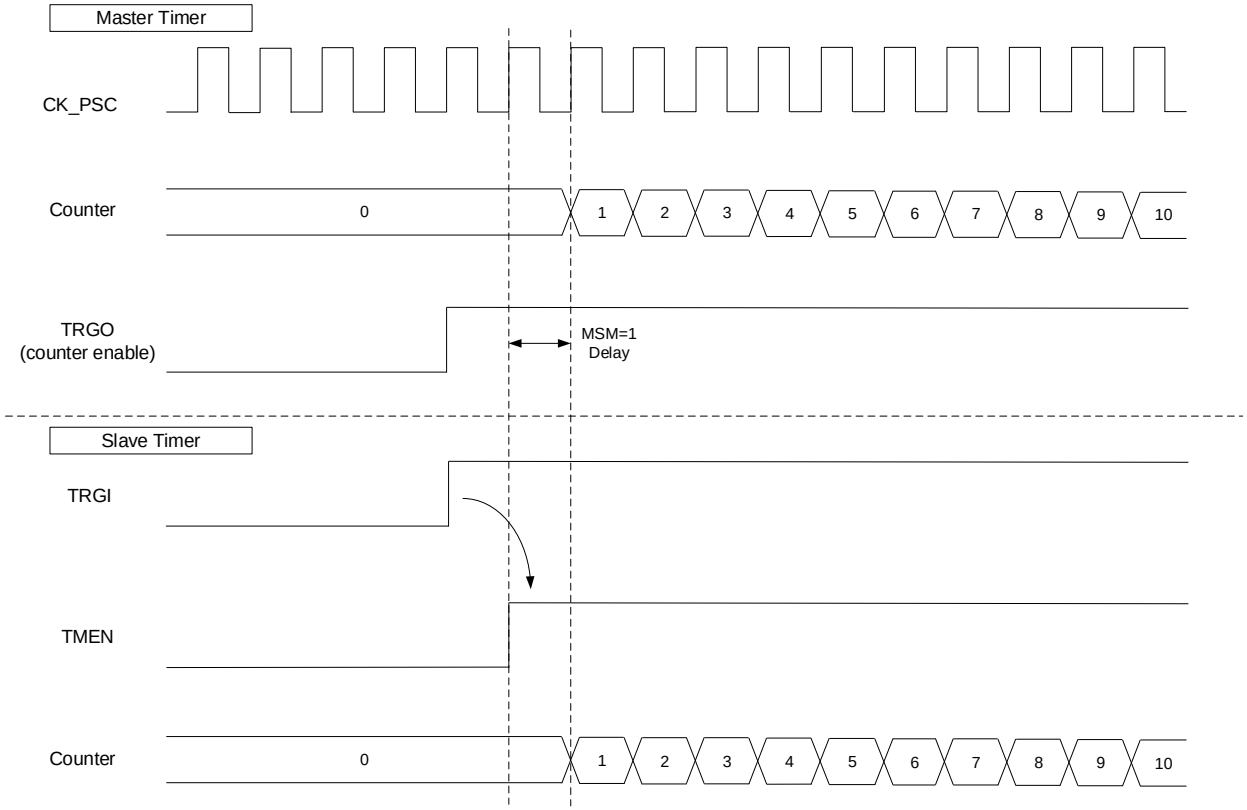
Figure 4.11-50: Triggering Slave timer using Master Timer update event



Using one timer to trigger another timer to start or stop counting

- Configure TIM0 to be in the master mode and to send its enable as a trigger output: write MMS bit in the TIM0_CR2 register to 001
- Use MSM bit in the TIM0_SMCR register to generate one APB clock delay to synchronize several timers
- Configure TIM0 to operate in the slave trigger mode and trigger source is TI0: write SMS and TS bit in the TIM0_SMCR register to 110 and 0111
- Configure the TIM4 to receive its input trigger source from the TIM0 trigger output: write TS bit in the TIM4_SMCR register to 0000
- Configure TIM4 to operate in the slave trigger mode: write SMS bit in the TIM4_SMCR register to 110
- Configure TIM0 and TIM4 counter set

Figure 4.11-51: Triggering Slave timer using Master Timer counter enable

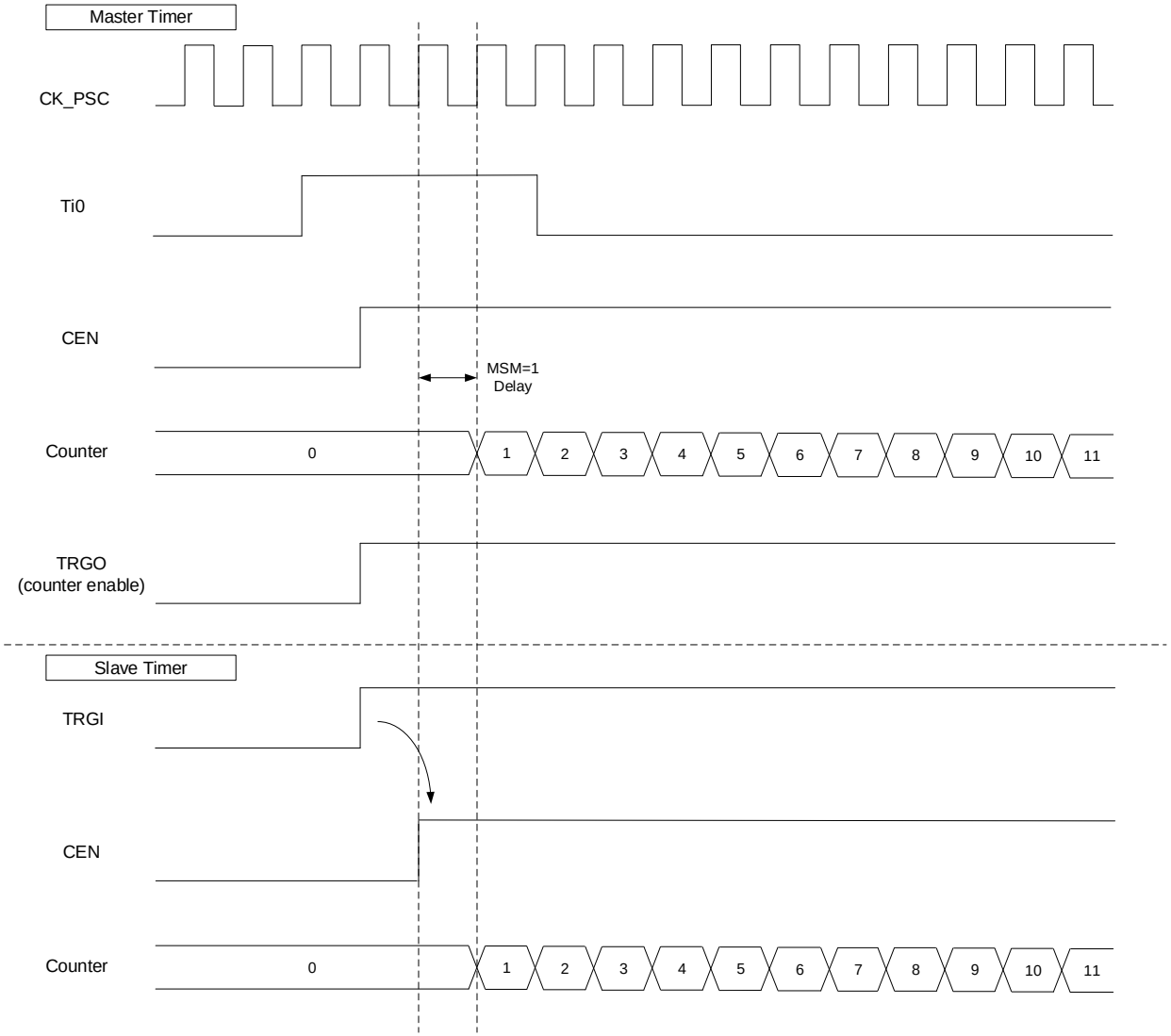


Using an external trigger to synchronously two timers

- Configure TIM0 and TIM4 counter set
- Configure the TIM4 to receive its input trigger source from the TIM0 trigger output: write TS bit in the TIM4_SMCR register to 0000
- Configure TIM4 to operate in the slave trigger mode: write SMS bit in the TIM4_SMCR register to 110
- Configure TIM0 to be in the master mode and to send its enable as a trigger output: write MMS bit in the TIM0_CR2 register to 001
- Configure TIM0 slave mode to receive its input trigger source from TIM0_CH0 pin: write CC0S bit in the TIM0_CCMR register to 01 and TS bit in the TIM0_SMCR register to 0111

Use MSM bit in the TIM0_SMCR register to generate one APB clock delay to synchronize several timers.

Figure 4.11-52: Synchronously two timers using external trigger



4.11.2.22 DEBUG MODE

When the microcontroller enters debug mode (Cortex™-M0 core halted), the TIMx counter stops count.

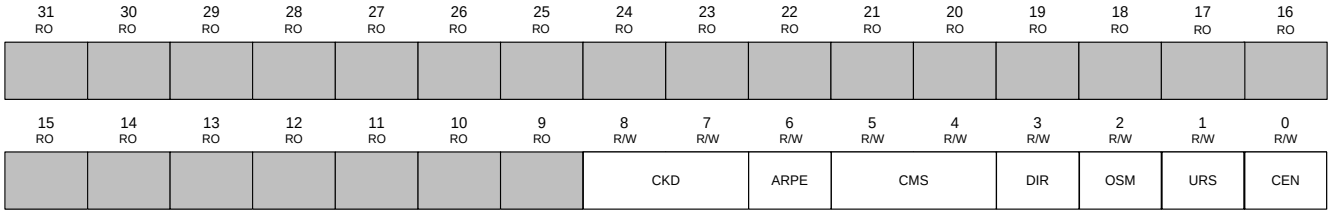
4.11.3 TIMER REGISTER MAP

Base address:

- Timer 4: 0x4000_0400
- Timer 5: 0x4000_0500

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	TIMx_CR1	R/W	0x0000_0000	TIMx Control Register 1	288
0x0004	TIMx_CR2	R/W	0x0000_0000	TIMx Control Register 2	290
0x0008	TIMx_SMCR	R/W	0x0000_0000	TIMx Slave Mode Control Register	292
0x000C	TIMx_EGR	WO	0x0000_0000	TIMx Event Generation Register	295
0x0010	TIMx_IER	WO	0x0000_0000	TIMx Interrupt Enable Register	296
0x0014	TIMx_IDR	WO	0x0000_0000	TIMx Interrupt Disable Register	297
0x0018	TIMx_IMR	RO	0x0000_0000	TIMx Interrupt Mask Register	298
0x001C	TIMx_RIS	RO	0x0000_0000	TIMx Raw Interrupt Status	299
0x0020	TIMx_ISC	R/W1C	0x0000_0000	TIMx Interrupt Status and Clear	300
0x0024	TIMx_RCR	R/W	0x0000_0000	TIMx Repetition Counter Register	301
0x0028	TIMx_CCMR	R/W	0x0000_0000	TIMx Capture/Compare Mode Register	302
0x002C	TIMx_CCER	R/W	0x0000_0000	TIMx Capture/Compare Enable Register	306
0x0030	TIMx_PSC	R/W	0x0000_0000	TIMx Prescaler	310
0x0034	TIMx_ARR	R/W	0x0000_0000	TIMx Auto-Reload Register	310
0x0038	TIMx_CNTR	R/W	0x0000_0000	TIMx Counter Register	311
0x003C	TIMx_CCR0	R/W	0x0000_0000	TIMx Capture/Compare Register 0	311
0x0040	TIMx_CCR1	R/W	0x0000_0000	TIMx Capture/Compare Register 1	312
0x0044	TIMx_CCR2	R/W	0x0000_0000	TIMx Capture/Compare Register 2	312
0x0048	TIMx_CCR3	R/W	0x0000_0000	TIMx Capture/Compare Register 3	313
0x004C	TIMx_ADCTS	R/W	0x0000_0000	TIMx ADC Trigger Select	313
0x0050	TIMx_BDTR	R/W	0x0000_0000	TIMx Break and Dead-Time Register	314

4.11.3.1 TIMX_CR1 - TIM 4 / 5 CONTROL REGISTER 1

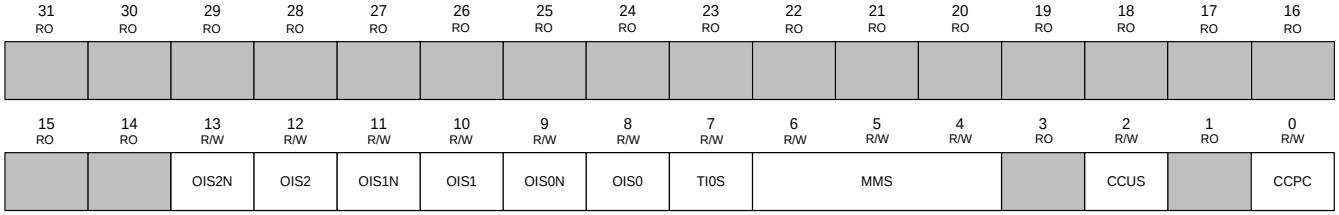


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:9	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
8:7	CKD	R/W	0x0	Clock Division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock (t_{DTS})used by the dead-time generators and the digital filters (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 program this value
6	ARPE	R/W	0	Auto-Reload Preload Enable 0: **TIMx_ARR** register is not buffered 1: **TIMx_ARR** register is buffered
5:4	CMS	R/W	0x0	Center-Aligned Mode Select 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt status of channels configured in output (CCxS=00 in **TIMx_CCMR** register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt status of channels configured in output (CCxS=00 in **TIMx_CCMR** register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt status of channels configured in output (CCxS=00 in **TIMx_CCMR** register) are set both when the counter is counting up or down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1).
3	DIR	R/W	0	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
2	OSM	R/W	0	One Shot Mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the CEN bit)

Bits	Name	Type	Reset	Description
1	UDIS	R/W	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).
0	CEN	R/W	0	Counter Enable 0: Counter disabled 1: Counter enabled 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.

4.11.3.2 TIMX_CR2 - TIM 4 / 5 CONTROL REGISTER 2

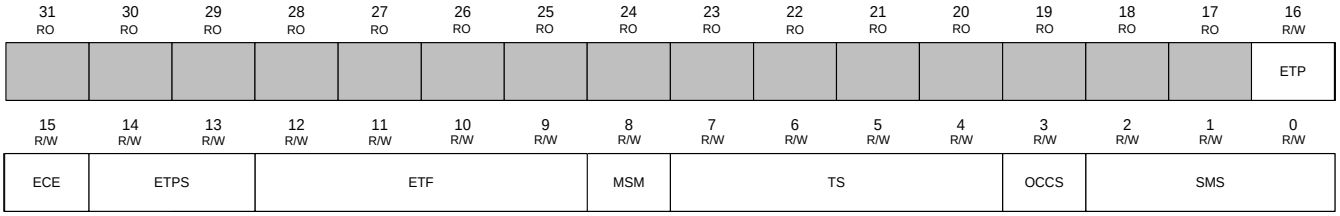


Offset: 0x0004

Bits	Name	Type	Reset	Description
31: 14	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
13	OIS2N	R/W	0	Output Idle state 2 (OC2N output) Refer to OIS0N bit
12	OIS2	R/W	0	Output Idle state 2 (OC2 output) Refer to OIS0 bit
11	OIS1N	R/W	0	Output Idle state 1 (OC1N output) Refer to OIS0N bit.
10	OIS1	R/W	0	Output Idle state 1 (OC1 output) Refer to OIS0 bit
9	OIS0N	R/W	0	Output Idle state 0 (OC0N output) 0: OC0N=0 after a dead-time when MOE=0 1: OC0N=1 after a dead-time when MOE=0 Note: This bit cannot be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in **TIMx_BDTR** register).
8	OIS0	R/W	0	Output Idle state 0 (OC0 output) 0: OC0=0 (after a dead-time if OC0N is implemented) when MOE=0 1: OC0=1 (after a dead-time if OC0N is implemented) when MOE=0 Note: This bit cannot be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in **TIMx_BDTR** register).
7	TI0S	R/W	0	TI0 Selection 0: The TIMx_CH0 pin is connected to TI0 input 1: The TIMx_CH0, CH1 and CH2 pins are connected to the TI0 input (XOR combination)
6:4	MMS[1:0]	R/W	0	Master mode selection(TRGO) These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the **TIMx_EGR** register is used as 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. 001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in **TIMx_SMCR** register). 010: Update - The update event is selected as trigger output (TRGO).

Bits	Name	Type	Reset	Description
				For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output (TRGO) send a positive pulse when the CC0I flag is to be set (even if it was already high), as soon as a capture 0 or a compare 0 match counter occurred. (TRGO). 100: Compare - OC0REF signal is used as trigger output (TRGO) 101: Compare - OC1REF signal is used as trigger output (TRGO) 110: Compare - OC2REF signal is used as trigger output (TRGO)
3	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	CCUS	R/W	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 in the **TIMx_EGR** 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit n the **TIMx_EGR** or when an rising edge occurs on TRGI Note: This bit acts only on channels that have a complementary output
1	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	CCPC	R/W	0	Capture/Compare Preloaded Control 0: CCxE, CCxNE and OCxM bits are not preloaded 1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a communication event (COM) occurs (COMG bit set or rising edge detected on TRGI, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output

4.11.3.3 TIMX_SMCR - TIM 4 / 5 SLAVE MODE CONTROL REGISTER



Offset: 0x0008

Bits	Name	Type	Reset	Description
31: 17	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	ETP	R/W	0	External Trigger Polarity This bit selects whether ETR or $\overline{ETR}$ is used for trigger operations 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.
15	ECE	R/W	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. Note: 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=1001). 2: It is possible to simultaneously use external clock mode 2 with the following slave modes: reset mode, gated mode and trigger mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 1001). 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
14:13	ETPS	R/W	0x0	External Trigger Prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8
12:9	ETF	R/W	0x0	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. The digital filter is made of an event counter in which N 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

Bits	Name	Type	Reset	Description
				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: $f_{SAMPLING} = f_{DTS} / 16$, 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 Note: Care must be taken that f_{DTS} is replaced in the formula by CK_INT when ETF[3:0] = 1, 2 or 3.
8	MSM	R/W	0	Master/Slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
7:4	TS	R/W	0	Trigger Selection This bit-field selects the trigger input to be used to synchronize the counter. 0000: Internal Trigger 0 (ITR0) 0001: Internal Trigger 1 (ITR1) 0010: Internal Trigger 2 (ITR2) 0011: Internal Trigger 3 (ITR3) 0100: Internal Trigger 4 (ITR4) 0101: Internal Trigger 5 (ITR5) 0110: TI0 Edge Detector (TI0F_ED) 0111: Filtered Timer Input 0 (TI0FP0) 1000: Filtered Timer Input 1 (TI1FP1) 1001: External Trigger input (ETRF) Note: These bits must be changed only when they are not used (e.g. when SMS=0000) to avoid wrong edge detections at the transition. See below table for more details on ITRx meaning for each Timer.

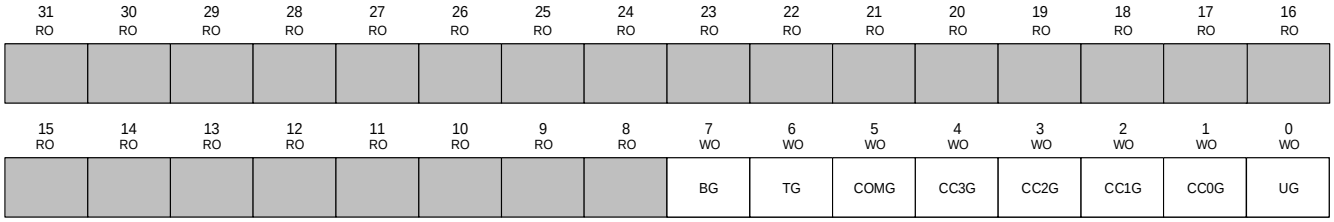
Table 4.11-2: TIMx Internal trigger connection

ITR0(TS = 0000)	ITR1(TS = 0001)	ITR2(TS = 0010)	ITR3(TS = 0011)	ITR4(TS = 0100)	ITR5(TS = 0101)
TIM0	TIM1	TIM2	TIM3	TIM4	TIM5

3	OCCS	R/W	0	OCREF Clear Selection This bit is used to select the OCREF clear source. 0: OCREF_CLR_INT is connected to ETRF 1: OCREF_CLR_INT is connected to COMP_TRIG
2:0	SMS	R/W	0x0	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' then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI0FP0 edge

Bits	Name	Type	Reset	Description
				depending on TI1FP1 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI0FP0 level. 011: Encoder mode 3 - Counter counts up/down on both TI0FP0 and TI1FP1 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 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 at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI0F_ED is selected as the trigger input (TS='0110'). Indeed, TI0F_ED outputs 1 pulse for each transition on TI0F, whereas the gated mode checks the level of the trigger signal.

4.11.3.4 TIMX_EGR - TIM 4 / 5 EVENT GENERATION REGISTER

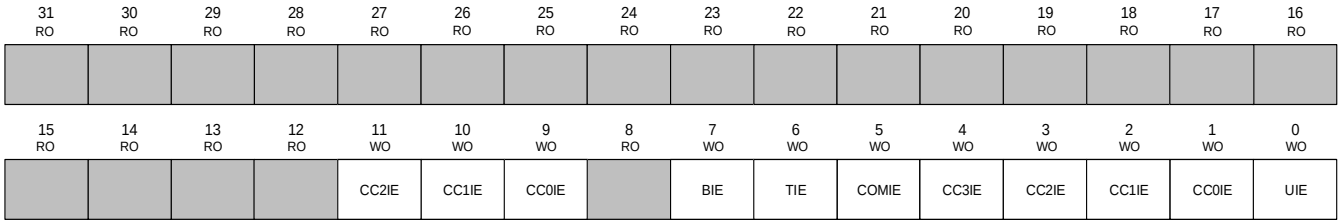


Offset: 0x000C

Bits	Name	Type	Reset	Description
31:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BG	WO	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 BIS status is set. Related interrupt can occur if enabled.
6	TG	WO	0	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware except slave gated mode. 0: No action 1: The TIS status is set in TIMx_ISC register. Related interrupt can occur if enabled.
5	COMG	WO	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 having a complementary output.
4	CC3G	WO	0	Capture/Compare 3 generation Channel CC3 is used to generate an interrupt to trigger internal ADC.
3	CC2G	WO	0	Capture/Compare 2 generation Refer to CC0G description
2	CC1G	WO	0	Capture/Compare 1 generation Refer to CC0G description
1	CC0G	WO	0	Capture/Compare 0 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 0: **If channel CC0 is configured as output:** Corresponding interrupt is sent if enabled. **If channel CC0 is configured as input:** The current value of the counter is captured in **TIMx_CCR0** register. The corresponding interrupt is sent if enabled.
0	UG	WO	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generates an update of the registers.

Bits	Name	Type	Reset	Description
				Note that the prescalercounter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the auto-reload value (**TIMx_ARR**) if DIR=1 (downcounting).

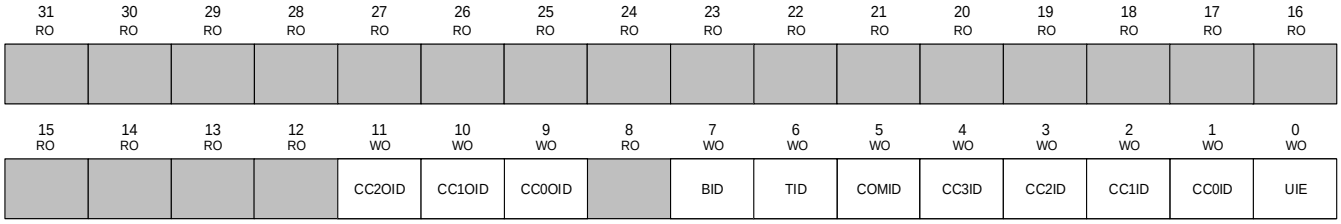
4.11.3.5 TIMX_IER - TIM 4 / 5 INTERRUPT ENABLE REGISTER



Offset: 0x0010

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2OIE	WO	0	Capture/Compare 2 overcapture Interrupt Enable 1: CC2O interrupt is enabled
10	CC1OIE	WO	0	Capture/Compare 1 overcapture Interrupt Enable 1: CC1O interrupt is enabled
9	CC0OIE	WO	0	Capture/Compare 0 overcapture Interrupt Enable 1: CC0O interrupt is enabled
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BIE	WO	0	Break Interrupt Enable 1: Break interrupt is enabled
6	TIE	WO	0	Trigger Interrupt Enable 1: Trigger interrupt is enabled
5	COMIE	WO	0	COM Interrupt Enable 1: COM interrupt is enabled
4	CC3IE	WO	0	Capture/Compare 3 Interrupt Enable 1: CC3 interrupt is enabled
3	CC2IE	WO	0	Capture/Compare 2 Interrupt Enable 1: CC2 interrupt enabled
2	CC1IE	WO	0	Capture/Compare 1 Interrupt Enable 1: CC1 interrupt enabled
1	CC0IE	WO	0	Capture/Compare 0 Interrupt Enable 1: CC0 interrupt enabled
0	UIE	WO	0	Update Interrupt Enable 1: Update interrupt enabled

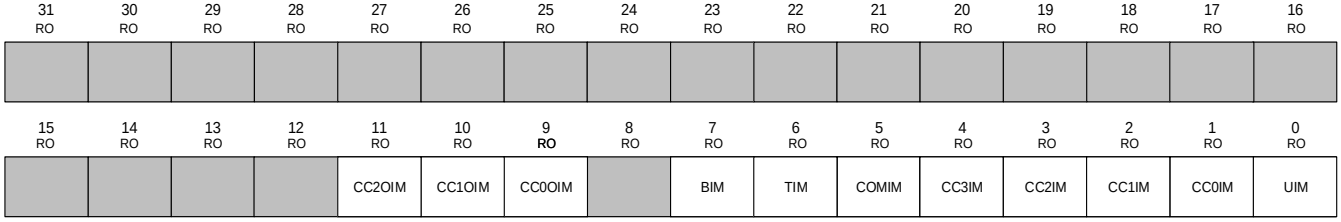
4.11.3.6 TIMX_IDR - TIM4 / 5 INTERRUPT DISABLE REGISTER



Offset: 0x0014

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2OID	WO	0	Capture/Compare 2 overcapture Interrupt Disable 1: CC2O interrupt is disable
10	CC1OID	WO	0	Capture/Compare 1 overcapture Interrupt Disable 1: CC1O interrupt is disable
9	CC0OID	WO	0	Capture/Compare 0 overcapture Interrupt Disable 1: CC0O interrupt is disable
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BID	WO	0	Break Interrupt Disable 1: Break interrupt is disabled
6	TID	WO	0	Trigger Interrupt Disable 1: Trigger interrupt is disabled
5	COMID	WO	0	COM Interrupt Disable 1: COM interrupt is disabled
4	CC3ID	WO	0	Capture/Compare 3 Interrupt Disable 1: CC3 interrupt is disabled
3	CC2ID	WO	0	Capture/Compare 2 Interrupt Disable 1: CC2 interrupt is disabled
2	CC1ID	WO	0	Capture/Compare 1 Interrupt Disable 1: CC1 interrupt is disabled
1	CC0ID	WO	0	Capture/Compare 0 Interrupt Disable 1: CC0 interrupt is disabled
0	UID	WO	0	Update Interrupt Disable 1: Update interrupt is disabled

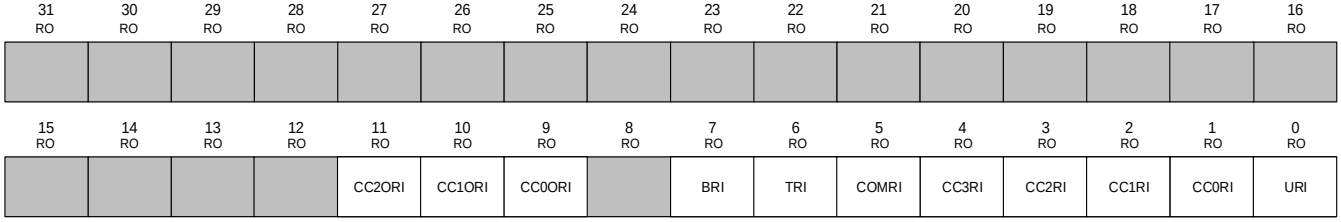
4.11.3.7 TIMX_IMR - TIM 4 / 5 INTERRUPT MASK REGISTER



Offset: 0x0018

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2OIM	RO	0	Capture/Compare 2 overcapture Interrupt Mask Status 0: CC2O interrupt is disable 1: CC2O interrupt is enable
10	CC1OIM	RO	0	Capture/Compare 1 overcapture Interrupt Mask Status 0: CC1O overcapture interrupt is disable 1: CC1O overcapture interrupt is enable
9	CC0OIM	RO	0	Capture/Compare 0 overcapture Interrupt Mask Status 0: CC0O interrupt is disable 1: CC0O interrupt is enable
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BIM	RO	0	Break Interrupt Mask Status 0: Break interrupt is disable 1: Break interrupt is enabled
6	TIM	RO	0	Trigger Interrupt Mask Status 0: Trigger interrupt is disable 1: Trigger interrupt is enabled
5	COMIM	RO	0	COM Interrupt Mask Status 0: COM interrupt is disable 1: COM interrupt is enabled
4	CC3IM	RO	0	Capture/Compare 3 Interrupt Mask Status 0: CC3 interrupt is disable 1: CC3 interrupt is enabled
3	CC2IM	RO	0	Capture/Compare 2 Interrupt Mask Status 0: CC2 interrupt is disable 1: CC2 interrupt is enabled
2	CC1IM	RO	0	Capture/Compare 1 Interrupt Mask Status 0: CC1 interrupt is disable 1: CC1 interrupt is enabled
1	CC0IM	RO	0	Capture/Compare 0 Interrupt Mask Status 0: CC0 interrupt is disable 1: CC0 interrupt is enabled.
0	UIM	RO	0	Update Interrupt Mask Status 0: Update interrupt is disable 1: Update interrupt is enabled

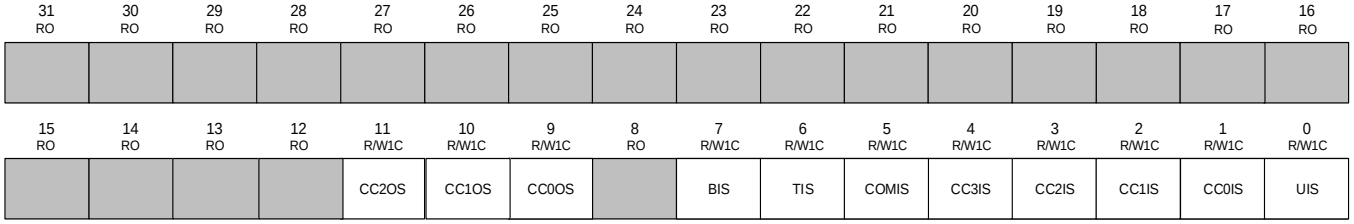
4.11.3.8 TIMX_RIS - TIM 4 / 5 RAW INTERRUPT STATUS



Offset: 0x001C

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2ORI	RO	0	Capture/Compare 2 overcapture Raw Interrupt Status 0: No interrupt 1: CC2O interrupt is asserting.
10	CC1ORI	RO	0	Capture/Compare 1 overcapture Raw Interrupt Status 0: No interrupt 1: CC1O interrupt is asserting.
9	CC0ORI	RO	0	Capture/Compare 0 overcapture Raw Interrupt Status 0: No interrupt 1: CC0O interrupt is asserting.
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BRI	RO	0	Break Raw Interrupt Status 0: No interrupt 1: Break interrupt is asserting.
6	TRI	RO	0	Trigger Raw Interrupt Status 0: No interrupt 1: Trigger interrupt is asserting.
5	COMRI	RO	0	COM Raw Interrupt Status 0: No interrupt 1: COM interrupt is asserting.
4	CC3RI	RO	0	Capture/Compare 3 Raw Interrupt Status 0: No interrupt 1: CC3 interrupt is asserting.
3	CC2RI	RO	0	Capture/Compare 2 Raw Interrupt Status 0: No interrupt 1: CC2 interrupt is asserting.
2	CC1RI	RO	0	Capture/Compare 1 Raw Interrupt Status 0: No interrupt 1: CC1 interrupt is asserting
1	CC0RI	RO	0	Capture/Compare 0 Raw Interrupt Status 0: No interrupt 1: CC0 interrupt is asserting
0	URI	RO	0	Update Raw Interrupt Status 0: No interrupt 1: Update interrupt is asserting

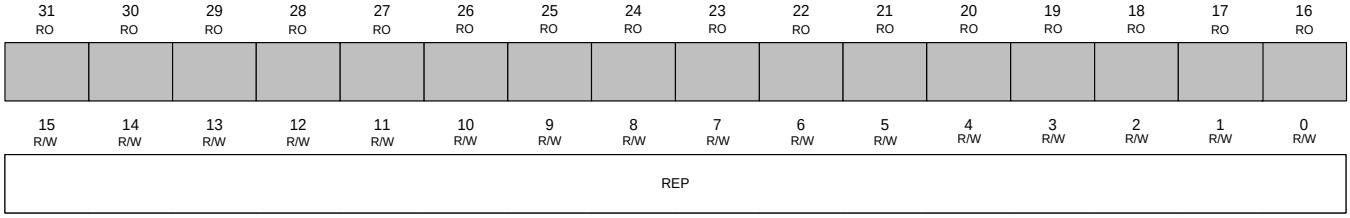
4.11.3.9 TIMX_ ISC - TIM 4 / 5 INTERRUPT STATUS AND CLEAR



Offset: 0x0020

Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2OS	R/W1C	0	Capture/Compare 2 overcapture Status and clear Refer to CC0OF
10	CC1OS	R/W1C	0	Capture/Compare 1 overcapture Status and clear Refer to CC0OF
9	CC0OS	R/W1C	0	Capture/Compare 0 overcapture Status and clear This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '1'. 0: No overcapture has been detected. 1: The counter value has been captured in **TIMx_CCR0** register while CC0RI in the TIMx_RIS was already set
8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7	BIS	R/W1C	0	Break Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: Break interrupt has been signaled.
6	TIS	R/W1C	0	Trigger Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: Trigger interrupt has been signaled.
5	COMIS	R/W1C	0	COM Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: COM interrupt has been signaled.
4	CC3IS	R/W1C	0	Capture/Compare 3 Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: CC3 interrupt has been signaled.
3	CC2IS	R/W1C	0	Capture/Compare 2 Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: CC2 interrupt has been signaled.
2	CC1IS	R/W1C	0	Capture/Compare 1 Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: CC1 interrupt has been signaled.
1	CC0IS	R/W1C	0	Capture/Compare 0 Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: CC0 interrupt has been signaled.
0	UIS	R/W1C	0	Update Interrupt Status and clear 0: No interrupt or interrupt has been masked. 1: Update interrupt has been signaled.

4.11.3.10 TIMX_RCR - TIM 4 / 5 REPETITION COUNTER REGISTER



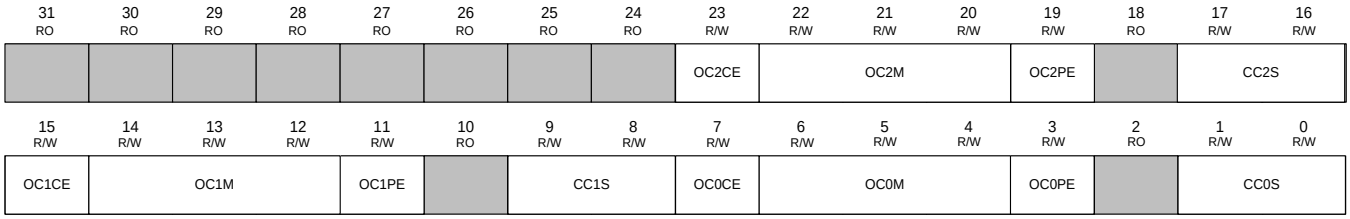
Offset: 0x0024

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	REP	R/W	0x0	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNTR related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event, any write to the **TIMx_RCR** register is not taken in account until the next repetition update event. It means in PWM mode (REP+1) corresponds to: – the number of PWM periods in edge-aligned mode the number of half PWM period in center-aligned mode.

4.11.3.11 TIMX_CCMR - TIM 4 / 5 CAPTURE/COMPARE MODE REGISTER

The channels can be used in input (capture mode) or in 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 in input and in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So you must take care that the same bit can have a different meaning for the input stage and for the output stage.

OUTPUT COMPARE MODE

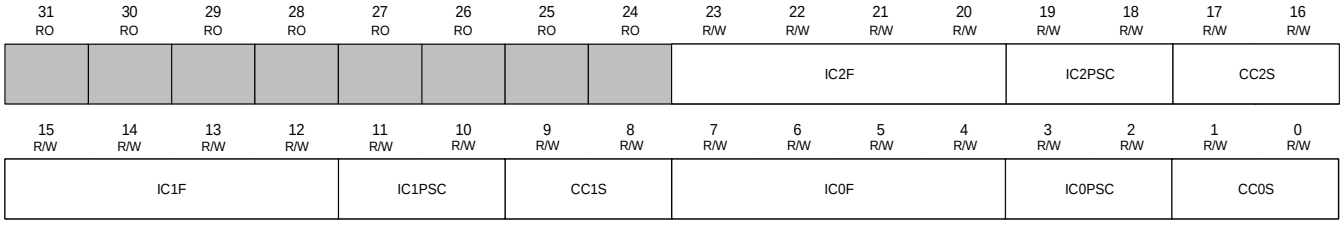


Offset: 0x0028

Bits	Name	Type	Reset	Description
31:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23	OC2CE	R/W	0	Output Compare 2 clear enable Refer to OC0CE
22:20	OC2M	R/W	0x0	Output Compare 2 mode Refer to OC0M
19	OC2PE	R/W	0	Output Compare 2 preload enable Refer to OCPE
18	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
17:16	CC2S	R/W	0x0	Capture/Compare 2 selection Refer to CC0S
15	OC1CE	R/W	0	Output Compare 1 clear enable Refer to OC0CE
14:12	OC1M	R/W	0x0	Output Compare 1 mode Refer to OC0M
11	OC1PE	R/W	0	Output Compare 1 preload enable. Refer to OCPE
10	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9:8	CC1S	R/W	0	Capture/Compare 1 selection Refer to CC0S
7	OC0CE	R/W	0	Output Compare 0 Clear Enable 0: OC0 is not affected by the OCREF_CLR_INPUT 1: OC0 is cleared as soon as a high level is detected on OCREF_CLR_INPUT
6:4	OC0M	R/W	0	Output Compare 0 mode 001: Set channel 0 to active level on match. OC0 signal is forced high when the counter **TIMx_CNTR** matches the capture/compare register 0 (**TIMx_CCR0**). 010: Set channel 0 to inactive level on match. OC0 signal is forced low when the counter **TIMx_CNTR** matches the capture/compare register

Bits	Name	Type	Reset	Description
				0 (**TIMx_CCR0**). 011: Toggle - OC0 toggles when **TIMx_CNTR** matches **TIMx_CCR0**. 100: Force inactive level - OC0 is forced low. 101: Force active level - OC0 is forced high. 110: PWM mode 1 - In upcounting, channel 0 is active as long as **TIMx_CNTR** < **TIMx_CCR1** else inactive. In downcounting, channel 0 is inactive as long as **TIMx_CNTR** > **TIMx_CCR0** else active. 111: PWM mode 2 - In upcounting, channel 0 is inactive as long as **TIMx_CNTR** < **TIMx_CCR0** else active. In downcounting, channel 0 is active as long as **TIMx_CNTR** > **TIMx_CCR0** else inactive. Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in **TIMx_BDTR** register) and CC0S='00' (the channel is configured in output). 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from "frozen" mode to "PWM" mode. 3: On channels having a complementary output, this bit field is preloaded. If the CCPC bit is set in the **TIMx_CR2** register then the OC0M active bits take the new value from the preloaded bits only when a COM event is generated.
3	OC0PE	R/W	0	Output Compare 0 preload enable 0: Preload register on **TIMx_CCR0** disabled. **TIMx_CCR0** can be written at anytime, the new value is taken in account immediately. 1: Preload register on **TIMx_CCR0** enabled. Read/Write operations access the preload register. **TIMx_CCR0** preload value is loaded in the active register at each update event. Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in **TIMx_BDTR** register) and CC0S='00' (the channel is configured in output). 2: The PWM mode can be used without validating the preload register only in one shot mode (PSM bit set in **TIMx_CR1** register). Else the behavior is not guaranteed.
2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1:0	CC0S	R/W	0	Capture/Compare 0 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC0 channel is configured as output Note: CC0S bits are writable only when the channel is OFF (CC0E = '0' in **TIMx_CCER**).

INPUT CAPTURE MODE



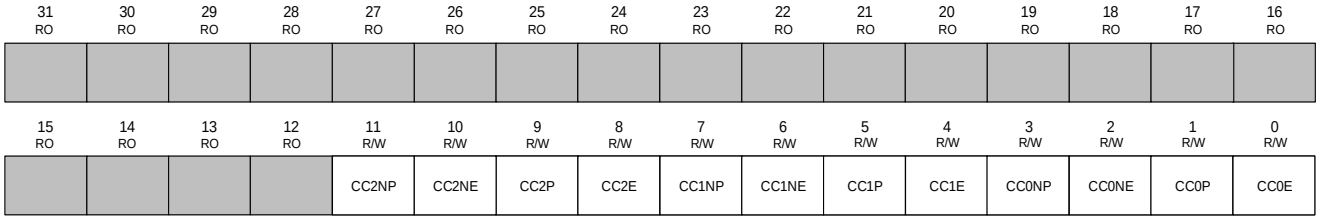
Offset: 0x0018

Bits	Name	Type	Reset	Description
31:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:20	IC2F	R/W	0x0	Input Capture 2 filter Refer to IC0F
19:18	IC2PSC	R/W	0x0	Input Capture 2 prescaler Refer to IC0PSC
17:16	CC2S	R/W	0x0	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: reserved 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (**TIMx_SMCR** register) Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in **TIMx_CCER**).
15:12	IC1F	R/W	0	Input Capture 1 Filter Refer to IC0F
11:10	IC1PSC	R/W	0x0	Input Capture 1 prescaler Refer to IC0PSC
9:8	CC1S	R/W	0x0	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 TI0 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 TS bit (**TIMx_SMCR** register) Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in **TIMx_CCER**).
7:4	IC0F	R/W	0	Input Capture 0 Filter This bit-field defines the frequency used to sample TI0 input and the length of the digital filter applied to TI0. The digital filter is made of an event counter in which N events are needed to validate a transition on the output: 000: No filter, sampling is done at f_{DTS} 0001: $f_{SAMPLING} = f_{CK_INT}$, N = 2

Bits	Name	Type	Reset	Description
				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: $f_{SAMPLING} = f_{DTS}$ / 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 Note: Care must be taken that f_{DTS} is replaced in the formula by CK_INT when ICxF[3:0] = 1, 2 or 3.
3:2	IC0PSC	R/W	0x0	Input Capture 0 prescaler This bit-field defines the ratio of the prescaler acting on CC0 input (IC0). The prescaler is reset as soon as CEN='0' (**TIMx_CR1** 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	CC0S	R/W	0x0	Capture/Compare 0 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC0 channel is configured as output 01: CC0 channel is configured as input, IC0 is mapped on TI0 10: CC0 channel is configured as input, IC0 is mapped on TI1 11: CC0 channel is configured as input, IC0 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (**TIMx_SMCR** register) Note: CC0S bits are writable only when the channel is OFF (CC0E = '0' in **TIMx_CCER**).

4.11.3.12 TIMX_CCER - TIM 4 / 5 CAPTURE/COMPARE ENABLE REGISTER

OUTPUT MODE



Offset: 0x002C

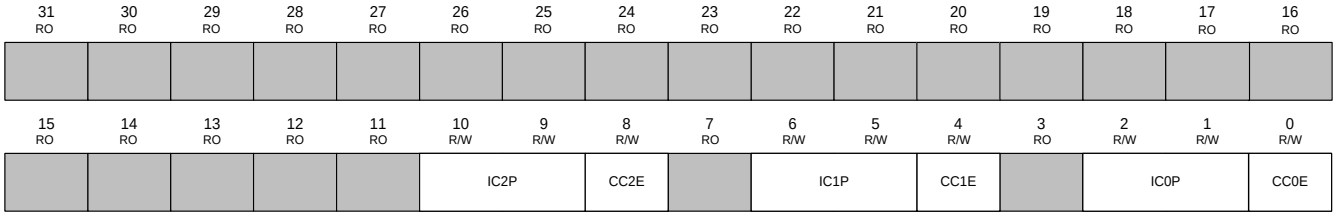
Bits	Name	Type	Reset	Description
31:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	CC2NP	R/W	0	Capture/Compare 2 complementary output polarity Refer to CC0NP
10	CC2NE	R/W	0	Capture/Compare 2 complementary output enable Refer to CC0NE
9	CC2P	R/W	0	Capture/Compare 2 output polarity Refer to CC0P
8	CC2E	R/W	0	Capture/Compare 2 output enable Refer to CC0E
7	CC1NP	R/W	0	Capture/Compare 1 complementary output polarity Refer to CC0NP
6	CC1NE	R/W	0	Capture/Compare 1 complementary output enable Refer to CC0NE
5	CC1P	R/W	0	Capture/Compare 1 output polarity Refer to CC0P
4	CC1E	R/W	0	Capture/Compare 1 output enable Refer to CC0E
3	CC0NP	R/W	0	Capture/Compare 0 complementary output polarity 0: OC0N active high 1: OC0N active low Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the **TIMx_CR2** register then the CC0NP active bit takes the new value from the preloaded bits only when a Commutation event is generated. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in **TIMx_BDTR** register).
2	CC0NE	R/W	0	Capture/Compare 0 complementary output enable 0: Off – OC0N is not active. OC0 level is then function of MOE, OSSI, OSSR, OIS0, OIS0N and CC0NE bits. 1: On – OC0N signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS0, OIS0N and CC0NE bits. Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the **TIMx_CR2** register then the CC0E active bit takes the new value from the preloaded bits only when a Commutation event is generated.

Bits	Name	Type	Reset	Description
1	CC0P	R/W	0	Capture/Compare 0 output polarity 0: OC0 active high 1: OC0 active low Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the **TIMx_CR2** register then the CC0P active bit takes the new value from the preloaded bits only when a Commutation event is generated. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in **TIMx_BDTR** register).
0	CC0E	R/W	0	Capture/Compare 0 output enable 0: Off – OC0 is not active. OC0 level is then function of MOE, OSSI, OSSR, OIS0, OIS0N and CC0NE bits. 1: On – OC0 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS0, OIS0N and CC0NE bits. Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the **TIMx_CR2** register then the CC0E active bit takes the new value from the preloaded bits only when a Commutation event is generated.

Table 4.11-3: Output control bits for complementary OCx and OCxN channels with break feature

Control bits					Output States(1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx output state	OCxN output state
1	X	0	0	0	Output Disabled (not driven by the timer) OCx=0, OCx_EN=1	Output Disabled (not driven by the timer) OCxN=0, OCxN_EN=1
		0	0	1	Output Disabled (not driven by the timer) OCx=0, OCx_EN=1	OCxREF + Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=0
		0	1	0	OCxREF + Polarity OCx=OCxREF xor CCxP, OCx_EN=0	Output Disabled (not driven by the timer) OCxN=0, OCxN_EN=1
		0	1	1	OCREF + Polarity + dead-time OCx_EN=0	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=0
		1	0	0	Output Disabled (not driven by the timer) OCx=CCxP, OCx_EN=1	Output Disabled (not driven by the timer) OCxN=CCxNP, OCxN_EN=1
		1	0	1	Off-State (output enabled with inactive state) OCx=CCxP, OCx_EN=0	OCxREF + Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=0
		1	1	0	OCxREF + Polarity OCx=OCxREF xor CCxP, OCx_EN=0	Off-State (output enabled with inactive state) OCxN=CCxNP, OCxN_EN=0
		1	1	1	OCREF + Polarity + dead-time OCx_EN=0	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=0
0	0	X	0	0	Output Disabled (not driven by the timer) OCx=CCxP, OCx_EN=1	Output Disabled (not driven by the timer) OCxN=CCxNP, OCxN_EN=1
	0		0	1	Output Disabled (not driven by the timer) Asynchronously: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1. Then if the clock is present: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	
	0		1	0		
	0		1	1		
	1		0	0	Output Disabled (not driven by the timer) OCx=CCxP, OCx_EN=1	Output Disabled (not driven by the timer) OCxN=CCxNP, OCxN_EN=1
	1		0	1	Off-State (output enabled with inactive state) Asynchronously: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0 Then if the clock is present: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	
	1		1	0		
	1		1	1		

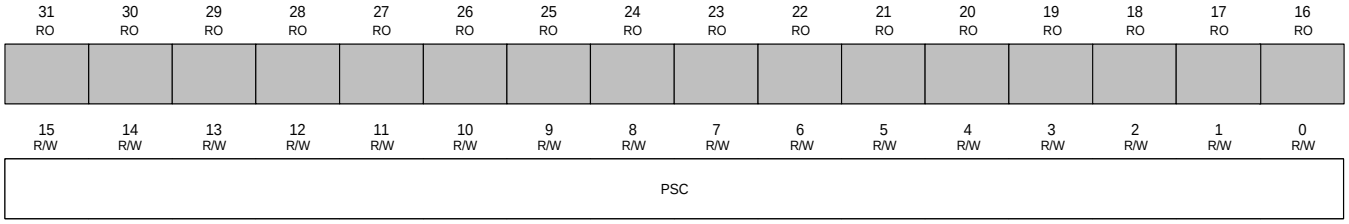
INPUT MODE



Offset: 0x0020

Bits	Name	Type	Reset	Description
31:11	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10:9	IC2P	R/W	0	Input Capture 2 input polarity Refer to IC0P
8	CC2E	R/W	0	Capture/Compare 2 input enable Refer to CC0E
7	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:5	IC1P	R/W	0	Input Capture 1 input polarity Refer to IC0P
4	CC1E	R/W	0	Capture/Compare 1 input enable Refer to CC0E
3	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:1	IC0P	R/W	0	Input Capture 0 input polarity IC0P bits select the active polarity of TI0FP0 and TI1FP0 for trigger or capture operations. 00: non-inverted/rising edge The circuit is sensitive to TIxFP0 rising edge (capture or trigger operations in reset, external clock or trigger mode), TIxFP0 is not inverted (trigger operation in gated mode or encoder mode). 01: inverted/falling edge The circuit is sensitive to TIxFP0 falling edge (capture or trigger operations in reset, external clock or trigger mode), TIxFP0 is inverted (trigger operation in gated mode or encoder mode). 10: reserved 11: non-inverted/both edges
0	CC0E	R/W	0	Capture/Compare 0 input enable 0: Input Capture is disabled 1: Input Capture is enabled

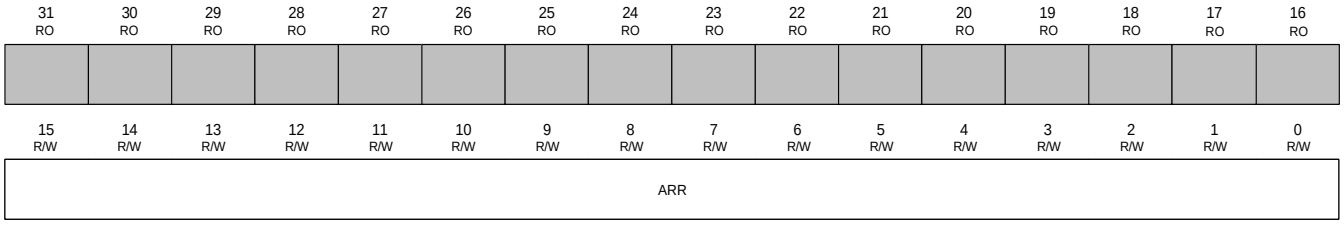
4.11.3.13 TIMX_PSC - TIM 4 / 5 PRESCALER



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	PSC	R/W	0x0	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 in the active prescaler register at each update event (including when the counter is cleared through UG bit of **TIMx_EGR** register or through trigger controller when configured in “reset mode”).

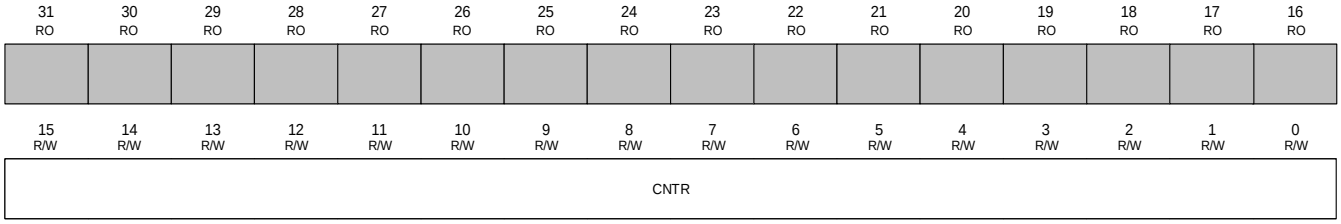
4.11.3.14 TIMX_ARR - TIM 4 / 5 AUTO-RELOAD REGISTER



Offset: 0x0034

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	ARR	R/W	0x0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to the Section 4.11.2.2 Timer-base unit for more details about ARR update and behavior The counter is blocked while the auto-reload value is null.

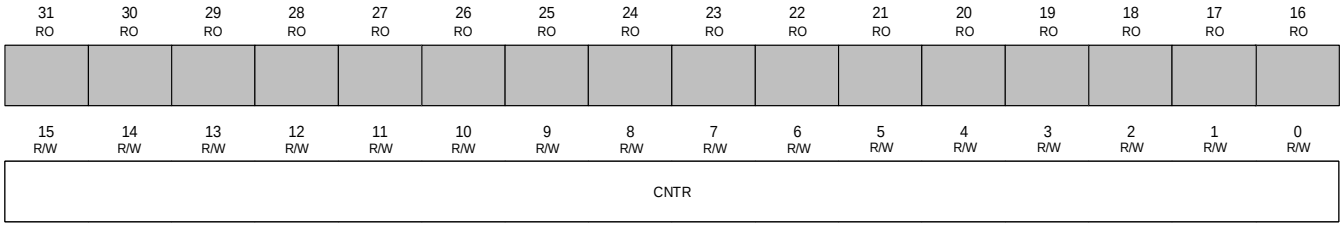
4.11.3.15 TIMX_CNTR - TIM 0 / 1 / 2 / 3 COUNTER



Offset: 0x0038

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	CNTR	R/W	0x0	Counter Register

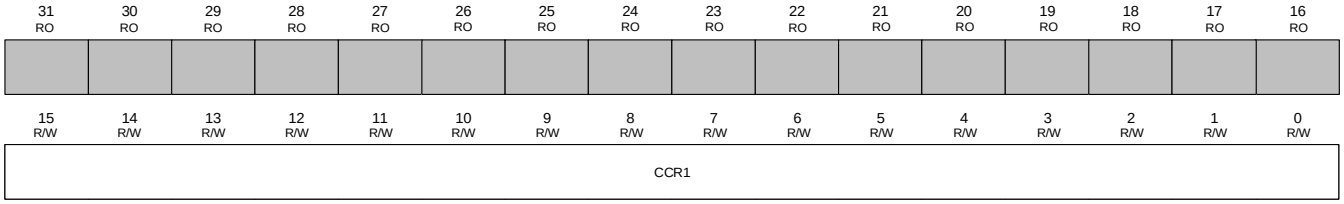
4.11.3.16 TIMX_CCR0 - TIM 4 / 5 CAPTURE/COMPARE REGISTER 0



Offset: 0x003C

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	CCR0	R/W	0x0	Capture/Compare Register 0 **If channel CC0 is configured as output:** CCR0 is the value to be loaded in the actual capture/compare 0 register (preload value). It is loaded permanently if the preload feature is not selected in the **TIMx_CCMR** register (bit OC0PE). Else the preload value is copied in the active capture/compare 0 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter **TIMx_CNTR** and signaled on OC0 output. **If channel CC0 is configured as input:** CCR0 is the counter value transferred by the last input capture 0 event (IC0)

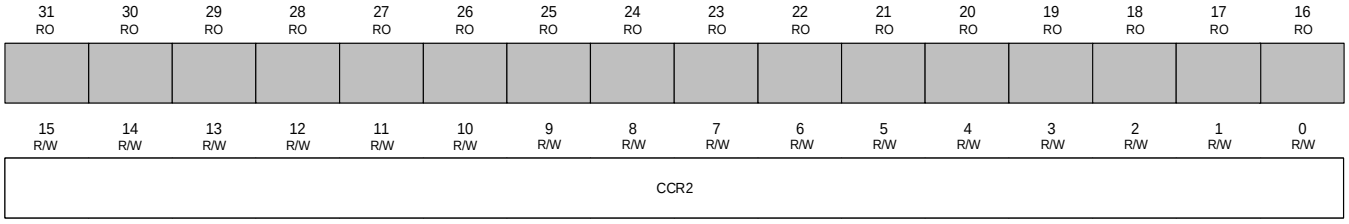
4.11.3.17 TIMX_CCR1 - TIM 4 / 5 CAPTURE/COMPARE REGISTER 1



Offset: 0x0040

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	CCR1	R/W	0x0	Capture/Compare Register 1 Refer to CCR0 in the **TIMx_CCR0**

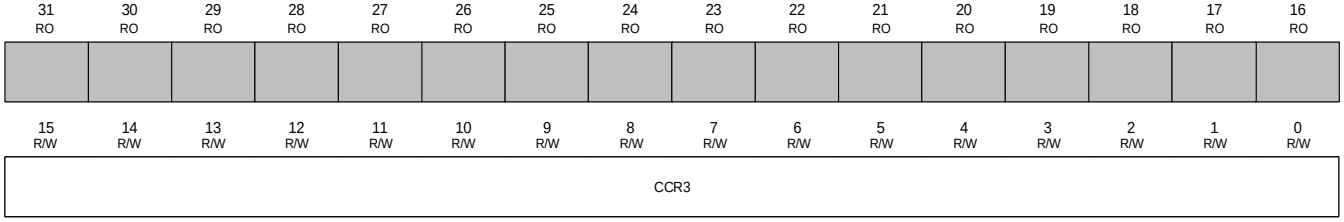
4.11.3.18 TIMX_CCR2 - TIM 4 / 5 CAPTURE/COMPARE REGISTER 2



Offset: 0x0044

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	CCR2	R/W	0x0	Capture/Compare Register 2 Refer to CCR0 in the **TIMx_CCR0**

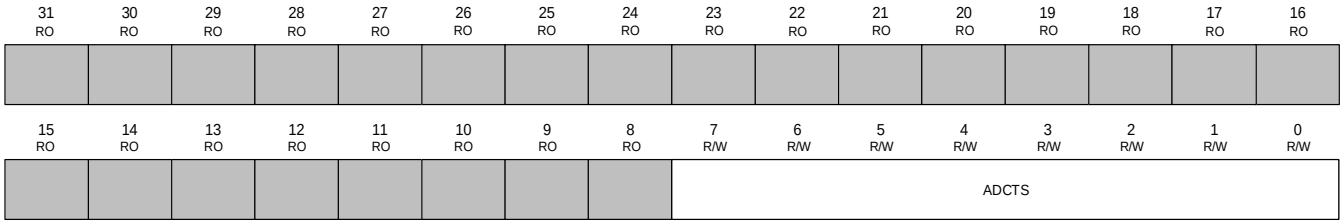
4.11.3.19 TIMX_CCR3 - TIM 4 / 5 CAPTURE/COMPARE REGISTER 3



Offset: 0x0048

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	CCR3	R/W	0x0	Capture/Compare Register 3 CCR3 is used to generate an interrupt to trigger internal ADC.

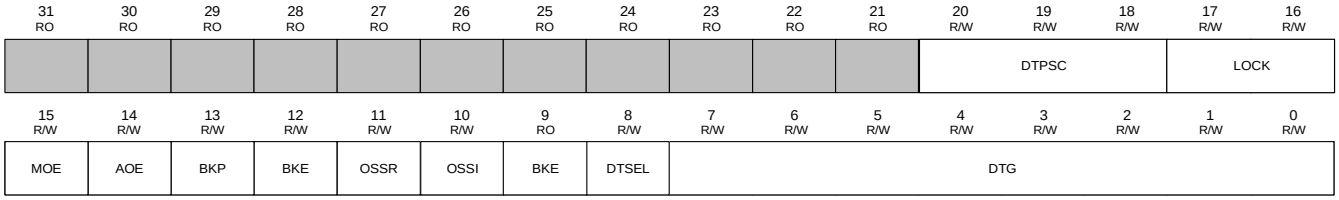
4.11.3.20 TIMX_ADCTS - TIM 4 / 5 ADC TRIGGER SELECT



Offset: 0x004C

Bits	Name	Type	Reset	Description
31:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	ADCTS	R/W	0	ADC Trigger Select Bit0 Select Update Event Raw Interrupt Bit1 Select CC0 Capture/Compare 0 Raw Interrupt Bit2 Select CC1 Capture/Compare 1 Raw Interrupt Bit3 Select CC2 Capture/Compare 2 Raw Interrupt Bit4 Select CC3 Capture/Compare 3 Raw Interrupt Bit5 Select COM Raw Interrupt Bit6 Select Slave Trigger Raw Interrupt Bit7 Select Break Event Raw Interrupt

4.11.3.21 TIMX_ BDTR- TIM 4 / 5 BREAK AND DEAD-TIME REGISTER



Offset: 0x0050

Bits	Name	Type	Reset	Description
31:21	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
20:18	DTPSC	R/W	0	Dead time prescaler These bits are increasing DT rang. Get more details on DTG
17:16	LOCK	R/W	0	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write protected. 01: LOCK Level 1 = DTG bits in **TIMx_BDTR** register, OISx and OISxN bits in **TIMx_CR2** register and BKE/BKP/AOE bits in **TIMx_BDTR** register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in **TIMx_CCER** register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 1 1: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in **TIMx_CCMR** 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.

Table 4.11-4: Lock table

Lock level	Protected register					
Lock level 1(LOCK bit 01)	DTG	OISx	OISxN	BKE	BKP	AOE
Lock level 2(LOCK bit 10)	DTG	OISx	OISxN	BKE	BKP	AOE
	CCxP	CCxNP	OSSR	OSSI		
Lock level 3(LOCK bit 11)	DTG	OISx	OISxN	BKE	BKP	AOE
	CCxP	CCxNP	OSSR	OSSI		
	OCxM	OCxPE				

15	MOE	R/W	0	Main output enable 0: OCx and OCxN outputs are disabled or forced to idle state. 1: OCx and OCxN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in **TIMx_CCER** register).
14	AOE	R/W	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 the break input is not be active) Note: This bit cannot be modified as long as LOCK level 1 has been programmed (LOCK bits in **TIMx_BDTR** register).

Bits	Name	Type	Reset	Description
13	BKP	R/W	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 programmed (LOCK bits in **TIMx_BDTR** register). Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.
12	BKE	R/W	0	Break enable 0: Break inputs BRK event disabled 1: Break inputs BRK event enabled *Note: This bit cannot be modified when LOCK level 1 has been programmed (LOCK bits in **TIMx_BDTR** register).* *Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.*
11	OSSR	R/W	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 is not implemented if no complementary output is implemented in the timer. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal=0). 1: When inactive, OC/OCN outputs are enabled with their inactive level as soon as CCxE=1 or CCxNE=1. Then, OC/OCN enable output signal is OCxP in the **TIMx_CCER** Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in **TIMx_BDTR** register).
10	OSSI	R/W	0	Off-state selection for Idle mode This bit is used when MOE=0 on channels configured as outputs. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal=0). 1: When inactive, OC/OCN outputs are forced first with their idle level as soon as CCxE=1 or CCxNE=1. OC/OCN enable output signal=1) Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in **TIMx_BDTR** register).
9	BKE	RO	0x0	Break enable 0: Break inputs BRK event disabled 1; Break inputs BRK event enabled Note: This bit cannot be modified when LOCK level 1 has been programmed (LOCK bits in **TIMx_BDTR** register). Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.
8	DTSEL	R/W	0	Dead Time selection 0: Rising dead time value=DTG[7:0], Falling dead time value=DTG[7:0] 1: Rising dead time value=DTG[7:4], Falling dead time value=DTG[3:0]
7:0	DTG	R/W	0	Dead-time generator This bit-field defines the duration of the dead-time inserted between the complementary outputs. DTrise&fal correspond to this duration. DTPSC[2:0]=0xx => DTrise&fall=DTL x tdtg with tdtg=tDTS DTPSC[2:0]=10x => DTrise&fall=(64+DTG)xtdtg with Tdtg=2xtDTS

Bits	Name	Type	Reset	Description
				DTPSC[2:0]=110 => DTrise&fall=(32+DTG)xtdtg with Tdtg=8xtDTS DTPSC[2:0]=111 => DTrise&fall=(32+DTG)xtdtg with Tdtg=16xtDTS Note1: DTPSC & DTSEL bit in **TIMx_BDTR** register Note2: This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in **TIMx_BDTR** register). Example if TDTS=125 ns (8 MHz), 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 63 us by 1 us steps, 64 us to 126 us by 2 us steps

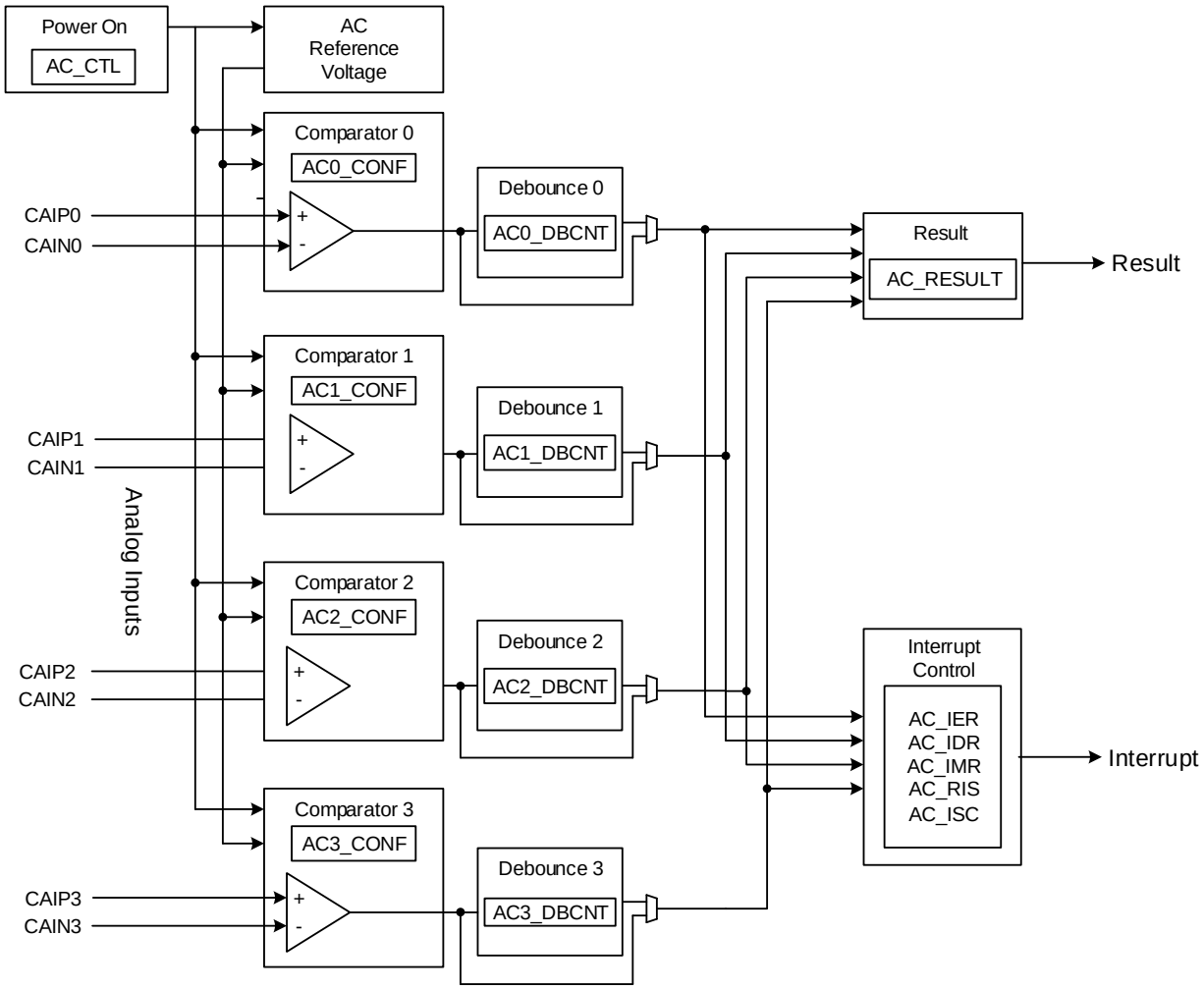
4.12 ANALOG COMPARATOR (AC)

Analog comparator is a peripheral which can compare the value of two analog voltages and shows the comparison result in the form of a logical output. The Analog Comparator supports four individual comparators (AC0, AC1, AC2, and AC3) each of which can provide output to a device pin and replace for an analog comparator on the boarder. AC can be used to trigger ADC via an interrupt or notify the application to start capturing a sample sequence. Independent external reference voltage.

- Four individual analog comparators
- A common external reference voltage.
- Shared internal reference voltage.
- Programmable debounce counter

4.12.1 BLOCK DIAGRAM

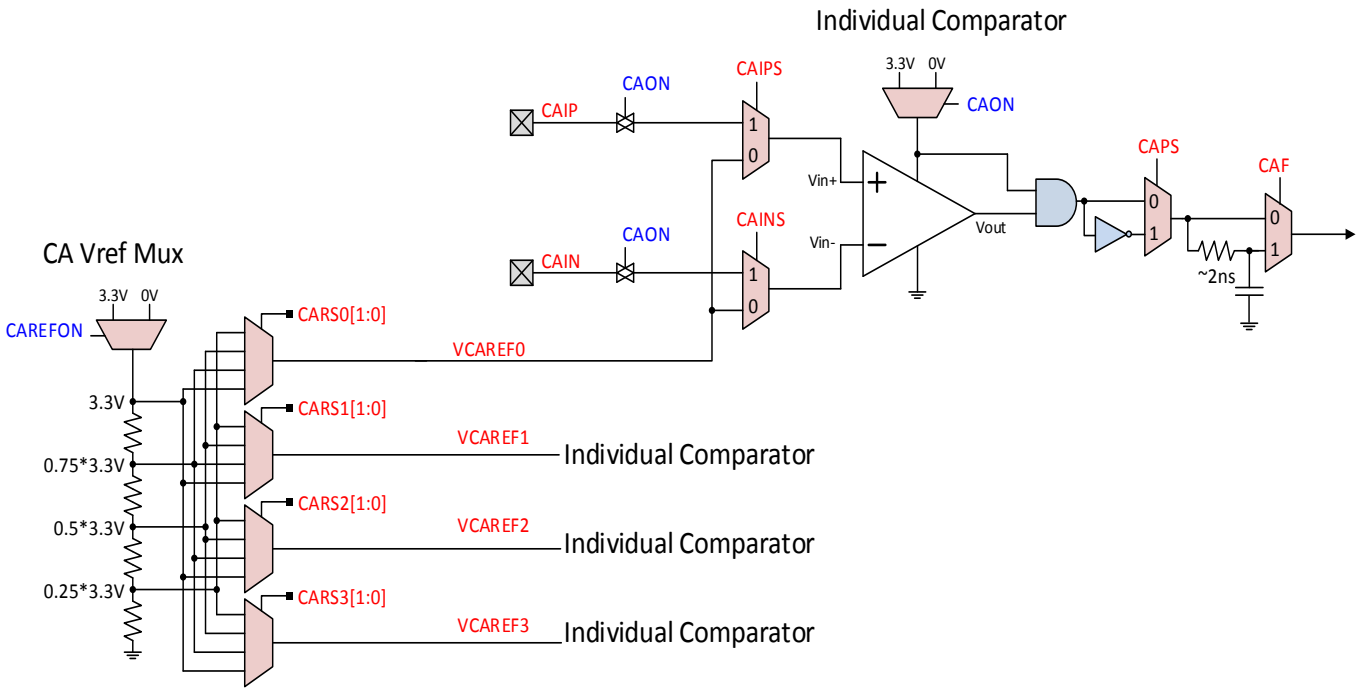
Figure 4.12-1: Analog Comparator Block Diagram



4.12.2 FUNCTIONAL DESCRIPTION

The comparator compares the CAIP and CAIN inputs whose voltage sources are configurable to produce an output. The following figure shows the structure of comparator unit.

Figure 4.12-2: comparator unit



4.12.2.1 INTERRUPT CONTROL

The interrupt is generated when a comparator's output is changed (i.e. Vout is changed.) and these interrupt in AC are controlled by a set of five registers.

- Interrupt Control (IER, IDR, IMR)

  AC Interrupt enable register **(AC_IER)** enables the interrupt request lines by writing a '1'. Similarly, AC Interrupt disable register **(AC_IDR)** disables the interrupt request lines by writing a '1'. *IER* and *IDR* are write only registers which control the masking of interrupts. The overall result of these two registers can be shown by **AC Interrupt Mask Register (AC_IMR)**. IMR is a read-only register using '1' or '0' to indicate if the interrupt request line is enabled/ or disabled.

- Interrupt Status Read (RIS)

  AC Raw Interrupt Status **(AC_RIS)** is a read-only register to read all interrupt status of the module.

- Interrupt Clear (ISC)

  AC Interrupt Status & Interrupt Clear Register **(AC_ISC)** is used to indicate the non-masked interrupt status of the module, since only now-masked interrupts are asserted to processor. Writing a '1' to the bit in this register can clear the corresponding interrupt status or disable the interrupt by writing 1 to *IDR*.

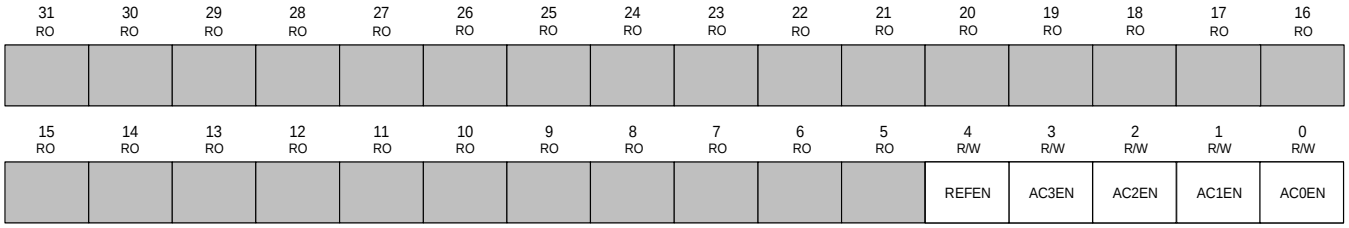
4.12.3 COMPARATOR REGISTER MAP

Base Address: 0x4001_C000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	AC_CTRL	R/W	0x0000_0000	AC Control Register	320
0x0004	AC0_CONF	R/W	0x0000_0000	AC0 Configuration Register	321
0x0008	AC0_BCNT	R/W	0x0000_0000	AC0 Debounce Register	322
0x000C	AC1_CONF	R/W	0x0000_0000	AC1 Configuration Register	321
0x0010	AC1_BCNT	R/W	0x0000_0000	AC1 Debounce Register	322
0x0014	AC2_CONF	R/W	0x0000_0000	AC2 Configuration Register	321
0x0018	AC2_BCNT	R/W	0x0000_0000	AC2 Debounce Register	322
0x001C	AC3_CONF	R/W	0x0000_0000	AC3 Configuration Register	321
0x0020	AC3_BCNT	R/W	0x0000_0000	AC3 Debounce Register	322
0x0024	AC_RESULT	RO	0x0000_0000	AC Output Result	323
0x0028	AC_IER	WO	0x0000_0000	AC Interrupt Enable Register	324
0x002C	AC_IDR	WO	0x0000_0000	AC Interrupt Disable Register	324
0x0030	AC_IMR	RO	0x0000_0000	AC Interrupt Mask Status	325
0x0034	AC_RIS	RO	0x0000_0000	AC Raw Interrupt Status	326
0x0038	AC_ISC	R/W1C	0x0000_0000	AC Interrupt Status and Clear	327

4.12.3.1 AC_CTL - AC CONTROL REGISTER

This register controls the comparator is valid. Each comparator can be individually enabled / disabled.

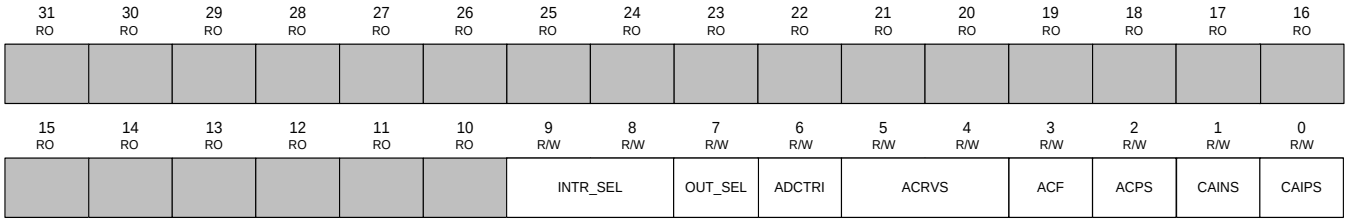


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*REFEN*	R/W	0	Comparator Reference Voltage Enable 0: Reference voltage is off. 1: Reference voltage is valid.
3	*AC3EN*	R/W	0	Comparator 3 Enable 0: Comparator 3 is off. 1: Comparator 3 is valid.
2	*AC2EN*	R/W	0	Comparator 2 Enable 0: Comparator 2 is off. 1: Comparator 2 is valid.
1	*AC1EN*	R/W	0	Comparator 1 Enable 0: Comparator 1 is off. 1: Comparator 1 is valid.
0	*AC0ED*	R/W	0	Comparator 0 Enable 0: Comparator 0 is off. 1: Comparator 0 is valid.

4.12.3.2 ACN_CONF - AC 0/1/2/3 CONFIGURATION REGISTER

This register controls the output of the comparator.



Offset:

CA0_CONF: 0x0004

CA1_CONF: 0x000C

CA2_CONF: 0x0014

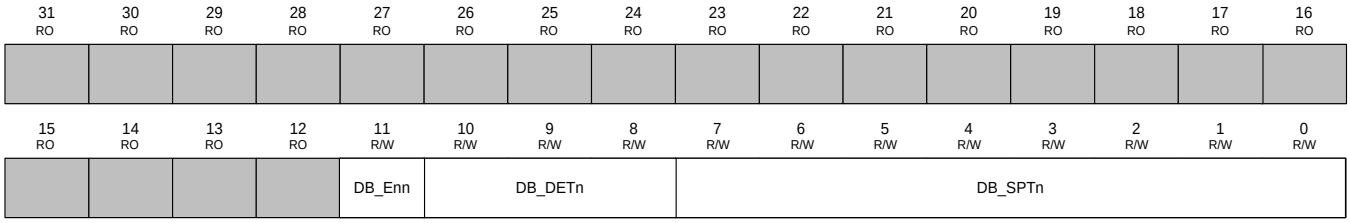
CA3_CONF: 0x001C

Bits	Name	Type	Reset	Description
31:10	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9:8	*INTR_SEL*	R/W	0x3	Comparator Interrupt generate edge 00 : Not use 01 : Rising edge 10 : Falling edge 11 : Both edge
7	*OUT_SEL*	R/W	0	Comparator result select 0 : normal 1 : after de-bounce
6	*ADCTRI*	R/W	0	ADC Trigger Enable 0: ADC trigger is off. 1: ADC trigger is valid.
5:4	*ACRVS*	R/W	0x0	Analog Comparator Reference Voltage Select 00 : 0.25 * 3.3V 01 : 0.5 * 3.3V 10 : 0.75 * 3.3V 11 : 3.3V
3	*ACF*	R/W	0	Analog Comparator Filter Circuit Enable 0: AC filter circuit is off. 1: AC filter circuit is valid.
2	*ACPS*	R/W	0	Analog Comparator Reverse Circuit Enable 0: AC reverse circuit is off. 1: AC reverse circuit is valid.
1	*CAINS*	R/W	0	Comparator Input Negative Select 0: Reference Voltage is selected. 1: Input Voltage is selected.
0	*CAIPS*	R/W	0	Comparator Analog Input Positive Select 0: Reference Voltage is selected. 1: Input Voltage is selected.

4.12.3.3 ACN_DBCNT - AC 0/1/2/3 DEBOUNCE COUNT

This register controls the comparator n debounce counter value, where n is the code for the comparator. For example, assume that the system clock frequency is 16MHz, and set 0x03 to DBCNT, when the input signal from CAO, he will ignore the above 4MHz signal. If the system clock frequency is 16MHz, RMS DBCNT as follows:

DBCNT: "0x0000_0000" The direct input.
DBCNT: "0x0000_0001" After more than 8 MHz signal is ignored, and check the input.
DBCNT: "0x0000_0011" After more than 4 MHz signal is ignored, and check the input.
DBCNT: "0x0000_0111" After more than 2 MHz signal is ignored, and check the input.
DBCNT: "0x0000_1111" After more than 1 MHz signal is ignored, and check the input.
DBCNT: "0x0001_1111" After more than 0.5 MHz signal is ignored, and check the input.
DBCNT: "0x0011_1111" After more than 250 KHz signal is ignored, and check the input.
DBCNT: "0x0111_1111" After more than 125 KHz signal is ignored, and check the input.
DBCNT: "0x1111_1111" After more than 62.5 KHz signal is ignored, and check the input.



Offset:

AC0_DBCNT: 0x0008

AC1_DBCNT: 0x0010

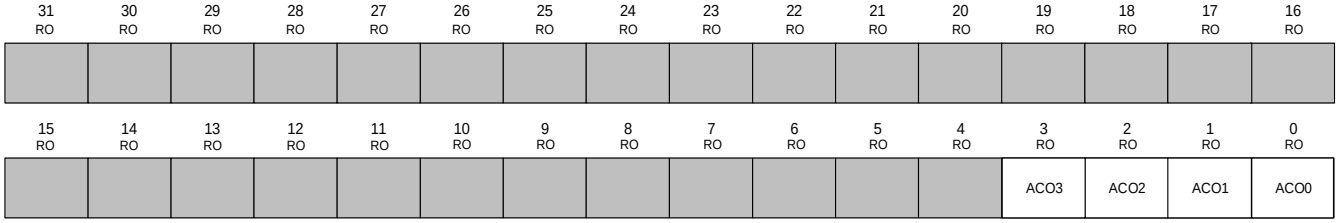
AC2_DBCNT: 0x0018

AC3_DBCNT: 0x0020

Bits	Name	Type	Reset	Description
31:12	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	*DB_Enn*	R/W	0	Comparator Debounce Enable Every Bit can be independently controlled switch Debounce
10:8	*DB_DETn*	R/W	0x0	Comparator Debounce Count times Register
7:0	*DB_SPTn*	R/W	0x0	Comparator Debounce Sample Period Control Register. If enable debounce function, this register must be set a value, and the value is bigger than zero.

4.12.3.4 AC_RESULT - AC OUTPUT RESULT

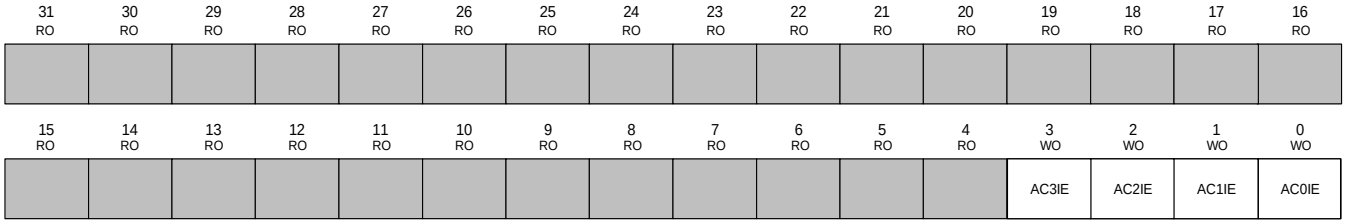
Reading this register to get the value of the comparator output.



Offset: 0x0024

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*ACO3*	RO	0	Comparator 3 Output Result Vin + and Vin - is the positive and negative input voltage (defined by CAIPS and CAINS bits in AC3_CONF) of comparator 3. 0: Vin + < Vin - 1: Vin + ≥ Vin -
2	*ACO2*	RO	0	Comparator 2 Output Result Vin + and Vin - is the positive and negative input voltage (defined by CAIPS and CAINS bits in AC2_CONF) of comparator 2. 0: Vin + < Vin - 1: Vin + ≥ Vin -
1	*ACO1*	RO	0	Comparator 1 Output Result Vin + and Vin - is the positive and negative input voltage (defined by CAIPS and CAINS bits in AC1_CONF) of comparator 1. 0: Vin + < Vin - 1: Vin + ≥ Vin -
0	*ACO0*	RO	0	Comparator 0 Output Result Vin + and Vin - is the positive and negative input voltage (defined by CAIPS and CAINS bits in AC0_CONF) of comparator 0. 0: Vin + < Vin - 1: Vin + ≥ Vin -

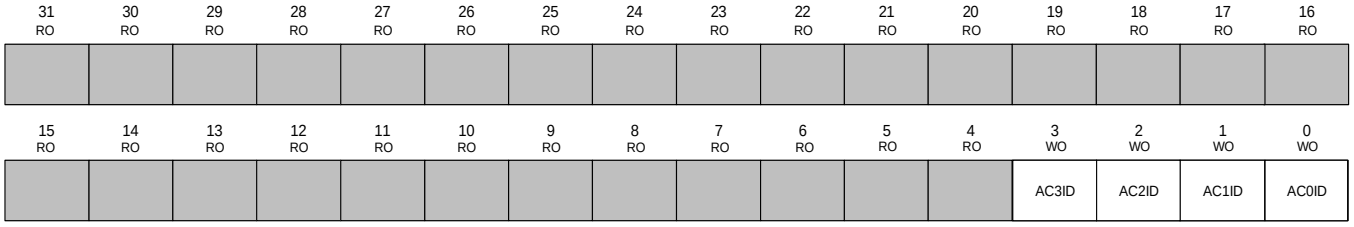
4.12.3.5 AC_IER - AC INTERRUPT ENABLE REGISTER



Offset: 0x0028

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*AC3IE*	WO	0	Comparator 3 Interrupt Enable 1: Interrupt is enabled.
2	*AC2IE*	WO	0	Comparator 2 Interrupt Enable 1: Interrupt is enabled.
1	*AC1IE*	WO	0	Comparator 1 Interrupt Enable 1: Interrupt is enabled.
0	*AC0IE*	WO	0	Comparator 0 Interrupt Enable 1: Interrupt is enabled.

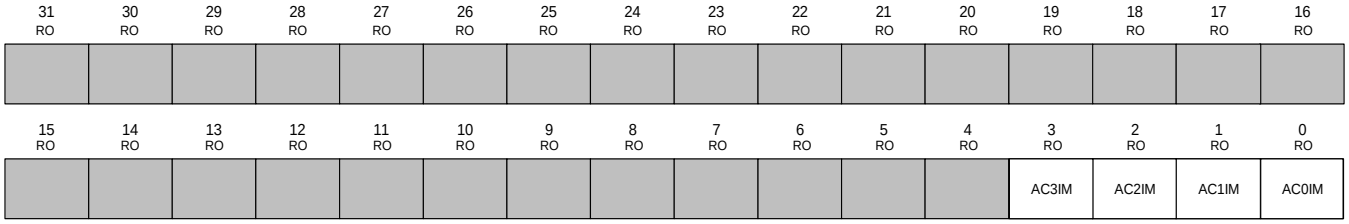
4.12.3.6 AC_IDR - AC INTERRUPT DISABLE REGISTER



Offset: 0x002C

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*AC3ID*	WO	0	Comparator 3 Interrupt Disabled 1: Interrupt is disabled.
2	*AC2ID*	WO	0	Comparator 2 Interrupt Disabled 1: Interrupt is disabled.
1	*AC1ID*	WO	0	Comparator 1 Interrupt Disabled 1: Interrupt is disabled.
0	*AC0ID*	WO	0	Comparator 0 Interrupt Disabled 1: Interrupt is disabled.

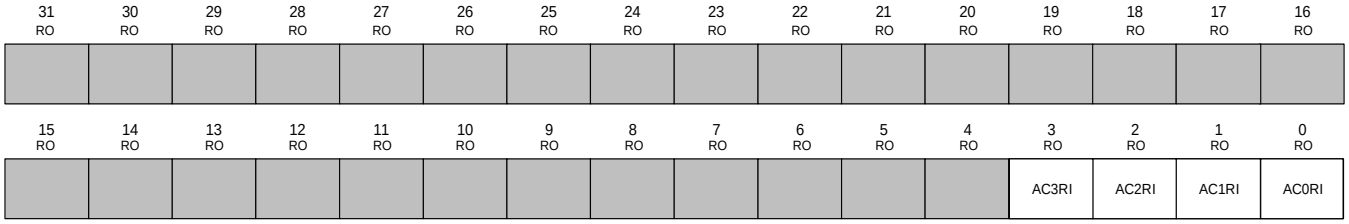
4.12.3.7 AC_IMR - AC INTERRUPT MASK STATUS



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*AC3IM*	RO	0	Comparator 3 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
2	*AC2IM*	RO	0	Comparator 2 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
1	*AC1IM*	RO	0	Comparator 1 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
0	*AC0IM*	RO	0	Comparator 0 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.

4.12.3.8 AC_RIS - AC RAW INTERRUPT STATUS

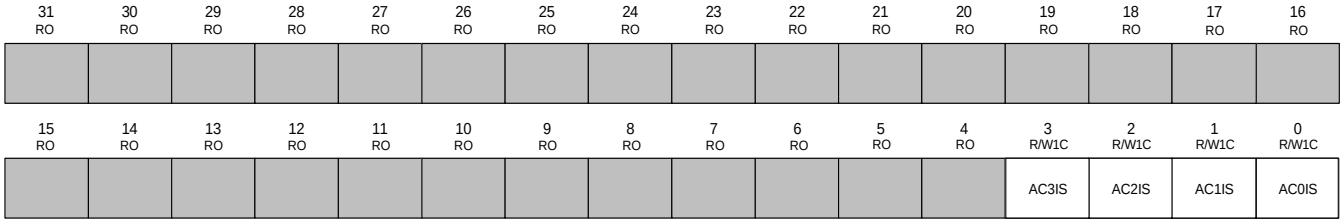


Offset: 0x0034

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*AC3RI*	RO	0	Comparator 3 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
2	*AC2RI*	RO	0	Comparator 2 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
1	*AC1RI*	RO	0	Comparator 1 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
0	*AC0RI*	RO	0	Comparator 0 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.12.3.9 AC_ISC - AC INTERRUPT STATUS AND CLEAR REGISTER

Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.



Offset: 0x0038

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*AC3IS*	R/W1C	0	Comparator 3 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt has been masked. 1: Comparator 3 interrupt has been signaled.
2	*AC2IS*	R/W1C	0	Comparator 2 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt has been masked. 1: Comparator 2 interrupt has been signaled.
1	*AC1IS*	R/W1C	0	Comparator 1 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt has been masked. 1: Comparator 1 interrupt has been signaled.
0	*AC0IS*	R/W1C	0	Comparator 0 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt has been masked. 1: Comparator 0 interrupt has been signaled.

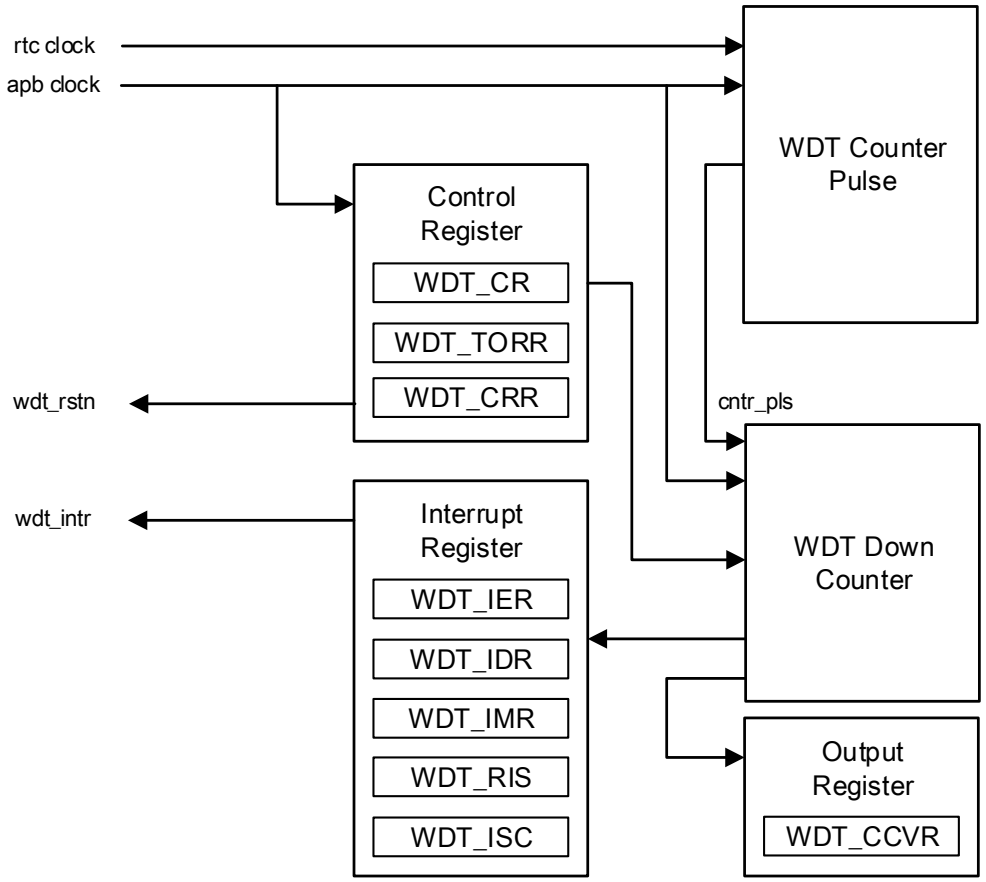
4.13 WATCH DOG TIMER (WDT)

The purpose of Watchdog Timer is to perform a system reset after the software failure or external device failure. This can prevent system from hanging for an infinite period of time. The WDT is configured to a predefined time-out period, and is constantly running when enabled. If the WDT is not cleared within the time-out period, it will issue a system reset. Besides, this Watchdog Timer supports the function to wake CPU up from Power-down mode. The watchdog timer includes a 32-bit free running counter with programmable time-out intervals.

- Two-phase mode to prevent system lockup
  - One phase mode: The WDT timer counts from a preset (timeout) value in descending order to zero. When the counter reaches zero, it can either generate interrupt or simply reset the system and it wrap to the timeout value and continues decrementing.
  - Two phase mode: The WDT can be programmed to generate an interrupt when a first timeout occurs. If it is not cleared by the time a second timeout occurs, then it generates a system reset. If a restart occurs at the same time the watchdog counter reaches zero, an interrupt is not generated.
- 32-bit free-running WDT counter for the watchdog timer timeout interval.
- Selectable timeout interval (2 ^ 4 to 2 ^ 18), the timeout interval is 104 ms ~ 26.316 s (if WDT_CLK = 10 kHz).
- Reset period= (1 / 10 kHz) * 63，if WDT_CLK = 10 kHz.

4.13.1 BLOCK DIAGRAM

Figure 4.13-1: WDT Block Diagram

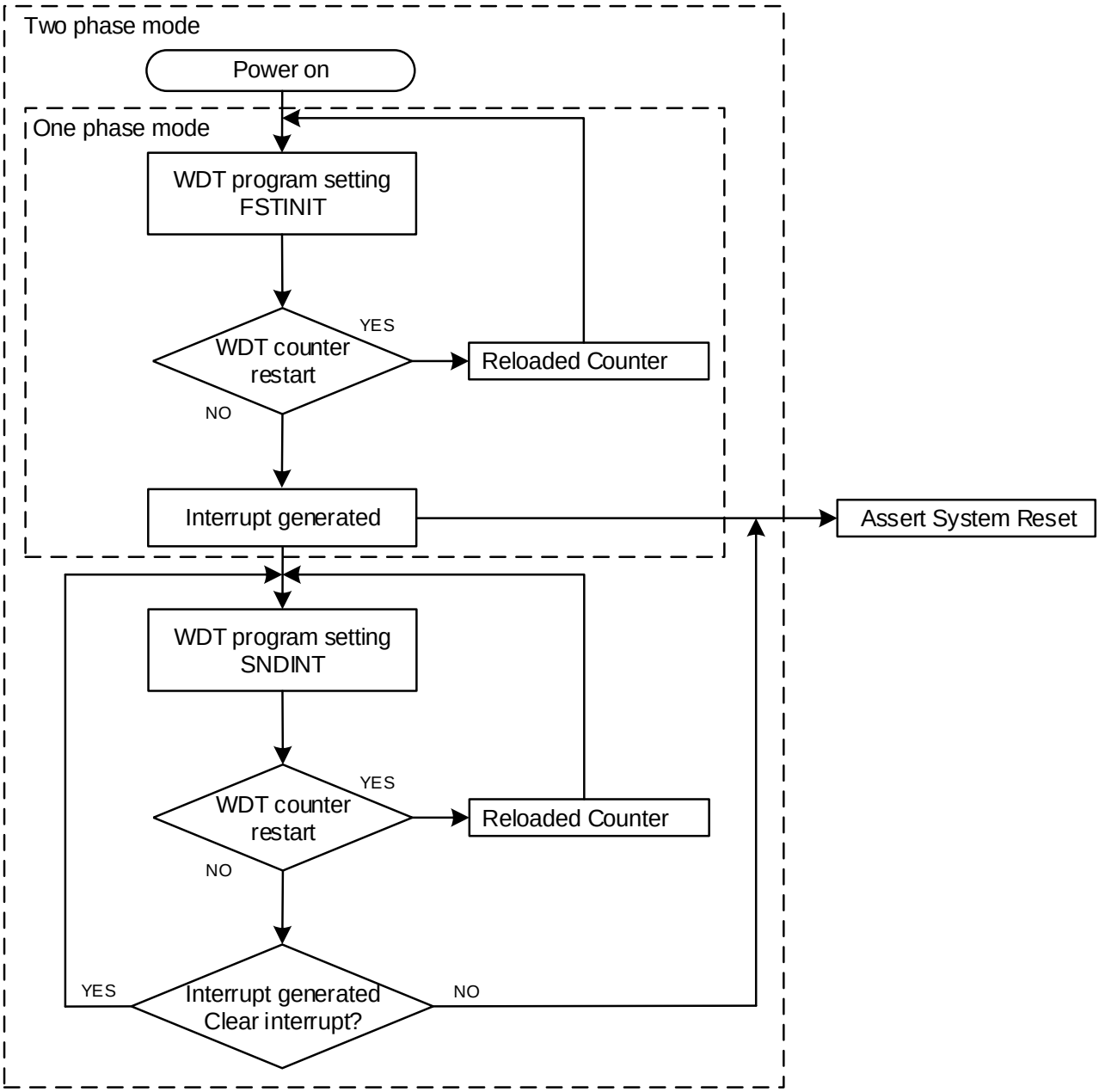


4.13.2 OPERATING FLOWCHART

Following procedure procedure:

1. Enable the related interrupt request by setting the IER bit in the WDT_IER register
2. Select MODSEL bit in WDT_CTRL to select one phase or two phase mode
3. Configure the FSTINIT and SNDINT bit in WDT_TORR to set watchdog restart counter
4. Restart watchdog counter by writing CRR bit = 0x76 in the WDT_CRR
5. Enable the watchdog register by writing ENABLE bit in the WDT_CTRL

Figure 4.13-2: WDT Block Diagram



4.13.3 FUNCTION DESCRIPTION

4.13.3.1 INTERRUPT CONTROL

WDT can be programmed to generate an interrupt (and then a system reset) when a timeout occurs. The interrupt in WDT are controlled by a set of five registers.

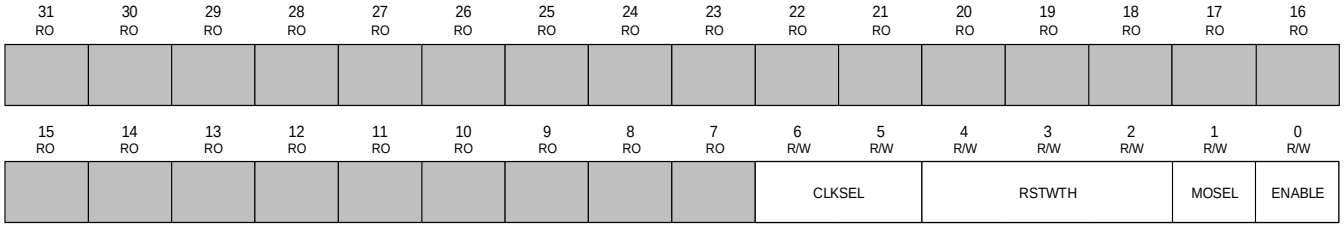
- Interrupt Control (IER, IDR, IMR)
  WDT Interrupt enable register **(WDT_IER)** enables the interrupt request lines by writing a '1'. Similarly, WDT Interrupt disable register **(WDT_IDR)** disables the interrupt request lines by writing a '1'. IER and IDR are write only register, the overall result of the above registers can be shown by WDT Interrupt Mask Register **(WDT_IMR).**IMR is a read-only register using '1' or '0' to indicate if the interrupt request line is enabled/ or disabled.
- Interrupt Status Read (RIS)
  WDT Raw Interrupt Status **(WDT_RIS)** is a read-only register to read the interrupt status from the module.
- Interrupt Clear (ISC)
  WDT Interrupt Status & Interrupt Clear Register **(WDT_ISC)** is used to read the masked interrupt status of the module, showing which interrupt is unmasked. Each bit in ISC is the logical AND of the respective bits in RIS and IMR. Writing a '1' to the bit in this register can clear the corresponding interrupt status but not the interrupt itself.

4.13.4 WDT REGISTER MAP

Base Address: 0x4000_4000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	WDT_CTRL	R/W	0x0000_0000	WDT Control Register	331
0x0004	WDT_TORR	R/W	0x0000_0000	WDT Timeout Range Register	332
0x0008	WDT_CCVR	RO	0x0000_FFFF	WDT Current Counter Value Register	333
0x000C	WDT_CRR	WO	0x0000_0000	WDT Counter Restart Register	333
0x0010	WDT_IER	WO	0x0000_0000	WDT Interrupt Enable Register	334
0x0014	WDT_IDR	WO	0x0000_0000	WDT Interrupt Disable Register	334
0x0018	WDT_IMR	RO	0x0000_0000	WDT Interrupt Mask Register	334
0x001C	WDT_RIS	RO	0x0000_0000	WDT Raw Interrupt Status Register	335
0x0020	WDT_ISC	R/W1C	0x0000_0000	WDT Interrupt Status and Clear Register	335

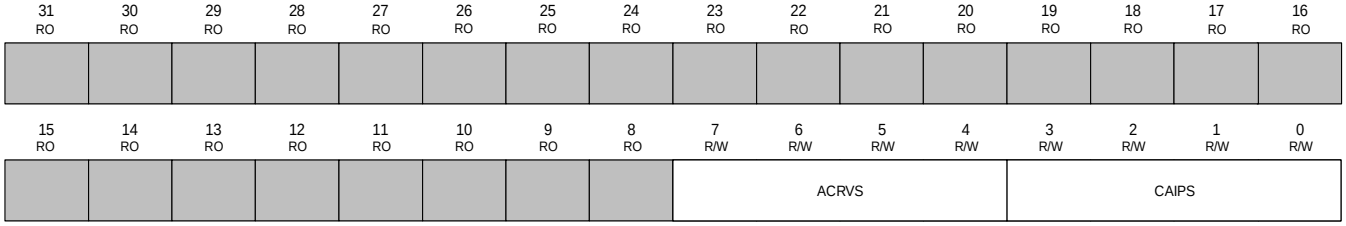
4.13.4.1 WDT_CTRL - WDT CONTROL REGISTER



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:5	*CLKSEL*	R/W	0	WDT Clock Source Select 0/1: APB clock 2: OSC32 clock 3: XTAL32 clock
4:2	*RSTWTH*	R/W	0	Reset Signal Pulse Width This field is used to select the number of pclk cycles for which the system reset stays asserted. The range of values available is 2 to 256 pclk cycles. 000: 2 pclk cycles 001: 4 pclk cycles 010: 8 pclk cycles 011: 16 pclk cycles 100: 32 pclk cycles 101: 64 pclk cycles 110: 128 pclk cycles 111: 256 pclk cycles
1	*MODSEL*	R/W	1	WDT Two Phase Mode Select 0: One phase mode is implemented 1: Two phase mode is implemented.
0	*ENABLE*	R/W	0	WDT Enable 0: WDT is disabled, no interrupts or system reset will be generated. 1: WDT is enabled. Once this bit is enabled, it can only be cleared by system reset.

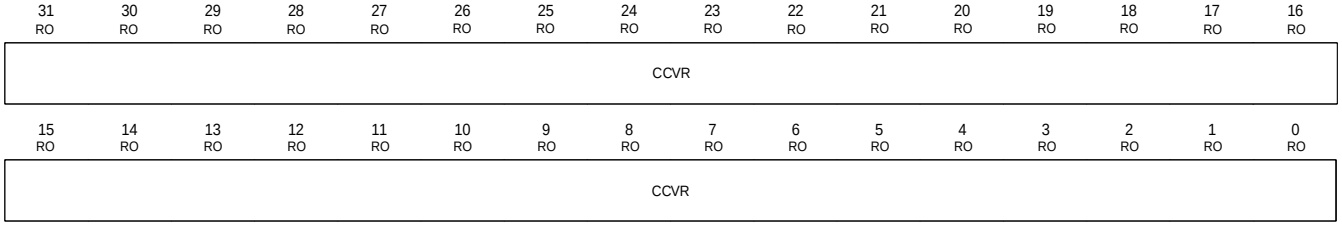
4.13.4.2 WDT_TORR - WDT TIMEOUT RANGE REGISTER



Offset: 0x0004

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:4	*SNDINIT*	R/W	0	Second Timeout Width Period Initialization. This timeout range have no effect under One phase mode. This field is used to select the timeout period that the watchdog counter restarts for the first counter restart. 0x0 : 0xFFFF 0x8 : 0xFFFFFF 0x1 : 0x1FFFF 0x9 : 0x1FFFFFF 0x2 : 0x3FFFF 0xA : 0x3FFFFFF 0x3 : 0x7FFFF 0xB : 0x7FFFFFF 0x4 : 0xFFFFF 0xC : 0xFFFFFFF 0x5 : 0x1FFFF 0xD : 0x1FFFFFF 0x6 : 0x3FFFFF 0xE : 0x3FFFFFFF 0x7 : 0x7FFFFF 0xF : 0x7FFFFFFF
3:0	*FSTINIT*	R/W	0	First Timeout Width Period Initialization This field is used to select the timeout period from which the watchdog counter restarts. 0x0 : 0xFFFF 0x8 : 0xFFFFFF 0x1 : 0x1FFFF 0x9 : 0x1FFFFFF 0x2 : 0x3FFFF 0xA : 0x3FFFFFF 0x3 : 0x7FFFF 0xB : 0x7FFFFFF 0x4 : 0xFFFFF 0xC : 0xFFFFFFF 0x5 : 0x1FFFF 0xD : 0x1FFFFFF 0x6 : 0x3FFFFF 0xE : 0x3FFFFFFF 0x7 : 0x7FFFFF 0xF : 0x7FFFFFFF

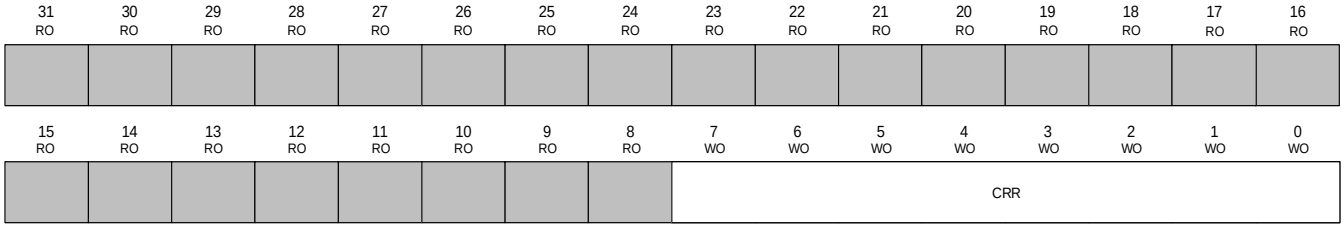
4.13.4.3 WDT_CCVR - WDT CURRENT COUNTER VALUE REGISTER



Offset: 0x0008

Bits	Name	Type	Reset	Description
31:0	*CCVR*	RO	0x0000FFFF	WDT Current Counter Value The current value of the internal counter.

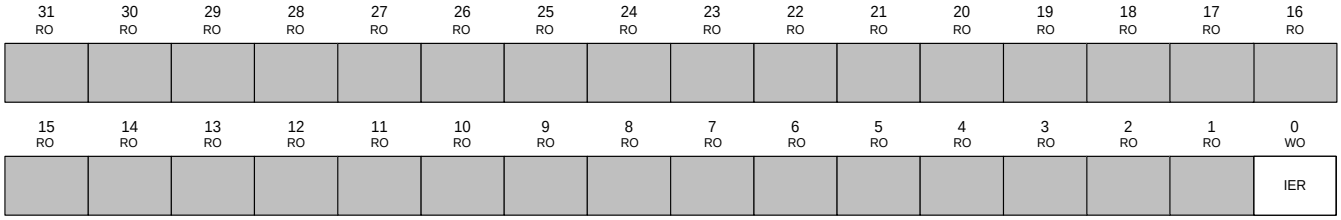
4.13.4.4 WDT_CRR - WDT COUNTER RESTART REGISTER



Offset: 0x000C

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*CRR*	WO	0x0	WDT Counter Restart This register is used to restart the WDT counter. Note: Write CRR bit = 0x76 to restart the WDT counter

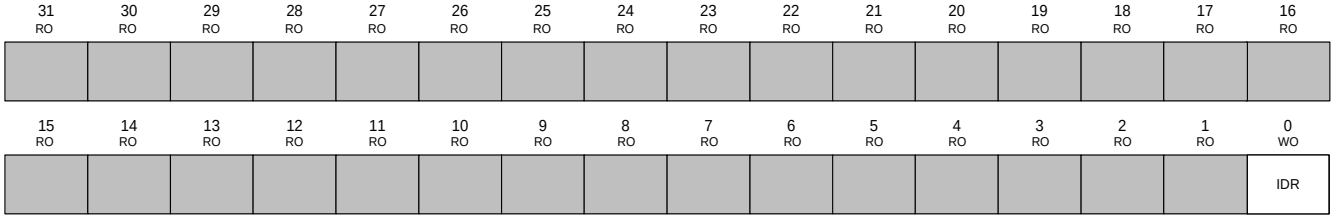
4.13.4.5 WDT_IER - WDT INTERRUPT ENABLE REGISTER



Offset: 0x0010

Bits	Name	Type	Reset	Description
31:1	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*IER*	WO	0	WDT Interrupt Enable 1: Interrupt is enabled.

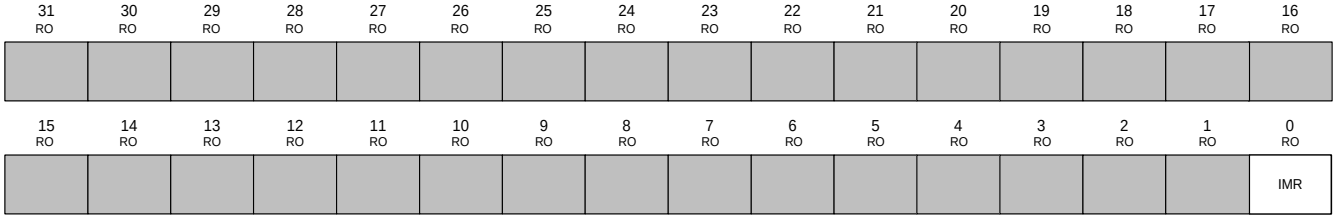
4.13.4.6 WDT_IDR - WDT INTERRUPT DISABLE REGISTER



Offset: 0x0014

Bits	Name	Type	Reset	Description
31:1	*reserved*	N/A	0	reserved
0	*IDR*	WO	0	WDT Interrupt Disable 1: Interrupt is disabled.

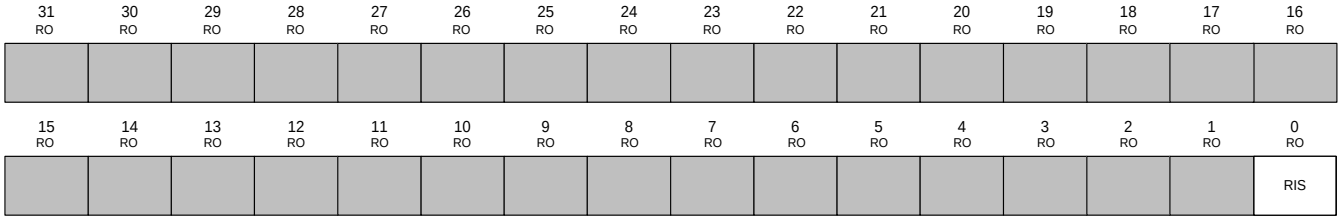
4.13.4.7 WDT_IMR - WDT INTERRUPT MASK REGISTER



Offset: 0x0018

Bits	Name	Type	Reset	Description
31:1	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*IMR*	RO	0	WDT Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.

4.13.4.8 WDT_RIS - WDT RAW INTERRUPT STATUS

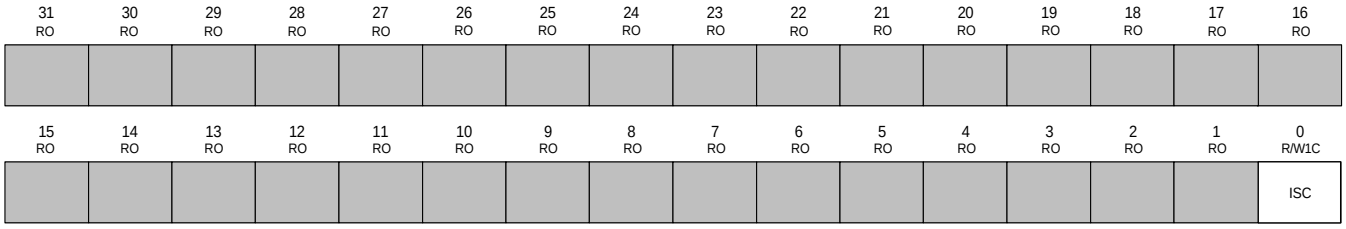


Offset: 0x001C

Bits	Name	Type	Reset	Description
31:1	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*RIS*	RO	0	WDT Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.13.4.9 WDT_ISC - WDT INTERRUPT STATUS AND CLEAR

Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.



Offset: 0x0020

Bits	Name	Type	Reset	Description
31:1	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*ISC*	R/W1C	0	WDT Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.

4.14 REAL TIME CLOCK (RTC)

Real Time Clock (RTC) provides user the real time and calendar message to keep track of time. The RTC can wake up the device from low-power modes.

The clock source of RTC is from a 32.768 kHz Low Speed External crystal or ceramic resonator, LSI RC or PLL reference clock source divided by 128. Refer to **CLOCK DISTRIBUTION**.

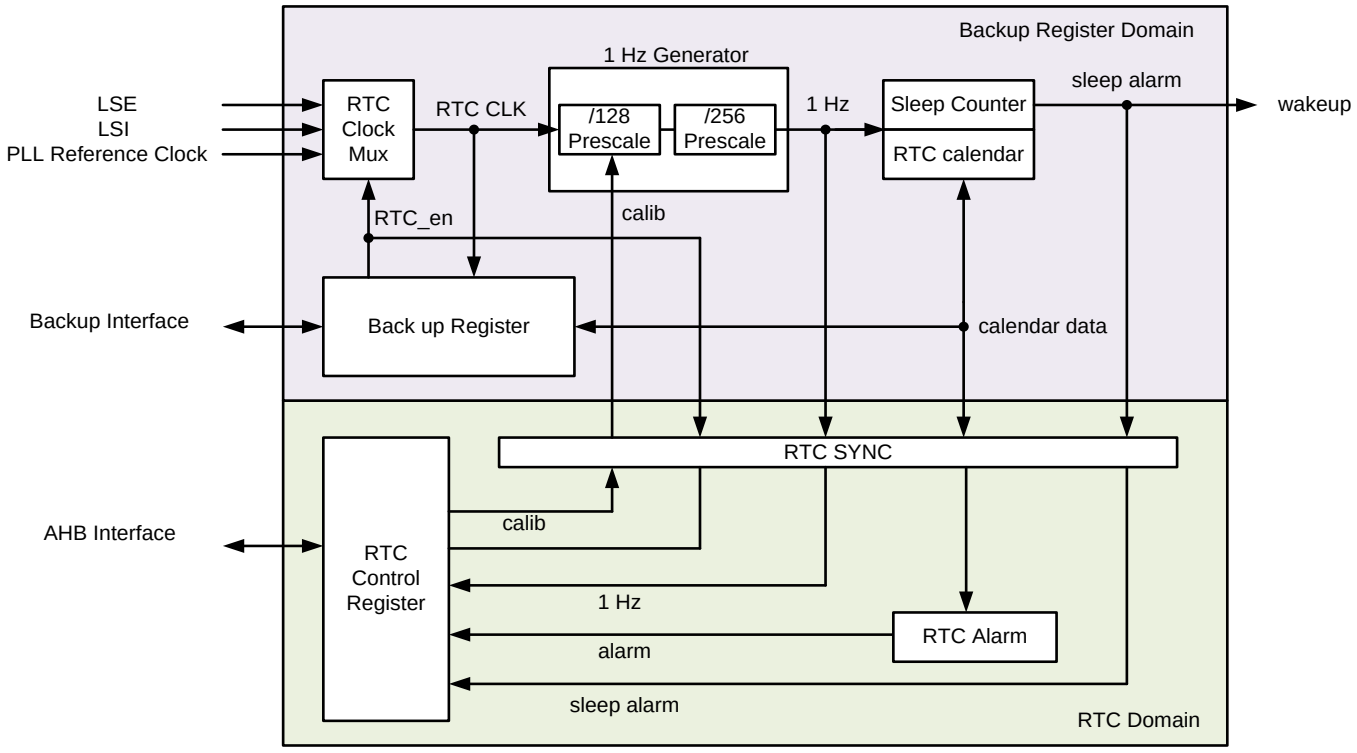
The RTC controller provides time message (hour, minute, second) as well as calendar message (year, month, week and day). The data message is expressed in BCD format. It also offers alarm function that user can preset the alarm time and alarm calendar in Alarm Calendar.

As long as the supply voltage remains in the operating range, the RTC can never stop. Refer to Low Power Mode.

- Calendar with year, month, week and day.
- Time counter with hour, minute and second.
- Programmable alarm with interrupt function which can be triggered by any combination of: year, month, day, week, hour, minute and second.
- Automatic leap year compensation.
- Rollover interrupts: rollover interrupts of year, month, day, week, hour, minute, second.
- All time and calendar message is expressed in BCD code.
- Trigger ADC or DAC function.
- Frequency compensative: clock deviation can be compensated by adjusting counter
- Backup Register

4.14.1 BLOCK DIAGRAM

Figure 4.14-1: I2C Block Diagram



4.14.2 FUNCTIONAL DESCRIPTION

4.14.2.1 RTC ENABLE

To enable RTC function by configuring the bit RTCEN in the RTC controller register (RTC_CTRL). RTC_CTRL is used to set the enable the RTC, configuring the prescale value, scale value, calibration value and calibration in decreasing or increasing.

For example: RTCEN = 1, CLKPSCAL = 0x7F and CLKSCAL = 0xFF. At this configuration, the timer counts 32768 times, then the value of seconds will increase by one.

4.14.2.2 TIME AND CALENDAR MESSAGE INITIALIZATION AND CONFIGURATION

To modify the RTC Time and Calendar message, user need to first disable the RTC function (i.e. RTCEN=0) and then configure the data in RTC_TIME & RTC_CAL registers. Once the setting is done, re-enable the RTC function (i.e. RTCEN=1).

Note: User can only write message to RTC_TIME & RTC_CAL register only when RTCEN is 0. If RTCEN is set to 1, all new written message will be ignored.

4.14.2.3 RTC STATUS

Before reading the data in RTC_TIME and RTC_CAL register, user should first check SYNCSTAT flag is set in RTC status register (RTC_STATUS). This flag indicates if the RTC synchronization procedure is completed, that is the value in RTC_TIME & RTC_CAL register are stable. This status gives precise reading operation to obtain the value from RTC_TIME & RTC_CAL register.

EMPTY bit in RTC_STATUS register is to let user know if the RTC_TIME and RTC_CAL registers are empty.

4.14.2.4 RTC STATUS

The RTC Frequency Compensative allows software to make digital compensation for a too-slow or too-fast oscillator. Following are the compensation examples for higher or lower frequency clock input.

The formula to calculate the calibrated value is as follow:

1. Measured ValueINTEGER - Expected ValueINTEGER= Calibration ValueINTEGER , positive value indicates the clock source is too fast, negative value indicate the clock source is too slow.

2. Measured ValueFRACTION x 60 = Calibration ValueFRACTION

Table 4.14-1: Compensation examples

Calibration for a too-fast oscillator	Calibration for a too-slow oscillator
Frequency counter measurement: 32773.65 Hz Calibration $Value_{INTEGER}$ = 32773 - 32768 = +5 Calibration $Value_{FRACTION}$ = 0.65 x 60 = +39 In this case, for every 1 second, compensate value is +5 and for every 60 seconds, compensate value is +39. **Register Configuration:** 1. Set 1 to the CALIBEN & CALIBCYC in RTC_CALIB register. 2. Configure the RTC_CTRL register as follow: a. Set 0x1 at RTCEN to enable the RTC counting; b. Set 0x1 at CLKSEL to select the RTC clock source; c. Set 0x7F & 0xFF at CLKPSCAL & CLKSCAL to let RTC count 32768 times; d. Set 0x0 & 0x5 at CALIBDIR & CALIBV, for every 1 second, RTC count 32773(32768+5) times. For every 60 seconds, change the CALIBV value from 0x5 to 0x2C (+5+39=44). At the other times, the CALIBV value is 0x5. This operation is used to compensate the fraction part error.	Frequency counter measurement: 32765.27 Hz Calibration $Value_{INTEGER}$ = 32765 - 32768 = -3 Calibration $Value_{FRACTION}$ = 0.27 x 60 = +16.2 In this case, for every 1 second, compensate value is -3 and for every 60 seconds, compensate value is +16. **Register Configuration:** 1. Set 1 to the CALIBEN & CALIBCYC in RTC_CALIB register. 2. Configure the **RTC_CTRL** register as follow: a. Set 0x1 at *RTCEN* to enable the RTC counting; b. Set 0x1 at *CLKSEL* to select the clock source; c. Set 0x7F & 0xFF at CLKPSCAL & CLKSCAL to let RTC count 32768 Hz; d. Set 0x1 & 0x3 at CALIBDIR & CALIBV, for every 1 second, RTC count 32765 (32768-3). For every 60 seconds, change the *CALIBV* value from 0x3 to 0xD (-3+39=44). At the other times, the *CALIBV* value is0x3. This operation is want to compensate the fraction part error.

4.14.2.5 RTC WAKE UP COUNTER

Wake up counter in RTC can be set as a 24 bits timer or as a 24 bits wake up counter by setting CNTSEL in RTC_WKUP register. When user select timer mode (CNTSEL =0x1), the function will start immediately when RTC enable is set to 1. When user select wake up counter mode (CNTSEL =0x2), the function will start immediately when RTC receives sleep signal.

Both timer mode and wake up counter mode support prescale function, prescale value can be 1, 2, 4, 8 and 16.

To disable the RTC wake up counter function, writes 0x0 to RTC_WKUP.CNTSEL.

4.14.2.6 RTC CLK PRESCALE

Prescale function can slow down the RTC clock from 32768 Hz to 1 Hz. This allows the RTC to keep time in seconds in the 32-bit counter. Prescale function in PT32C302 must be only used to generate 1 Hz clock. To be specific, if the prescale value CLKPSCAL and scale value CLKSCAL in RTC_CTRL register are not set to 0x7F and 0xFF, the prescale function will be ignored.

4.14.2.7 ALARM INTERRUPT

Before reading the data in RTC_TIME and RTC_CAL register, user should first check SYNCSTAT flag is set in RTC status register (RTC_STATUS). This flag indicates if the RTC synchronization procedure is completed, that is the value in RTC_TIME & RTC_CAL register are stable. This status gives precise reading operation to obtain the value from RTC_TIME & RTC_CAL register.

EMPTY bit in RTC_STATUS register is to let user know if the RTC_TIME and RTC_CAL registers are empty.

4.14.2.8 RTC READ BACKUP REGISTER

RTC provides user to read time and date data form backup register or from RTC Controller register (RTC_TIME & RTC_CAL register). User can set RTC_BK_CAL.BKREGEN to decide which register will be selected. If user choose to read time and date data form backup register, it needs longer time (approach 19 times AHB CLK) to receive data.

After reading from backup register, clear RTC_BK_CAL.BKREGEN to disable this function.

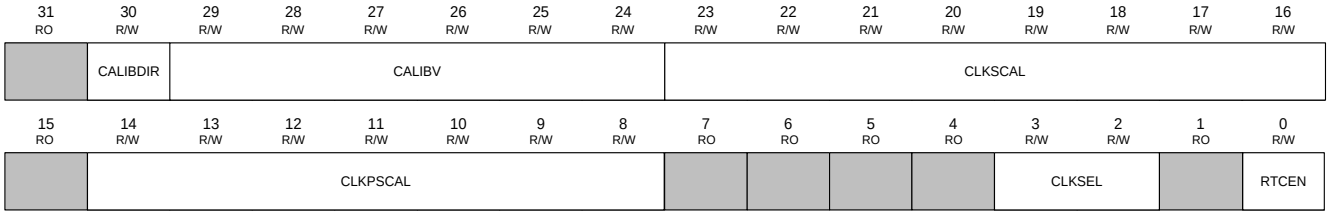
4.14.3 RTC REGISTER MAP

Base Address: 0x5007_0000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	RTC_CTRL	R/W	0x00FF_7F00	RTC Control Register	342
0x0004	RTC_WKUP	R/W	0x00FF_FFFF	RTC Wakeup Counter Register	343
0x0008	RTC_TIME	R/W	0x0000_0000	RTC Calendar Time Set & Current Register	344
0x000C	RTC_CAL	R/W	0x0000_0000	RTC Calendar Date Set & Current Register	345
0x0010	RTC_ALMTIME	R/W	0x0000_0000	RTC Alarm Time Register	346
0x0014	RTC_ALMCAL	R/W	0x0000_0000	RTC Alarm Calendar Register	347
0x0018	RTC_ALMEN	R/W	0x0000_0000	RTC Alarm Enable Set Register	348
0x001C	RTC_CALIB	R/W	0x0000_0000	RTC Calibration Set Register	349
0x0020	RTC_TRIG	R/W	0x0000_0000	RTC Trigger Select Register	350
0x0024	RTC_IER	WO	0x0000_0000	RTC Interrupt Enable Register	351
0x0028	RTC_IDR	WO	0x0000_0000	RTC Interrupt Disable Register	352
0x002C	RTC_IMR	RO	0x0000_0000	RTC Interrupt Mask Register	353
0x0030	RTC_RIS	RO	0x0000_0000	RTC Raw Interrupt Status Register	355
0x0034	RTC_ISC	R/W1C	0x0000_0000	RTC Interrupt Status and Clear Register	357
0x0038	reserved	N/A	0x0000_0000	Reserved	-
0x003C	reserved	N/A	0x0000_0000	Reserved	-
0x0040	RTC_STATUS	RO	0x0000_0000	RTC Status Register	359
0x0044	RTC_BK_CALEN	WO	0x0000_0000	RTC Read Backup Calendar Register Enable	359

4.14.3.1 RTC_CTRL - RTC CONTROL REGISTER

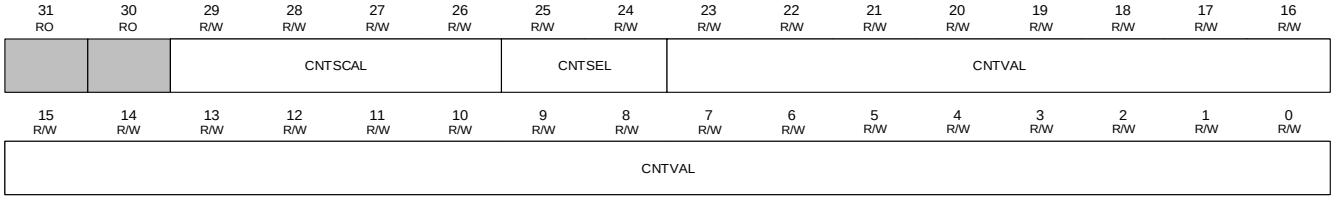
This register is used to temporarily disable any updates in RTC timer and calendar while writing registers in RTC.



Offset: 0x0000

Bits	Name	Type	Reset	Description
31	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
30	*CALIBDIR*	R/W	0	Calibration Value Recovery Direction 0: Increase the counting number of RTC clock by CALIBV. 1: Decrease the counting number of RTC clock by CALIBV.
29:24	*CALIBV*	R/W	0x0	RTC Clock Calibration Value
23:16	*CLKSCAL*	R/W	0xFF	RTC Clock Scale Division factor up to 256
15	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
14:8	*CLKPSCAL*	R/W	0x7F	RTC Clock Prescale Division factor up to 128
7:4	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:2	*CLKSEL*	R/W	0x0	RTC Clock Source Select 00: No clock is select 01: LSE oscillator clock is used as source 10: LSI oscillator clock is used as source 11: PLL Reference clock source which is divided by 128, is used as source.
1	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*RTCEN*	R/W	0	RTC Initialization 0: RTC counter is disabled. 1: RTC counter is enabled.

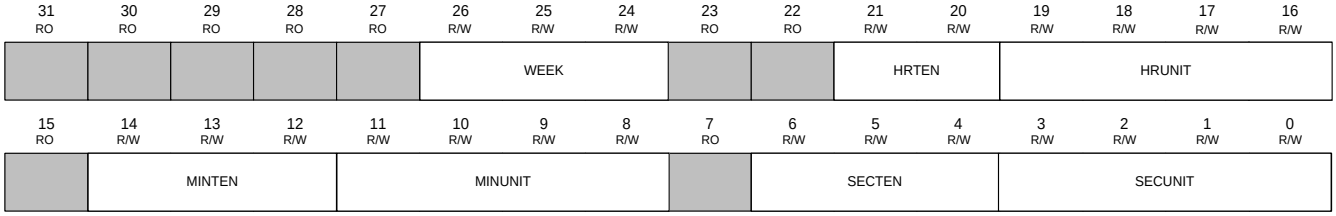
4.14.3.2 RTC_WKUP - RTC WAKEUP COUNTER REGISTER



Offset: 0x0004

Bits	Name	Type	Reset	Description
31:30	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
29:26	*CNTSCAL*	R/W	0x0	RTC Wakeup Counter Scale Value 0x0: Divided by 1 0x1: Divided by 2 0x2: Divided by 4 0x4: Divided by 8 0x8: Divided by 16
25:24	*CNTSEL*	R/W	0x0	RTC Wakeup Counter Events Select 0x0: No event is selected 0x1: RTC start 0x2: Sleep Start 0x3: reserved
23:0	*CNTVAL*	R/W	0xFFFFFF	RTC Wakeup Counter Value Set the counter value.

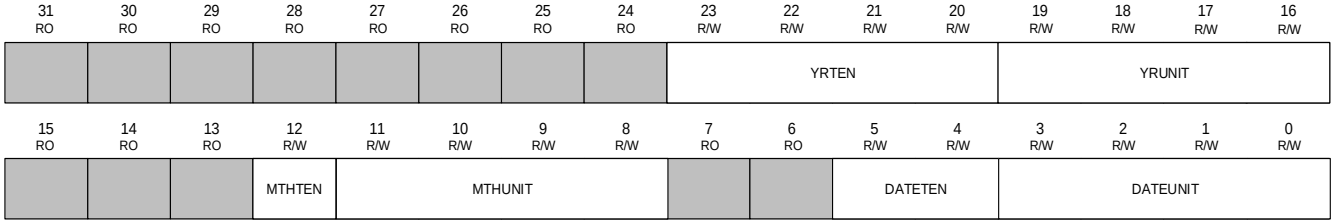
4.14.3.3 RTC_TIME - RTC TIME SET REGISTER AND VALUE



Offset: 0x0008

Bits	Name	Type	Reset	Description
31:27	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26:24	*WEEK*	R/W	0x0	Week Value Weeks in BCD format.
23:22	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
21:20	*HRTEN*	R/W	0x0	Hour Ten Hour tens in BCD format.
19:16	*HRUNIT*	R/W	0x0	Hour Unit Hour units in BCD format.
15	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
14:12	*MINTEN*	R/W	0x0	Minute Ten Minute tens in BCD format.
11:8	*MINUNIT*	R/W	0x0	Minute Unit Minute units in BCD format.
7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:4	*SECTEN*	R/W	0x0	Second Ten Second tens in BCD format.
3:0	*SECUNIT*	R/W	0x0	Second Unit Second units in BCD format.

4.14.3.4 RTC_CAL - RTC Calendar Set Register and Value

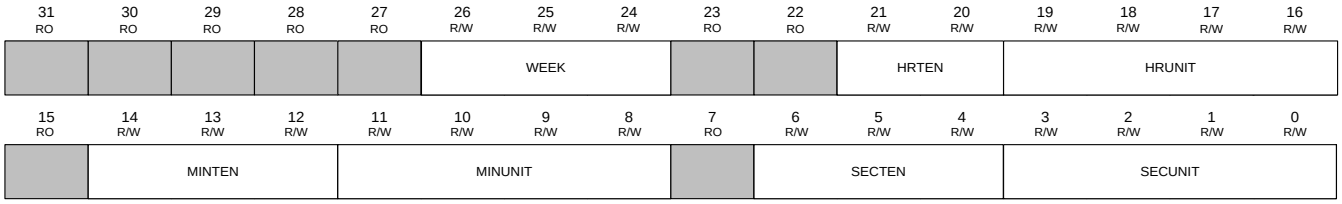


Offset: 0x000C

Bits	Name	Type	Reset	Description
31:24	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:20	*YRTEN*	R/W	0x0	Year Ten Year tens in BCD format.
19:16	*YRUNIT*	R/W	0x0	Year Unit Year units in BCD format
15:13	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
12	*MTHTEN*	R/W	0x0	Month Ten Month tens in BCD format.
11:8	*MTHUNIT*	R/W	0x0	Month Unit Month units in BCD format.
7:6	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5:4	*DATETEN*	R/W	0x0	Date Ten Date tens in BCD format.
3:0	*DATEUNIT*	R/W	0x0	Date Unit Date units in BCD format.

4.14.3.5 RTC_ALMTIME - RTC ALARM TIME SET REGISTER

This is the register the CPU writes to when filling the TX FIFO. Reading from this register returns bytes from RX FIFO.

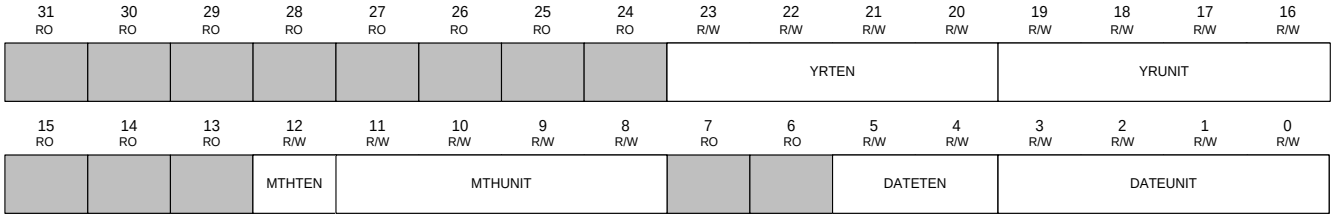


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:27	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
26:24	*WEEK*	R/W	0x0	Alarm Week Value Weeks in BCD format.
23:22	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
21:20	*HRTEN*	R/W	0x0	Alarm Hour Ten Hour tens in BCD format.
19:16	*HRUNIT*	R/W	0x0	Alarm Hour Unit Hour units in BCD format.
15	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
14:12	*MINTEN*	R/W	0x0	Alarm Minute Ten Minute tens in BCD format.
11:8	*MINUNIT*	R/W	0x0	Alarm Minute Unit Minute units in BCD format.
7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:4	*SECTEN*	R/W	0x0	Alarm Second Ten Second tens in BCD format.
3:0	*SECUNIT*	R/W	0x0	Alarm Second Unit Second units in BCD format.

4.14.3.6 RTC_ALMCAL - RTC ALARM CALENDAR SET REGISTER

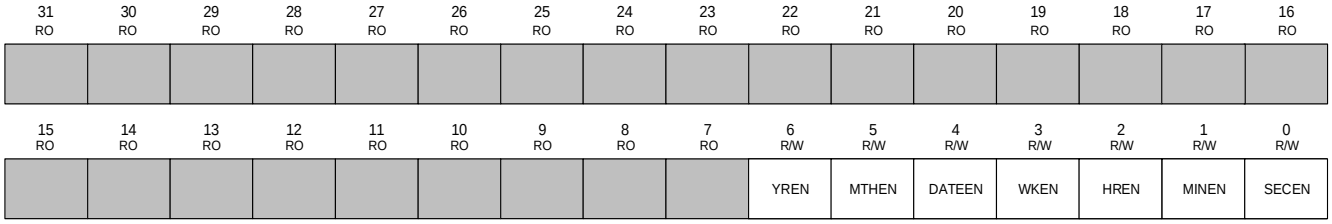
This register is used along with **RTC_ARTIME** to set the RTC alarm.



Offset: 0x0014

Bits	Name	Type	Reset	Description
31:24	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
23:20	*YRTEN*	R/W	0x0	Alarm Year Ten Year tens in BCD format.
19:16	*YRUNIT*	R/W	0x0	Alarm Year Unit Year units in BCD format
15:13	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
12	*MTHTEN*	R/W	0x0	Alarm Month Ten Month tens in BCD format.
11:8	*MTHUNIT*	R/W	0x0	Alarm Month Unit Month units in BCD format.
7:6	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5:4	*DATETEN*	R/W	0x0	Alarm Date Ten Date tens in BCD format.
3:0	*DATEUNIT*	R/W	0x0	Alarm Date Unit Date units in BCD format.

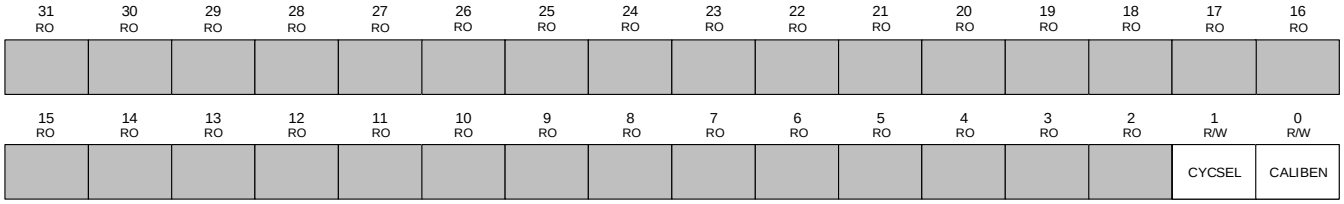
4.14.3.7 RTC_ALMEN - RTC ALARM ENABLE SET REGISTER



Offset: 0x0018

Bits	Name	Type	Reset	Description
31: 7	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6	*YREN*	R/W	0	Alarm Function of Year Enable 0: Alarm function of year is disabled. 1: Alarm function of year is enabled
5	*MTHEN*	R/W	0	Alarm Function of Month Enable 0: Alarm function of month is disabled. 1: Alarm function of month is enabled
4	*DATEEN*	R/W	0	Alarm Function of Date Enable 0: Alarm function of date is disabled. 1: Alarm function of date is enabled
3	*WKEN*	R/W	0	Alarm Function of Week Enable 0: Alarm function of week is disabled. 1: Alarm function of week is enabled
2	*HREN*	R/W	0	Alarm Function of Hour Enable 0: Alarm function of hour is disabled. 1: Alarm function of hour is enabled
1	*MINEN*	R/W	0	Alarm Function of Minute Enable 0: Alarm function of minute is disabled. 1: Alarm function of minute is enabled
0	*SECEN*	R/W	0	Alarm Function of Second Enable 0: Alarm function of second is disabled. 1: Alarm function of second is enabled

4.14.3.8 RTC_CALIB - RTC CALIBRATION SET REGISTER



Offset: 0x001C

Bits	Name	Type	Reset	Description
31: 2	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*CALIBCYC*	R/W	0	Calibration Cycle Select When CALIBEN is set, 0: The 60-second calibration cycle period is selected. 1: The 1-second calibration cycle period is selected.
0	*CALIBEN*	R/W	0	Calibration Function Enable 0: Calibration of RTC is disabled. 1: Calibration of RTC is enabled

4.14.3.9 RTC_TRIG - RTC TRIGGER SELECT REGISTER

RTC_TRIG register is used to enable individual bit for triggering the ADC.

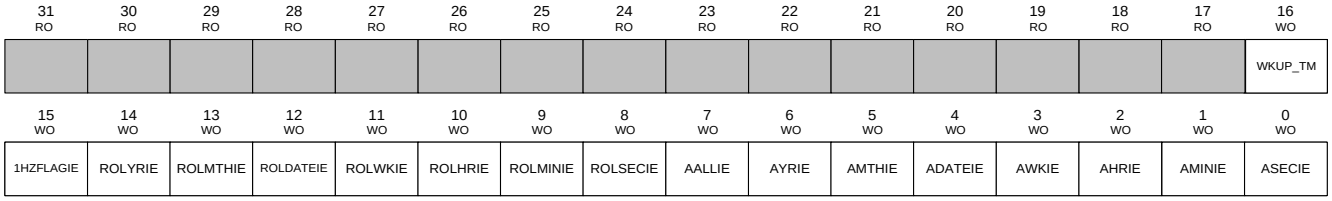
31 RO	30 RO	29 RO	28 RO	27 RO	26 RO	25 RO	24 RO	23 RO	22 RO	21 RO	20 RO	19 RO	18 RO	17 RO	16 R/W
															TRIG_WKUP_TM

15 R/W	14 R/W	13 R/W	12 R/W	11 R/W	10 R/W	9 R/W	8 R/W	7 R/W	6 R/W	5 R/W	4 R/W	3 R/W	2 R/W	1 R/W	0 R/W
ONESEC	RYR	RMTH	RDATE	RWK	RHR	RMIN	RSEC	AALL	AYR	AMTH	ADATE	AWK	AHR	AMIN	ASEC

Offset: 0x0020

Bits	Name	Type	Reset	Description
31: 17	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	*TRIG_WKUP_TM*	R/W	0	RTC Trigger Wakeup Counter Time Match
15	*ONESEC*	R/W	0	RTC Trigger 1 second Flag This bit is related to the signal RTC_CLK. 1: RTC_CLK ADC trigger is enabled. For every one second, the RTC generates an ADC trigger.
14	*RYR*	R/W	0	Rollover Year Enable 1: Year rollover ADC trigger is enabled.
13	*RMTH*	R/W	0	Rollover Month Enable 1: Month rollover ADC trigger is enabled.
12	*RDATE*	R/W	0	Rollover Date Enable 1: Date rollover ADC trigger is enabled.
11	*RWK*	R/W	0	Rollover Week Enable 1: Week rollover ADC trigger is enabled.
10	*RHR*	R/W	0	Rollover Hour Enable 1: Hour rollover ADC trigger is enabled.
9	*RMIN*	R/W	0	Rollover Minute Enable 1: Minute rollover ADC trigger is enabled.
8	*RSEC*	R/W	0	Rollover Second Enable 1: Second rollover ADC trigger is generated.
7	*AALL*	R/W	0	Alarm All Enable 1: When RTC_TIME and RTC_CAL are matched with the setting in RTC_ALMTIME and RTC_ALMCAL, the ADC trigger signal is generated.
6	*AYR*	R/W	0	Year Alarm Enable 1: ADC trigger is generated when the alarm year value is matched.
5	*AMTH*	R/W	0	Month Alarm Enable 1: ADC trigger is generated when the alarm month value is matched.
4	*ADATE*	R/W	0	Date Alarm Enable 1: ADC trigger is generated when the alarm date value is matched.
3	*AWK*	R/W	0	Week Alarm Enable 1: ADC trigger is generated when the alarm week value is matched.
2	*AHR*	R/W	0	Hour Alarm Enable 1: ADC trigger is generated when the alarm hour value is matched.
1	*AMIN*	R/W	0	Minute Alarm Enable 1: ADC trigger is generated when the alarm minute value is matched.
0	*ASEC*	R/W	0	Second Alarm Enable 1: ADC trigger is generated when the alarm second value is matched.

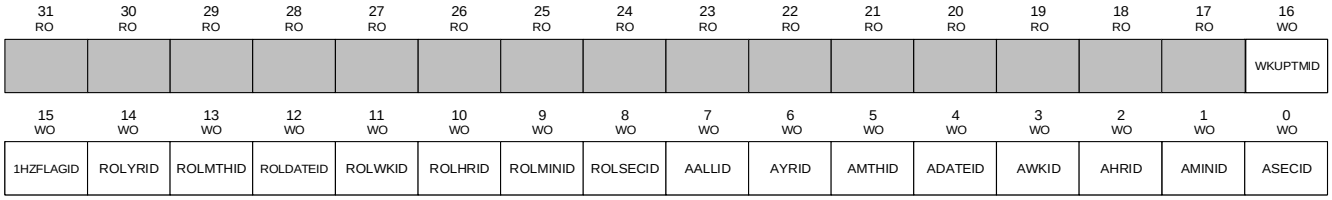
4.14.3.10 RTC_IER - RTC INTERRUPT ENABLE REGISTER



Offset: 0x0024

Bits	Name	Type	Reset	Description
31: 17	*reserved*	N/A	0	Reserved
16	*WKUP_TM*	WO	0	RTC Wakeup Counter Time Match Interrupt Enable
15	*1HZFLAGIE*	WO	0	RTC Trigger 1Hz Flag Interrupt Enable This bit is to control the output of signal RTC_CLK. 1: For every one second, an interrupt is enabled.
14	*ROLYRIE*	WO	0	Rollover Year Interrupt Enable 1: Year rollover interrupt is enabled.
13	*ROLMTHIE*	WO	0	Rollover Month Interrupt Enable 1: Month rollover interrupt is enabled.
12	*ROLDATEIE*	WO	0	Rollover Date Interrupt Enable 1: Date rollover interrupt is enabled.
11	*ROLWKIE*	WO	0	Rollover Week Interrupt Enable 1: Week rollover interrupt is enabled.
10	*ROLHRIE*	WO	0	Rollover Hour Interrupt Enable 1: Hour rollover interrupt is enabled.
9	*ROLMINIE*	WO	0	Rollover Minute Interrupt Enable 1: Minute rollover interrupt is enabled.
8	*ROLSECIE*	WO	0	Rollover Second Interrupt Enable 1: Second rollover interrupt is enabled.
7	*AALLIE*	WO	0	Alarm All Interrupt Enable 1: When RTC_TIME and RTC_CAL are matched with the setting in RTC_ALMTIME and RTC_ALMCAL, the interrupt is enabled.
6	*AYRIE*	WO	0	Year Alarm Interrupt Enable 1: Interrupt is enabled is generated when the alarm year value is matched.
5	*AMTHIE*	WO	0	Month Alarm Interrupt Enable 1: Interrupt is enabled is generated when the alarm month value is matched.
4	*ADATEIE*	WO	0	Date Alarm Interrupt Enable 1: Interrupt is enabled when the alarm date value is matched.
3	*AWKIE*	WO	0	Week Alarm Interrupt Enable 1: Interrupt is enabled when the alarm week value is matched.
2	*AHRIE*	WO	0	Hour Alarm Interrupt Enable 1: Interrupt is enabled when the alarm hour value is matched.
1	*AMINIE*	WO	0	Minute Alarm Interrupt Enable 1: Interrupt is enabled when the alarm minute value is matched.
0	*ASECIE*	WO	0	Second Alarm Interrupt Enable 1: Interrupt is enabled when the alarm second value is matched.

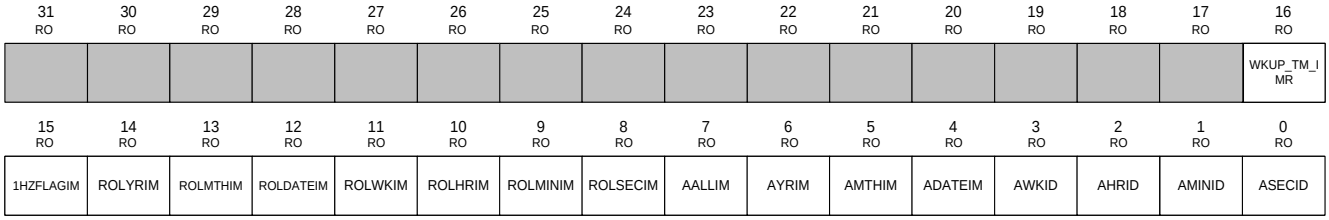
4.14.3.11 RTC_IDR - RTC INTERRUPT DISABLE REGISTER



Offset: 0x0028

Bits	Name	Type	Reset	Description
31: 17	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	*WKUPTMID*	WO	0	RTC Wakeup Counter Time Match Interrupt Disable 1: Interrupt is disabled when the wakeup counter value match the configured wakeup value.
15	*1HZFLAGID*	WO	0	RTC Trigger 1Hz Flag Interrupt Disable This bit is related to the signal RTC_CLK. 1: For every one second, an interrupt is disabled.
14	*ROLYRID*	WO	0	Rollover Year Interrupt Disable 1: Year rollover interrupt is disabled.
13	*ROLMTHID*	WO	0	Rollover Month Interrupt Disable 1: Month rollover interrupt is disabled.
12	*ROLDATEID*	WO	0	Rollover Date Interrupt Disable 1: Date rollover interrupt is disabled.
11	*ROLWKID*	WO	0	Rollover Week Interrupt Disable 1: Week rollover interrupt is disabled.
10	*ROLHRID*	WO	0	Rollover Hour Interrupt Disable 1: Hour rollover interrupt is disabled.
9	*ROLMINID*	WO	0	Rollover Minute Interrupt Disable 1: Minute rollover interrupt is disabled.
8	*ROLSECID*	WO	0	Rollover Second Interrupt Disable 1: Second rollover interrupt is disabled.
7	*AALLID*	WO	0	Alarm All Interrupt Disable 1: When RTC_TIME and RTC_CAL are matched with the setting in RTC_ALMTIME and RTC_ALMCAL, the interrupt is disabled.
6	*AYRID*	WO	0	Year Alarm Interrupt Disable 1: Interrupt is disabled is generated when the alarm year value is matched.
5	*AMTHID*	WO	0	Month Alarm Interrupt Disable 1: Interrupt is disabled is generated when the alarm month value is matched.
4	*ADATEID*	WO	0	Date Alarm Interrupt Disable 1: Interrupt is disabled when the alarm date value is matched.
3	*AWKID*	WO	0	Week Alarm Interrupt Disable 1: Interrupt is disabled when the alarm week value is matched.
2	*AHRID*	WO	0	Hour Alarm Interrupt Disable 1: Interrupt is disabled when the alarm hour value is matched.
1	*AMINID*	WO	0	Minute Alarm Interrupt Disable 1: Interrupt is disabled when the alarm minute value is matched.
0	*ASECID*	WO	0	Second Alarm Interrupt Disable 1: Interrupt is disabled when the alarm second value is matched.

4.14.3.12 RTC_IMR - RTC INTERRUPT MASK REGISTER

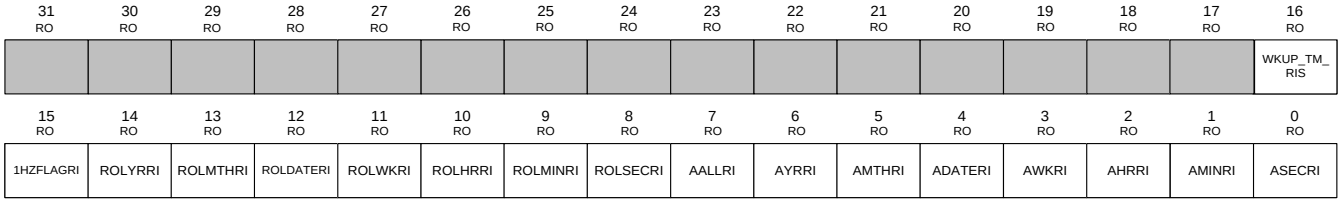


Offset: 0x002C

Bits	Name	Type	Reset	Description
31: 17	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	*WKUP_TM_IMR*	RO	0	RTC Wakeup Counter Time Match Interrupt Mask
15	*1HZFLAGIM*	RO	0	RTC Trigger 1Hz Flag Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
14	*ROLYRIM*	RO	0	Rollover Year Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
13	*ROLMTHIM*	RO	0	Rollover Month Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
12	*ROLDATEIM*	RO	0	Rollover Date Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
11	*ROLWKIM*	RO	0	Rollover Week Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
10	*ROLHRIM*	RO	0	Rollover Hour Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
9	*ROLMINIM*	RO	0	Rollover Minute Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
8	*ROLSECIM*	RO	0	Rollover Second Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
7	*AALLIM*	RO	0	Alarm All Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
6	*AYRIM*	RO	0	Year Alarm Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
5	*AMTHIM*	RO	0	Month Alarm Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
4	*ADATEIM*	RO	0	Date Alarm Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.

Bits	Name	Type	Reset	Description
3	*AWKID*	RO	0	Week Alarm Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
2	*AHRID*	RO	0	Hour Alarm Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
1	*AMINID*	RO	0	Minute Alarm Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
0	*ASECID*	RO	0	Second Alarm Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.

4.14.3.13 RTC_RIS - RTC RAW INTERRUPT STATUS REGISTER



Offset: 0x0030

Bits	Name	Type	Reset	Description
31: 17	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	*WKUP_TM_RIS*	RO	0	RTC Wakeup Counter Time Match. Raw Interrupt status.
15	*1HZFLAGRI*	RO	0	RTC Trigger 1Hz Flag Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
14	*ROLYRRI*	RO	0	Rollover Year Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
13	*ROLMTHRI*	RO	0	Rollover Month Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
12	*ROLDATERI*	RO	0	Rollover Date Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
11	*ROLWKRI*	RO	0	Rollover Week Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
10	*ROLHRRI*	RO	0	Rollover Hour Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
9	*ROLMINRI*	RO	0	Rollover Minute Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
8	*ROLSECRI*	RO	0	Rollover Second Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
7	*AALLRI*	RO	0	Alarm All Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
6	*AYRRI*	RO	0	Year Alarm Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
5	*AMTHRI*	RO	0	Month Alarm Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
4	*ADATERI*	RO	0	Date Alarm Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

Bits	Name	Type	Reset	Description
3	*AWKRI*	RO	0	Week Alarm Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
2	*AHRRI*	RO	0	Hour Alarm Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
1	*AMINRI*	RO	0	Minute Alarm Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
0	*ASECRI*	RO	0	Second Alarm Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.14.3.14 RTC_ISC - RTC INTERRUPT STATUS AND CLEAR REGISTER

Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C
															WKUP_TM_ISC

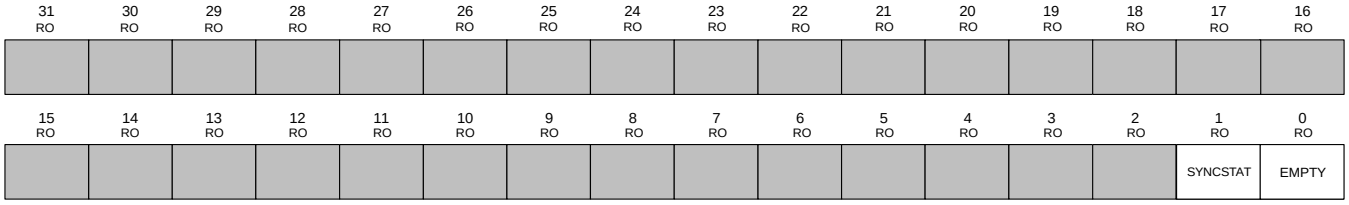
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
1HZFLAGIS	RYRIS	RMTHIS	RDATEIS	RWKIS	RHRIS	RMINIS	RSECIS	AALLIS	AYRIS	AMTHIS	ADATEIS	AWKIS	AHRIS	AMINIS	ASECIS

Offset: 0x0034

Bits	Name	Type	Reset	Description
31: 17	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	*WKUP_TM_ISC*	R/W1C	0	RTC Wakeup Counter Time Match. Interrupt status and Clear.
15	*1HZFLAGIS*	R/W1C	0	RTC Trigger 1Hz Flag Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
14	*RYRIS*	R/W1C	0	Rollover Year Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
13	*RMTHIS*	R/W1C	0	Rollover Month Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
12	*RDATEIS*	R/W1C	0	Rollover Date Raw Interrupt Status 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
11	*RWKIS*	R/W1C	0	Rollover Week Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
10	*RHRIS*	R/W1C	0	Rollover Hour Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
9	*RMINIS*	R/W1C	0	Rollover Minute Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
8	*RSECIS*	R/W1C	0	Rollover Second Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
7	*AALLIS*	R/W1C	0	Alarm All Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
6	*AYRIS*	R/W1C	0	Year Alarm Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
5	*AMTHIS*	R/W1C	0	Month Alarm Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.

Bits	Name	Type	Reset	Description
4	*ADATEIS*	R/W1C	0	Date Alarm Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
3	*AWKIS*	R/W1C	0	Week Alarm Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
2	*AHRIS*	R/W1C	0	Hour Alarm Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
1	*AMINIS*	R/W1C	0	Minute Alarm Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
0	*ASECIS*	R/W1C	0	Second Alarm Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.

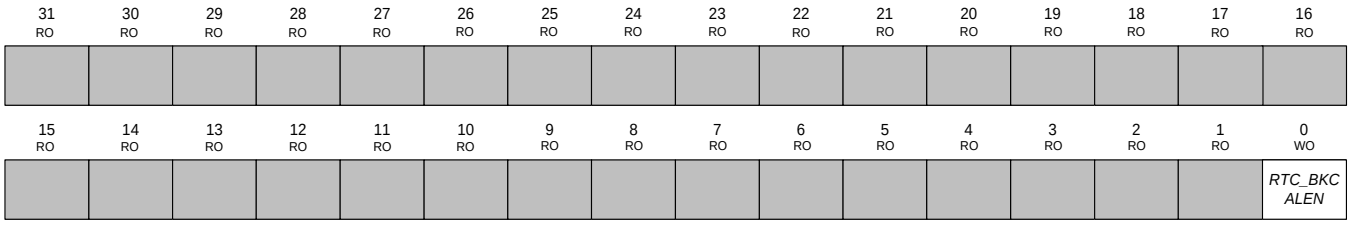
4.14.3.15 RTC_STATUS - RTC STATUS REGISTER



Offset: 0x0040

Bits	Name	Type	Reset	Description
31:2	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*SYNCSTAT*	RO	0	RTC Synchronization Status 0: Synchronization of RTC is on-going 1: Synchronization of RTC is done
0	*EMPTY*	RO	0	RTC_TIME and RTC_CAL Status 0: **RTC_TIME** and **RTC_CAL** are not empty 1: **RTC_TIME** and **RTC_CAL** are empty

4.14.3.16 RTC_BK_CAL - RTC READ BACKUP CALENDAR REGISTER



Offset: 0x0044

Bits	Name	Type	Reset	Description
31:1	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*RTC_BKCALEN*	WO	0	Read RTC Backup Register Enable 0 :Read RTCTIME & RTCCAL From control register. 1 :Read RTCTIME & RTCCAL From backup register.

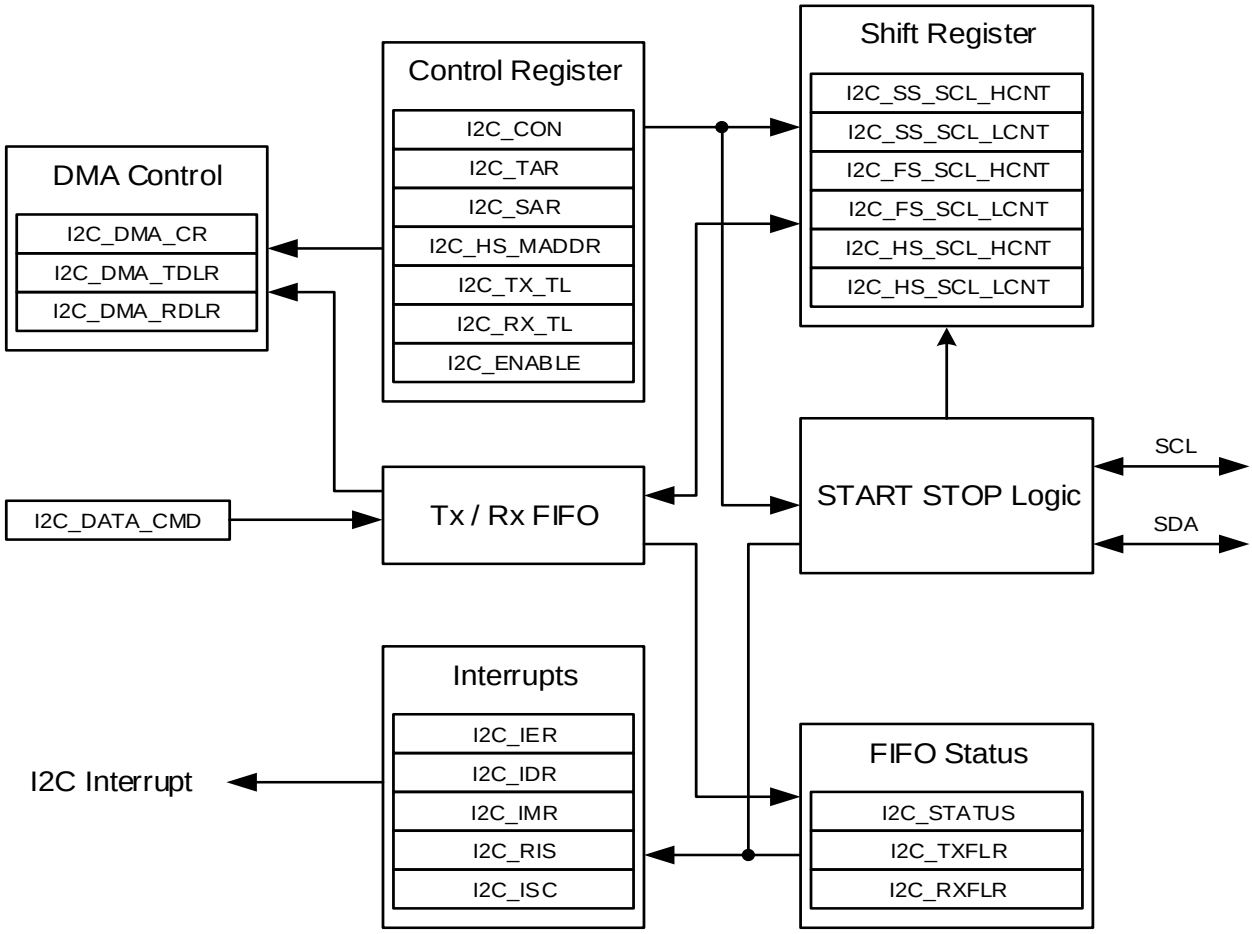
4.15 INTER INTEGRATED CIRCUIT (I2C)

I2C is a two-wire, bi-directional serial bus that provides a simple and effective method of data exchange between devices. The I2C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously. Data is transferred between a Master and a Slave synchronously to serial clock (SCK) on the serial data (SDA) line on a byte-by-byte basis. Each data byte is 8-bit long.

- Double wire serial transmission interface.
- Selectable transmission speed:
  - Standard Mode : 100 Kb/s
  - Fast Mode : 400 Kb/s
  - High Speed Mode : 3.4 Mb/s
- Master / Slave operation supports both transmitting and receiving data
- Support restart function.
- Support multi-master operation.
- Support 7- or 10-bit address.
- Support 7- or 10-bit transmission format.
- Support General Call and Start Byte.
- DMA handshaking interface

4.15.1 BLOCK DIAGRAM

Figure 4.15-1: I2C Block Diagram



4.15.2 FUNCTIONAL DESCRIPTION

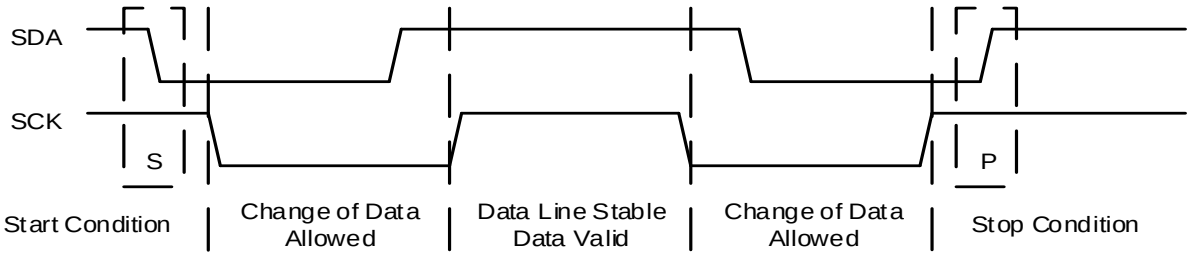
4.15.2.1 I2C BUS PROTOCOL

START AND STOP CONDITION PROTOCOL

The I2C specification defines a Start condition as a transition of the SDA line from HIGH to LOW state, while the SCK line is high. A Start condition is always generated by the master and signifies the transition of the bus from an Idle to an Active state. There is one SCK clock pulse for each data bit with the MSB being transmitted first. An acknowledge bit follows each transferred byte. A transition on the SDA line while SCK is high is interpreted as a command (START or STOP). Each bit is sampled during the high period of SCK; therefore, the SDA line may be changed only during the low period of SCK and must be held stable during the high period of SCK.

When the bus is IDLE both the SCK and SDA signals are pulled high through external pull-up resistors on the bus. When the master wants to start a transmission on the bus, the master issues a START condition. This is defined to be a high-to-low transition of the SDA signal while SCK is 1. When the master wants to terminate the transmission, the master issues a STOP condition. This is defined to be a Low-to-high transition of the SDA line while SCK is 1. The following figure shows the timing of the START and STOP conditions. When data is being transmitted on the bus, the SDA line must be stable when SCK is1.

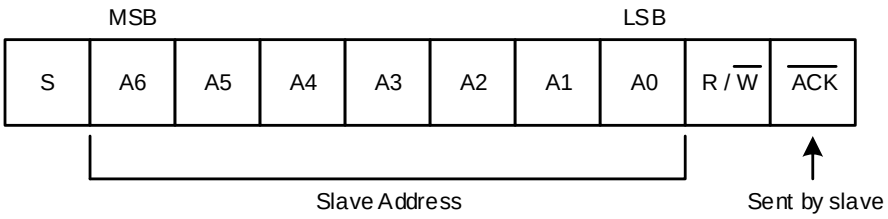
Figure 4.15-2: Start and Stop Condition



I2C ADDRESSING SLAVE PROTOCOL

There are two address formats: the 7-bit address format and the 10-bit address format. During the 7-bit address format, the first seven bits (bits 7:1) of the first byte set the slave address and the LSB bit (bit 0)is the R/W bit. When Bit 8 is set to 0, the master writes to the slave. When Bit 8(R/W) is set to 1, the master reads from the slave. Data is transmitted most significant bit (MSB) first. During 10-bit addressing, two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first five bits (bits 7:3) notify the slaves that this is a10-bit transfer followed by the next two bits (bits 2:1), which set the slaves address bits 9:8, and the LSB bit (Bit 8) is the R/W bit. The second byte transferred sets bits 7:0 of the slave address.

Figure 4.15-3: bit address format

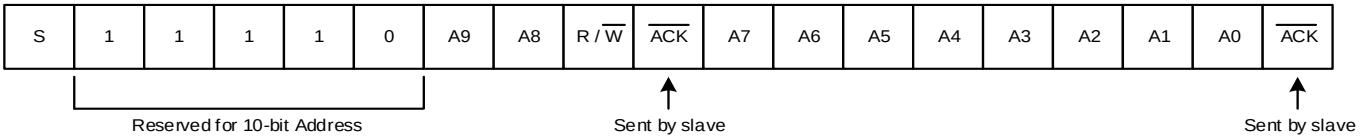


S = start condition

R / $\overline{W}$ = Read / Write Pulse

$\overline{ACK}$ = Acknowledge

Figure 4.15-4: 10-bit address format



S = start condition
R / $\overline{W}$ = Read / Write Pulse
$\overline{ACK}$ = Acknowledge

Table 4.15-1: Definition of Bits in First Byte

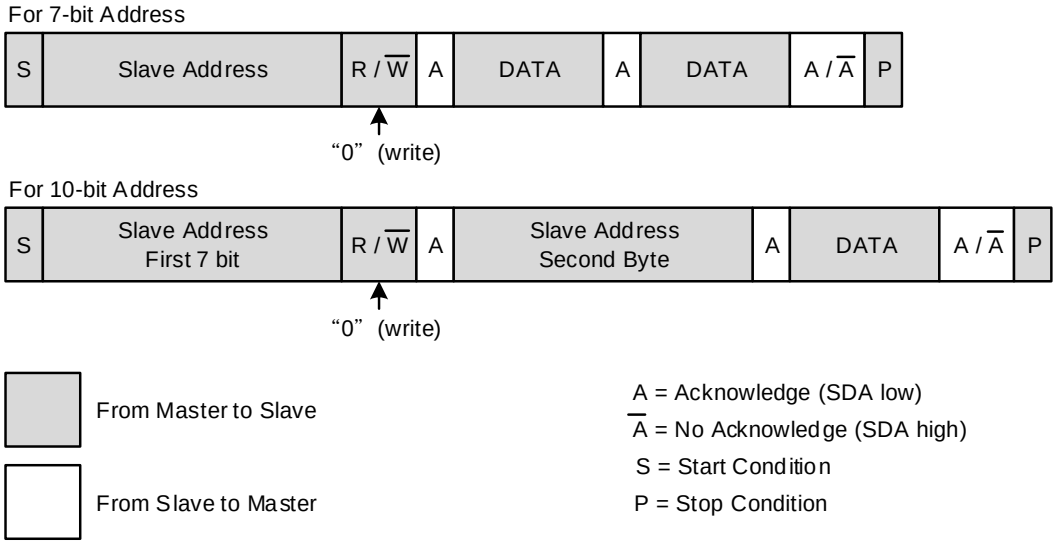
Slave Address	R/W Bit	Description
0000 000	0	General Call Address.
0000 000	1	START byte
0000 001	X	CBUS address
1111 0XX	X	10-bit slave addressing

I2C TRANSMITTING AND RECEIVING PROTOCOL

All data is transmitted in byte format, with no limit on the number of bytes transferred per data transfer. After the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the acknowledge signal. When a slave-receiver does not respond with an acknowledge pulse, the master aborts the transfer by issuing a STOP condition. The slave shall leave the SDA line high so the master can abort the transfer.

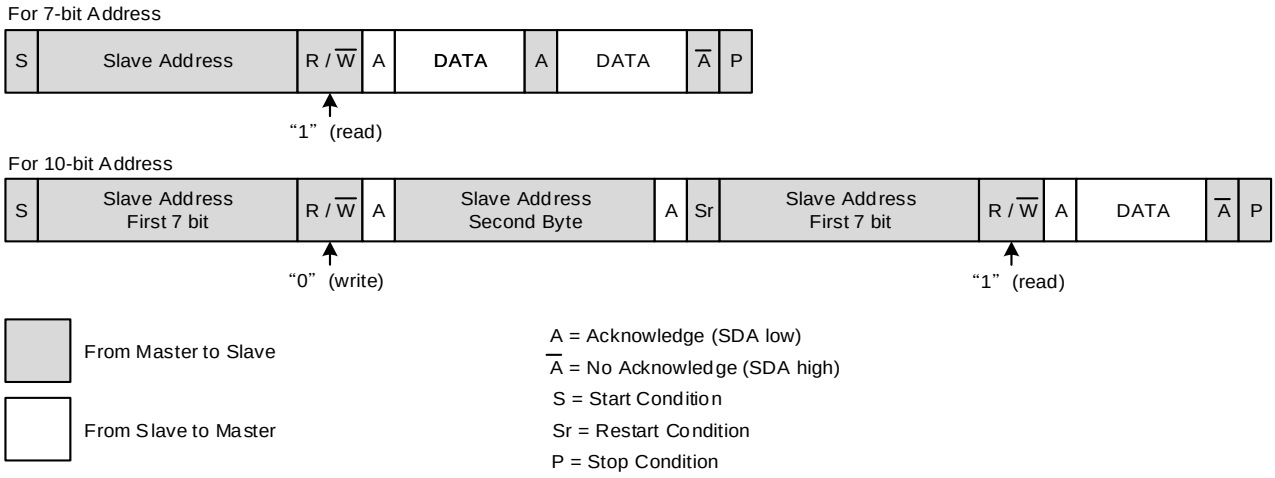
If the master-transmitter is transmitting data, then the slave-receiver responds to the master-transmitter with an acknowledge pulse after every byte of data is received. The transmitting format as follows:

Figure 4.15-5: Master - Transmitter Protocol



If the master is receiving data, then the master responds to the slave-transmitter with an acknowledge pulse after a byte of data has been received, except for the last byte. This is the way the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge so that the master can issue a STOP condition.

Figure 4.15-6: Master - Receiver Protocol



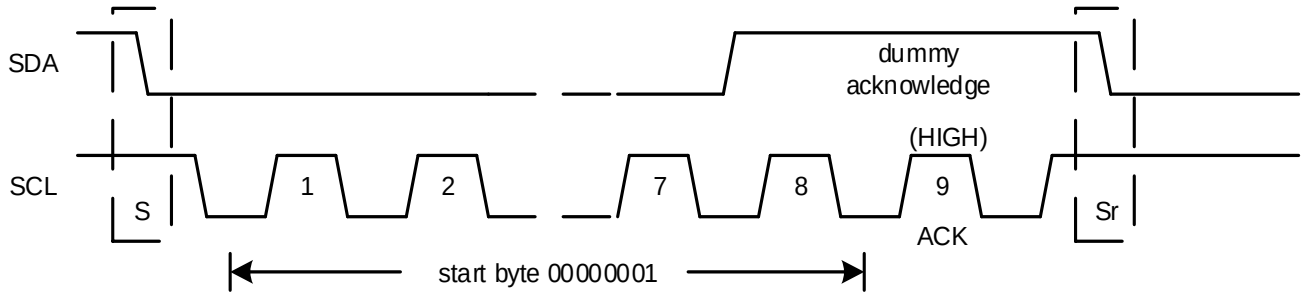
I2C START BYTE TRANSFER PROTOCOL

The START BYTE transfer protocol is set up for systems that do not have an on board dedicated I2Chardware module. When the I2C is addressed as a slave, it always samples the I2C bus at the highest speed supported so that it never requires a START BYTE transfer. However, when theI2C is a master, it supports the generation of START BYTE transfers at the beginning of every transfer in case a slave device requires it. The START BYTE protocol consists of seven zero being transmitted followed by a 1, as illustrated by the follows. This allows the processor that is polling the bus to under-sample the address phase until 0 is detected. Once the microcontroller detects a 0, it switches from the under sampling rate to the correct rate of the master.

The START BYTE procedure is as follows:

1. Master generates a START condition.
2. Master transmits the START byte (0000 0001).
3. Master transmits the ACK clock pulse.
4. No slave sets the ACK signal to 0.
5. Master generates a repeated START (Sr) condition.

Figure 4.15-7: Start Byte Transfer



I2C TRANSMISSION SPEED ALGORITHM INTRODUCED

Before starting any I2C bus transaction, the SCK clock duty cycle must be set via the following registers

- In Standard Mode(100 Kbps)：The minimum value for High time of the SCK clock is 4000ns and for Low time of the SCK clock is 4700ns.

  SCK waveform cycle is set by the following registers:
  - I2C_SS_SCK_HCNT
  - I2C_SS_SCK_LCNT
- In Fast Mode(400 Kbps)：The minimum value for High time of the SCK clock is 600ns and for Low time of the SCK clock is 1300ns.

  SCK waveform cycle is set by the following registers:
  - I2C_FS_SCK_HCNT
  - I2C_FS_SCK_LCNT
- In High Speed Mode(3.4 Mbps)：

  Loading below 100pF: The High-Cycle minimum is 60ns, The Low-Cycle minimum is 160ns.

  Loading below 400pF: The minimum value for High time of the SCK clock is 120ns and for Low time of the SCK clock is 320ns.

  SCK waveform cycle is set by the following registers:
  - I2C_HS_SCK_HCNT
  - I2C_HS_SCK_LCNT

Note: In the high speed mode, because host transfer the master code will use the fast mode define, so the fast mode register (**I2C_FS_SCK_HCNT** & **I2C_FS_SCK_LCNT**) also want to setting.

The Algorithm is as follows:

For Example, it system frequency is 16MHz, I2C is selected fast mode.

High time of the SCK clock:

- 600ns = 62.5ns (16MHz) * I2C_FS_SCK_HCNT => I2C_FS_SCK_HCNT set by 10(0xA)

Low time of the SCK clock:

1300ns = 62.5ns (16MHz) * I2C_FS_SCK_LCNT => I2C_FS_SCK_LCNT set by21 (0x15)

4.15.2.2 INTERRUPTS

The interrupt in I2C are controlled by a set of five registers.

- Interrupt Control (IER, IDR, IMR)

  I2C Interrupt enable register **(I2C_IER)** enables the interrupt request lines by writing a '1'. Similarly, I2C Interrupt disable register **(I2C_IDR)** disables the interrupt request lines by writing a '1'. IER and IDR are write only register, the overall result of the above registers can be shown by WDT Interrupt Mask Register **(I2C_IMR).**IMR is a read-only register using '1' or '0' to indicate if the interrupt request line is enabled/ or disabled.

- Interrupt Status Read (RIS)

  I2C Raw Interrupt Status **(I2C_RIS)** is a read-only register to read the interrupt status from the module. Bits in this register show the true status of I2C. The I2C can generate interrupts when the following conditions are observed:

  - Timeout occurs
  - General Call address is received and it is acknowledged
  - Start condition on bus detected
  - Stop condition on bus detected
  - I2C activity captured
  - Slave transmission done
  - Transmit abort
  - Slave transmission requested
  - Transmit buffer empty
  - Transmit buffer overflow
  - Receive buffer full
  - Receive buffer overflow
  - Receive buffer underflow

- Interrupt Clear (ISC)

  I2C Interrupt Status & Interrupt Clear Register **(I2C_ISC)** is used to read the masked interrupt status of the module, showing which interrupt is unmasked. Each bit in ISC is the logical AND of the respective bits in RIS and IMR. Writing a '1' to the bit in this register can clear the corresponding interrupt.

4.15.2.3 OPERATING MODES

I2C can operate in four different modes: Slave Transmitter, Slave Receiver, Master Transmitter, or Master Receiver to support data transfer as a master and as a slave.

A master initiates a data transfer and generates the clock signal SCL. Any device addressed by a master is considered a slave. This section introduces these modes and their programming model.

Note: I2C should be set to operate only as an I2C master or as an I2C slave, never set both simultaneously.

SLAVE MODE

Initial Configuration:

To use I2C as a slave, perform the following steps:

1. Disable the I2C by writing a 0 to the I2C_ENABLE.ENABLE bit.
2. Write to the I2C_SAR register set the slave address. This is the address to which the I2C responds. Note that this step is not necessary if the reset value for the I2C slave address is chosen.
3. Write to the I2C_CTRL register to specify the type of addressing (7- or 10-bit depends on needs) and whether the I2C acts as master or slave mode.
4. Enable the I2C by writing a 1 to the I2C_ENABLE.ENABLE bit.

Slave Transmitter Mode

In this mode, the I2C is a slave and transmit data to a master. This mode is entered when the slave address transmitted by the master is identical to its own address with a set R/ $\overline{W}$ (i.e. R/ $\overline{W}$ = 1). The slave transmitter shifts the serial data out on SDA with the clock pulses generated by the master device. The slave device does not generate the clock.

When the I2C is acting as slave-transmitter, following steps occur:

1. The other I2C master device initiates an I2C transfer with an address that matches the slave address in the I2C_SAR register.
2. The I2C acknowledge the sent address and recognizes the direction of data transfer indicating that it is acting as a slave-transmitter.
3. The I2C asserts the RDREQRI in the I2C_RIS register and waits for software to respond.
4. If there is any data remaining in the Tx FIFO before receiving the read request, then the I2C asserts a TXABRTRI in the I2C_RIS register to flush the old data from the Tx FIFO.
5. Software then writes to the I2C_DATA_CMD.DAT field with the data to be written and writes 0 to I2C_DATA_CMD.CMD.
6. Software must clear the RDREQRI and TXABRTRI by writing a 1 to their corresponding bits - RDREQIS and TXABRTIS in the I2C_ISC register
7. The I2C transmits the byte.
8. The master may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

Slave Receiver Mode

In this mode, the I2C is a slave and receive data from a master. Slave receiver mode is entered when the slave address transmitted by the master is identical to its own address and cleared R/ $\overline{W}$ bit is received (i.e. R/ $\overline{W}$ = 0). In slave receiver mode, serial data bits received on SDA are shifted in with the clock pulse generated by the master device.

When the I2C is acting as slave- receiver, following steps occur:

1. The other master initiates an I2C transfer with an address in the I2C_SAR register.
2. The I2C acknowledges the sent address and recognizes the direction of data transfer indicating that it is acting as a slave-receiver.
3. The I2C receives the transmitted byte and place it in the receive buffer.
4. The status and interrupt bits corresponding to the receive buffer is updated.
5. Software must read the DAT from the I2C_DATA_CMD register.
6. The other master may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

MASTER MODE

Initial Configuration

To use I2C as a master, perform the following steps:

1. Disable the I2C by writing a 0 to the I2C_ENABLE.ENABLE bit.
2. Write to the I2C_CTRL register to set the maximum speed mode supported for slave operation and whether the I2C starts its transfers in 7- or 10-bit addressing mode when the device is a slave.
3. Write to the I2C_TAR register the address of the I2C device to be addressed. It also indicates whether adding a START byte or issuing a general call is going to occur.
4. Write to the IC_HS_MADDR register the desired master code for I2C. Note that this step is only applicable for high-speed mode transfers.
5. Enable the I2C by writing a 1 to the I2C_ENABLE.ENABLE bit.
6. Commands and data to be sent may be written now to the I2C_DATA_CMD register. If the I2C_DATA_CMD register is written before the I2C is enabled, the data and commands are lost as the buffers are kept cleared when I2C is not enabled.

Master Transmitter and Master Receiver Mode

The I2C supports switching back and forth between reading and writing dynamically. To transmit data, write the data to be written to the *DAT* field of the **I2C_DATA_CMD** register. **I2C_DATA_CMD.***CMD* should be written to 0 for write operations. Subsequently, a read command may be issued by writing "don't cares" to the *DAT* field of the **I2C_DATA_CMD** register, and a 1 should be written to the *CMD* bit.

4.15.2.4 DMA CONTROLLER

I2C in PT32C302 supports DMA signaling to indicate when data is ready to be read or when the transmit FIFO needs data. System software can choose the DMA burst mode by programming an appropriate value into the threshold registers.

To enable the DMA controller interface on the I2C controller, you must write to the DMA control register (**I2C_DMA_CTRL**) bits. Writing a 1 into the *TXEN* bit field of **I2C_DMA_CTRL** register enables the I2C controller transmit handshaking interface. Writing a 1 into the *RXEN* bit field of the **I2C_DMA_CTRL** register enables the I2C controller receive handshaking interface.

4.15.3 I2C REGISTER MAP

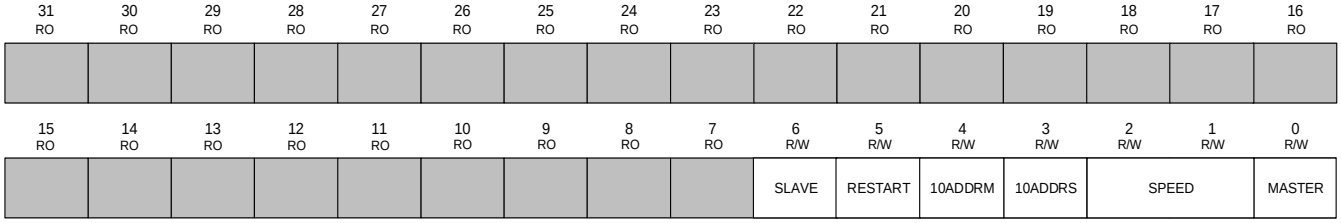
I2C Base Address:

- I2C0: 0x4800_8000
- I2C1: 0x4800_8100

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	I2C_CON	R/W	0x0000_0025	I2C Control Register	370
0x0004	I2C_TAR	R/W	0x0000_0055	I2C Target Address Register	371
0x0008	I2C_SAR	R/W	0x0000_0055	I2C Slave Address Register	371
0x000C	I2C_HS_MADDR	R/W	0x0000_0000	I2C Master Code Address Register	372
0x0010	I2C_DATA_CMD	R/W	0x0000_0000	I2C Tx/Rx Data Buffer and Commend	372
0x0014	I2C_SS_SCL_HCNT	R/W	0x0000_0040	I2C Standard Speed Clock SCL High Count	373
0x0018	I2C_SS_SCL_LCNT	R/W	0x0000_004B	I2C Standard Speed Clock SCL Low Count	373
0x001C	I2C_FS_SCL_HCNT	R/W	0x0000_000A	I2C Fast Speed Clock SCL High Count	374
0x0020	I2C_FS_SCL_LCNT	R/W	0x0000_0015	I2C Fast Speed Clock SCL Low Count	374
0x0024	I2C_HS_SCL_HCNT	R/W	0x0000_0001	I2C High Speed Clock Mode SCL High Count	375
0x0028	I2C_HS_SCL_LCNT	R/W	0x0000_0007	I2C High Speed Clock Mode SCL Low Count	375
0x002C	I2C_IER	WO	0x0000_0000	I2C Interrupt Enable Register	376
0x0030	I2C_IDR	WO	0x0000_0000	I2C Interrupt Disable Register	377
0x0034	I2C_IMR	RO	0x0000_0000	I2C Interrupt Mask Status Register	378
0x0038	I2C_RIS	RO	0x0000_0000	I2C Raw Interrupt Status Register	379
0x003C	I2C_ISC	R/W1C	0x0000_0000	I2C Interrupt Status and Clear Register	381
0x0040	I2C_RX_TL	R/W	0x0000_0000	I2C Receive FIFO Threshold Register	382
0x0044	I2C_TX_TL	R/W	0x0000_0000	I2C Transmit FIFO Threshold Register	382
0x048	reserved	N/A	0x0000_0000	Reserved	-
0x04C	reserved	N/A	0x0000_0000	Reserved	-
0x050	reserved	N/A	0x0000_0000	Reserved	-
0x04C	reserved	N/A	0x0000_0000	Reserved	-
0x050	reserved	N/A	0x0000_0000	Reserved	-
0x054	reserved	N/A	0x0000_0000	Reserved	-
0x060	reserved	N/A	0x0000_0000	Reserved	-
0x064	reserved	N/A	0x0000_0000	Reserved	-
0x068	reserved	N/A	0x0000_0000	Reserved	-
0x006C	I2C_ENABLE	R/W	0x0000_0000	I2C Enable Register	383
0x0070	I2C_STATUS	RO	0x0000_0006	I2C Status Register	383
0x0074	I2C_TXFLR	RO	0x0000_0000	I2C Transmit FIFO Level Register	384
0x0078	I2C_RXFLR	RO	0x0000_0000	I2C Receive FIFO Level Register	384
0x007C	I2C_TIMEOUT	R/W	0x0000_0000	I2C Timeout Enable Register	385
0x0080	I2C_TX_ABRT	R	0x0000_0000	I2C Transmit Abort Status Register	386
0x0084	reserved	N/A	0x0000_0000	Reserved	-
0x0088	I2C_DMA_CR	R/W	0x0000_0000	I2C DMA Control Register	387
0x008C	I2C_DMA_TDLR	R/W	0x0000_0000	I2C DMA Transmit Threshold Register	388
0x0090	I2C_DMA_RDLR	R/W	0x0000_0000	I2C DMA Receive Threshold Register	388

4.15.3.1 I2C_CTRL - I2C CONTROL REGISTER

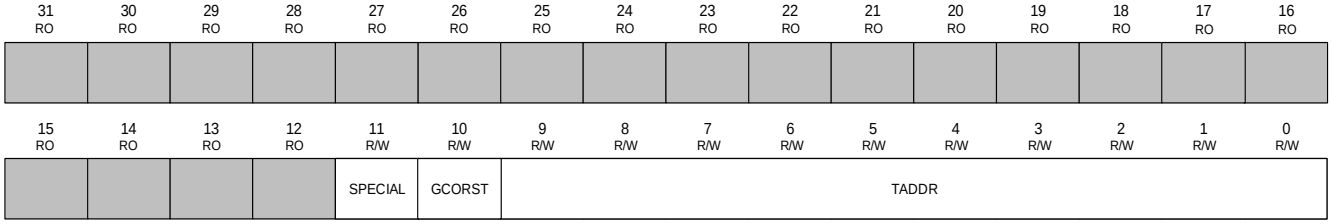
This register can be written only when the I2C is disabled. Writes at other times have no effect.



Offset: 0x0000

Bits	Name	Type	Reset	Description
31: 7	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6	*SLAVE*	R/W	0	I2C Slave Disable This bit controls whether I2C has its slave disabled after reset. 0: Slave is enabled. 1: Slave is disabled.
5	*RESTART*	R/W	1	I2C Restart Enable This bit determines whether restart conditions may be sent when acting as a master. When RESTART is disabled, the master is prohibited from performing the following functions: - Send multiple bytes per transfer (split) - Change direction within a transfer (split) - Send a start byte - Perform any high-speed mode operation - Perform combined format transfer in 10-bit addressing mode - Read operation only under 10-bit address 0: Restart is disabled. 1: Restart is enabled.
4	*10ADDRM*	R/W	0	10 Bit Addressing Master Mode This bit controls whether I2C starts its transfers in 10-bit addressing mode when acting as a master. 0: 7-bit addressing 1: 10-bit addressing
3	*10ADDRS*	R/W	0	10 Bit Addressing Slave Mode When acting as a slave, this bit controls whether I2C responds to 7 or 10 bit addresses. 0: 7-bit addressing 1: 10-bit addressing
2:1	*SPEED*	R/W	0x2	I2C Operating Speed 0: Reserved 1: Standard mode (100 kbit/s) 2: Fast mode (400 kbit/s) 3: High speed mode (3.4 Mbit/s)
0	*MASTER*	R/W	1	I2C Master Disable This bit controls if the I2C master mode is enabled. 0: Master is disabled. 1: Master is enabled.

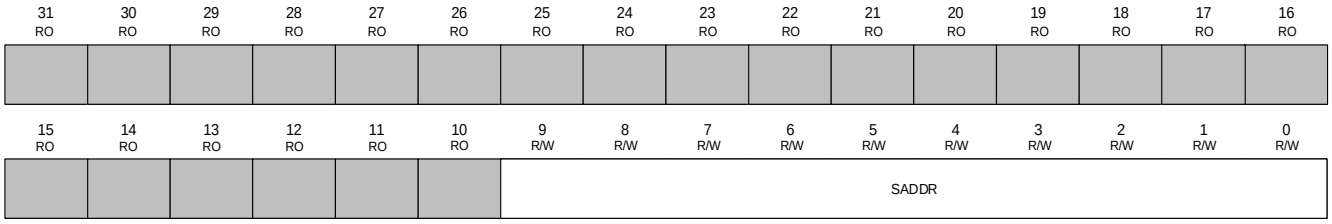
4.15.3.2 I2C_TAR - I2C TARGET ADDRESS REGISTER



Offset: 0x0004

Bits	Name	Type	Reset	Description
31:12	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	*SPECIAL*	R/W	0	Special Command This bit indicates whether software would like to perform a General call or Start byte I2C command. 0: Ignore bit 10 *GCORST* and use **I2C_TAR** normally. 1: Perform special I2C command as specified in *GCORST* bit.
10	*GCORST*	R/W	0	General Call Or Start Byte Command If *SPECIAL* is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the I2C or General Call Address after issuing a General Call, only writes may be performed. I2C remains in general call mode until the *SPECIAL* bit value is cleared. 0: General call address 1: START byte
9:0	*TADDR*	R/W	0x55	I2C Target Address Register This is the target address for any master transactions. When transmitting a General Call, these bits are ignored.

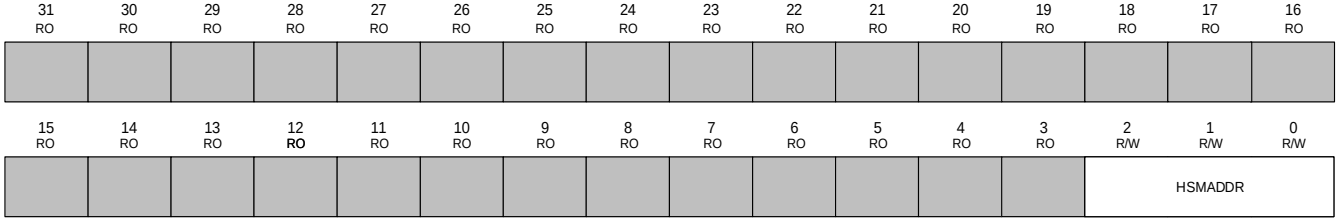
4.15.3.3 I2C_SAR - I2C SLAVE ADDRESS REGISTER



Offset: 0x0008

Bits	Name	Type	Reset	Description
31:10	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
9:0	*SADDR*	R/W	0x55	I2C Slave Address Register This field holds the slave address when the I2C is operating as a slave. For 7-bit addressing, only Field Bits [6:0] of the Slave Address Register are used. This register can be written only when the I2C interface is disabled.

4.15.3.4 I2C_HS_MADDR - I2C HIGH SPEED MASTER CODE ADDRESS REGISTER

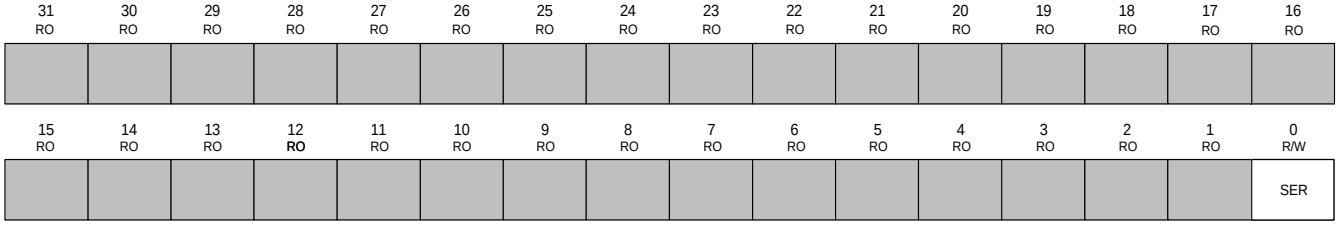


Offset: 0x000C

Bits	Name	Type	Reset	Description
31:3	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*HSMADDR*	R/W	0x1	I2C High Speed Master Code Address This field holds the value of the I2C HS mode master code.

4.15.3.5 I2C_DATA_CMD - I2C TX/RX DATA BUFFER AND COMMAND

This register is valid only when the SPI is configured as a master device. The register enables the individual slave select output lines from the SPI Master.

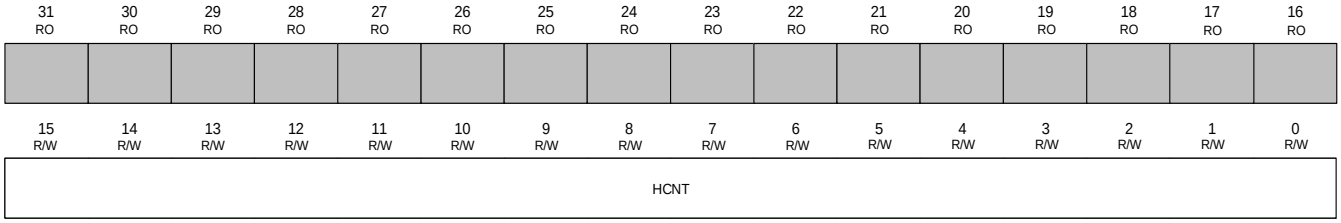


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:9	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
8	*CMD*	R/W	0	Rx/Tx Command This bit controls whether a read or write is performed. This bit does not control the direction when the I2C acts as a slave. It controls only the direction when it acts as a master. In slave-receiver mode, this bit is a 'don't care' because writes to this register are not required. In slave-transmitter mode, a '0' indicates that the CPU data is to be transmitted. 0: Master write 1: Master read
7:0	*DAT*	R/W	0x0	Rx/Tx Data Buffer This register contains the data to be transmitted or received on the I2C bus. Read these bits returns the data received on the I2C interface. Write these bits to send data out on the I2C interface.

4.15.3.6 I2C_SS_SCL_HCNT - I2C STANDARD MODE SCK HIGH COUNT

This register sets the SCL clock high-period count for standard speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing.

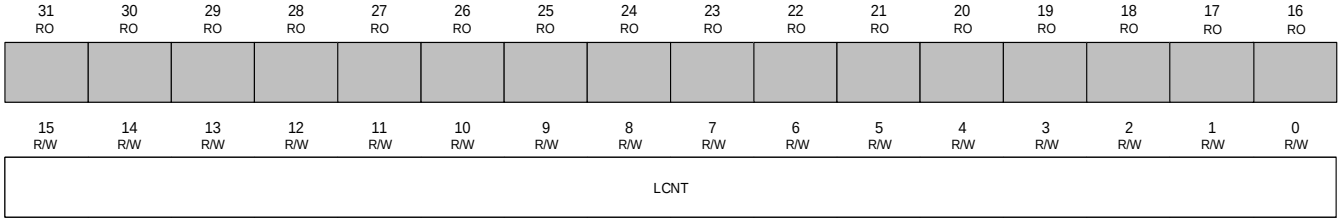


Offset: 0x0014

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*HCNT*	R/W	0x40	Standard Speed I2C Clock SCL High Count These bits set the SCL clock high-period count for standard speed. This bit only can be written when the I2C interface is disabled. Writes at other times have no effect.

4.15.3.7 I2C_SS_SCK_LCNT - I2C STANDARD MODE SCK LOW COUNT

This register sets the SCL clock low-period count for standard speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing.

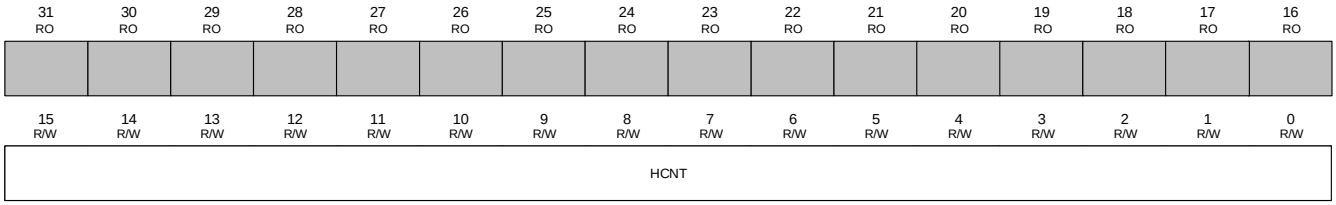


Offset: 0x0018

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*LCNT*	R/W	0x4B	Standard Speed I2C Clock SCL Low Count These bits set the SCL clock low-period count for standard speed. This bit only can be written when the I2C interface is disabled. Writes at other times have no effect.

4.15.3.8 I2C_FS_SCL_HCNT - I2C FAST MODE SCL HIGH COUNT

This register sets the SCL clock high-period count for fast speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing.

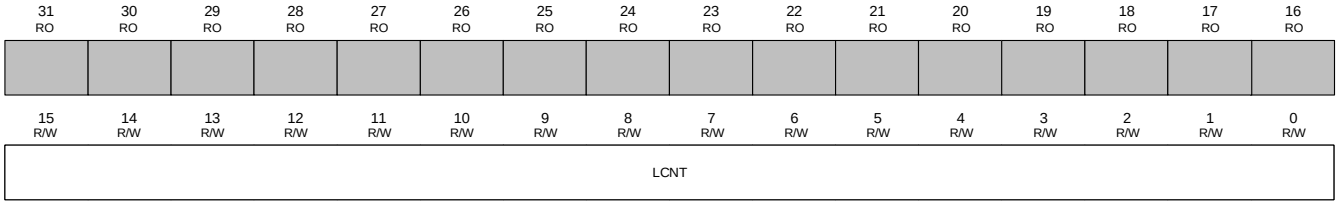


Offset: 0x001C

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*HCNT*	R/W	0x0A	Fast Speed I2C Clock SCL High Count These bits set the SCL clock high-period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. This bit only can be written when the I2C interface is disabled. Writes at other times have no effect.

4.15.3.9 I2C_FS_SCL_LCNT - I2C FAST MODE SCL LOW COUNT

This register sets the SCL clock low period count. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing.

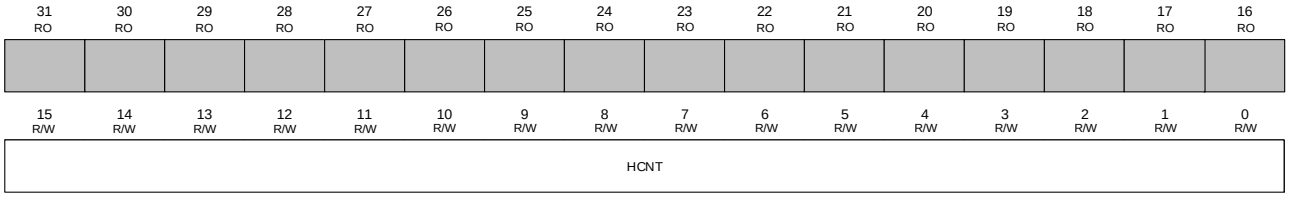


Offset: 0x0020

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*LCNT*	R/W	0x15	Fast Speed I2C Clock SCL Low Count These bits set the SCL clock low-period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. This bit only can be written when the I2C interface is disabled. Writes at other times have no effect.

4.15.3.10 I2C_HS_SCL_HCNT - I2C HIGH SPEED MODE SCL HIGH COUNT

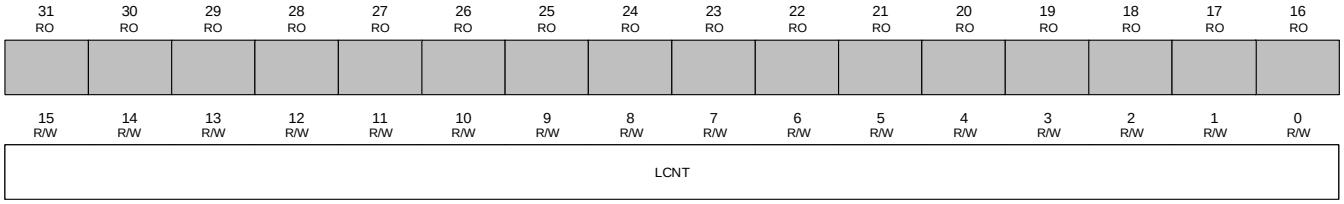
This register sets the SCL clock high-period count for high speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing.



Offset: 0x0024

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*HCNT*	R/W	0x01	High Speed I2C Clock SCL High Count These bits set the SCL clock high-period count for high speed. This bit only can be written when the I2C interface is disabled. Writes at other times have no effect.

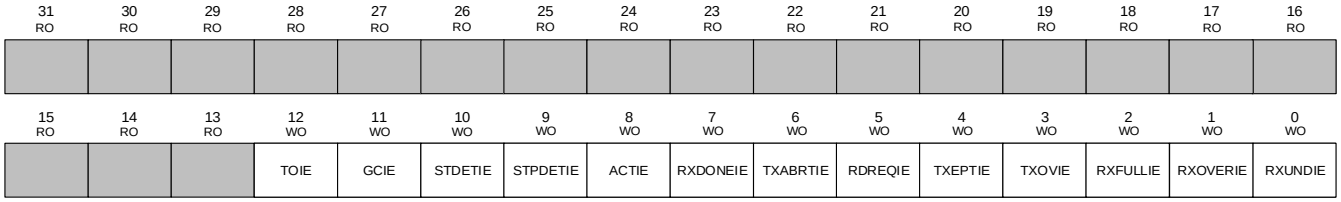
4.15.3.11 I2C_HS_SCL_LCNT - I2C HIGH SPEED MODE SCL LOW COUNT



Offset: 0x0028

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*LCNT*	R/W	0x07	High Speed I2C Clock SCL High Count These bits set the SCL clock low-period count for high speed. This bit only can be written when the I2C interface is disabled. Writes at other times have no effect.

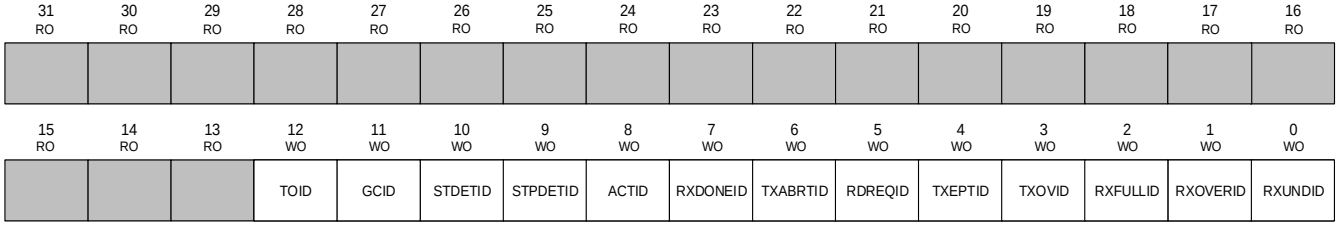
4.15.3.12 I2C_IER - I2C INTERRUPT ENABLE REGISTER



Offset: 0x002C

Bits	Name	Type	Reset	Description
31:13	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
12	*TOIE*	WO	0	Timeout Interrupt Enable 1: Interrupt is enabled.
11	*GCIE*	WO	0	General Call Interrupt Enable 1: Interrupt is enabled.
10	*STDETIE*	WO	0	Start Detection Interrupt Enable 1: Interrupt is enabled.
9	*STPDETIE*	WO	0	Stop Detection Interrupt Enable 1: Interrupt is enabled.
8	*ACTIE*	WO	0	I2C Activity Detection Interrupt Enable 1: Interrupt is enabled.
7	*RXDONEIE*	WO	0	Rx Transmission Done Interrupt Enable 1: Interrupt is enabled.
6	*TXABRTIE*	WO	0	Tx Abort Interrupt Enable 1: Interrupt is enabled.
5	*RDREQIE*	WO	0	Rx Request Data Interrupt Enable 1: Interrupt is enabled.
4	*TXEPTIE*	WO	0	Tx Empty Interrupt Enable 1: Interrupt is enabled.
3	*TXOVIE*	WO	0	Tx Overflow Interrupt Enable 1: Interrupt is enabled.
2	*RXFULLIE*	WO	0	Rx Full Interrupt Enable 1: Interrupt is enabled.
1	*RXOVERIE*	WO	0	Rx Overflow Interrupt Enable 1: Interrupt is enabled.
0	*RXUNDIE*	WO	0	Rx Underflow Interrupt Enable 1: Interrupt is enabled.

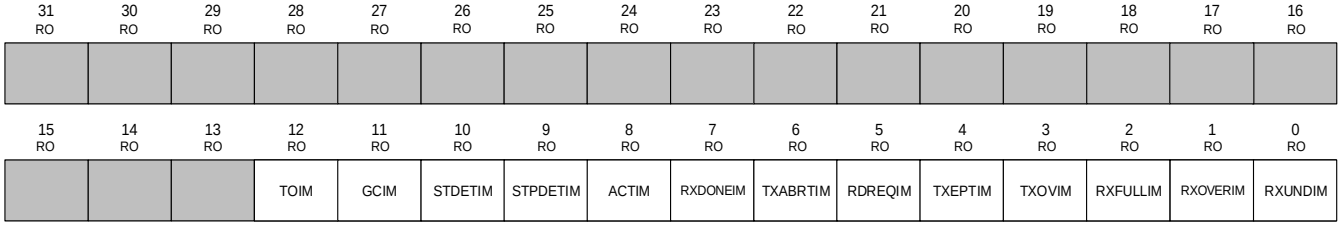
4.15.3.13 I2C_IDR - I2C INTERRUPT DISABLE REGISTER



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:13	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
12	*TOID*	WO	0	Timeout Interrupt Disable 1: Interrupt is disabled.
11	*GCID*	WO	0	General Call Interrupt Disable 1: Interrupt is disabled.
10	*STDETID*	WO	0	Start Detection Interrupt Disable 1: Interrupt is disabled.
9	*STPDETID*	WO	0	Stop Detection Interrupt Disable 1: Interrupt is disabled.
8	*ACTID*	WO	0	I2C Activity Detection Interrupt Disable 1: Interrupt is disabled.
7	*RXDONEID*	WO	0	Rx Transmission Done Interrupt Disable 1: Interrupt is disabled.
6	*TXABRTID*	WO	0	Tx Abort Interrupt Disable 1: Interrupt is disabled.
5	*RDREQID*	WO	0	Rx Request Data Interrupt Disable 1: Interrupt is disabled.
4	*TXEPTID*	WO	0	Tx Empty Interrupt Disable 1: Interrupt is disabled.
3	*TXOVID*	WO	0	Tx Overflow Interrupt Disable 1: Interrupt is disabled.
2	*RXFULLID*	WO	0	Rx Full Interrupt Disable 1: Interrupt is disabled.
1	*RXOVERID*	WO	0	Rx Overflow Interrupt Disable 1: Interrupt is disabled.
0	*RXUNDID*	WO	0	Rx Underflow Interrupt Disable 1: Interrupt is disabled.

4.15.3.14 I2C_IMR - I2C INTERRUPT MASK STATUS REGISTER

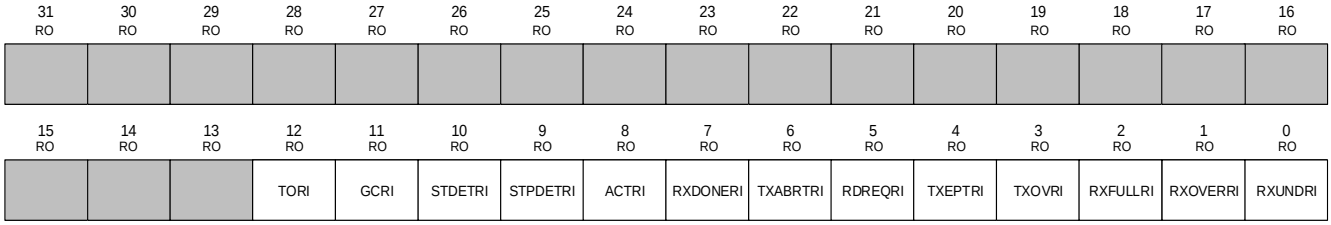


Offset: 0x0034

Bits	Name	Type	Reset	Description
31:13	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
12	*TOIM*	RO	0	Timeout Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
11	*GCIM*	RO	0	General Call Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
10	*STDETIM*	RO	0	Start Detection Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
9	*STPDETIM*	RO	0	Stop Detection Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
8	*ACTIM*	RO	0	I2C Activity Detection Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
7	*RXDONEIM*	RO	0	Rx Transmission Done Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
6	*TXABRTIM*	RO	0	Tx Abort Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
5	*RDREQIM*	RO	0	Rx Request Data Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
4	*TXEPTIM*	RO	0	Tx Empty Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
3	*TXOVIM*	RO	0	Tx Overflow Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
2	*RXFULLIM*	RO	0	Rx Full Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.
1	*RXOVERIM*	RO	0	Rx Overflow Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.

Bits	Name	Type	Reset	Description
0	*RXUNDIM*	RO	0	Rx Underflow Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.

4.15.3.15 I2C_RIS - I2C RAW INTERRUPT STATUS REGISTER



Offset: 0x0038

Bits	Name	Type	Reset	Description
31:13	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
12	*TORI*	RO	0	Timeout Raw Interrupt Status This interrupt is set while the bus is not idle, and that the SCL signal remains low longer than the time configured in the **I2C_TIMEOUT** register 0: No interrupt 1: Timeout interrupt is asserting.
11	*GCRI*	RO	0	General Call Raw Interrupt Status Set only when a General Call address is received and it is acknowledged. 0: No interrupt 1: General call interrupt is asserting.
10	*STDETRI*	RO	0	Start Detection Raw Interrupt Status Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode. 0: No interrupt 1: Start detection interrupt is asserting.
9	*STPDETRI*	RO	0	Stop Detection Raw Interrupt Status Indicates whether a STOP condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode. 0: No interrupt 1: Stop detection interrupt is asserting.
8	*ACTRI*	RO	0	I2C Activity Detection Raw Interrupt Status This bit captures I2C activity and stays set until it is cleared. 0: No interrupt 1: Interrupt is asserting.
7	*RXDONERI*	RO	0	Rx Transmission Done Raw Interrupt Status When the I2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done. 0: No interrupt 1: Interrupt is asserting.

Bits	Name	Type	Reset	Description
6	*TXABRTRI*	RO	0	Tx Abort Raw Interrupt Status This bit indicates if I2C, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave. 0: No interrupt 1: Interrupt is asserting.
5	*RDREQRI*	RO	0	Rx Request Data Raw Interrupt Status This bit is set when I2C is acting as a slave and another I2C master is attempting to read data from I2C. 0: No interrupt 1: Interrupt is asserting.
4	*TXEPTRI*	RO	0	Tx Empty Raw Interrupt Status This bit is set when the transmit buffer is at or below the threshold value set in the **I2C_TX_TL** register. 0: No interrupt 1: Interrupt is asserting.
3	*TXOVRI*	RO	0	Tx Overflow Raw Interrupt Status This bit is set during transmit if the transmit buffer is full and the processor attempts to issue another I2C command by writing to the **I2C_DATA_CMD** register. 0: No interrupt 1: Interrupt is asserting.
2	*RXFULLRI*	RO	0	Rx Full Raw Interrupt Status Set when the receive buffer reaches or goes above the *RXTL* threshold in the **I2C_RX_TL** register. 0: No interrupt 1: Interrupt is asserting.
1	*RXOVERRI*	RO	0	Rx Overflow Raw Interrupt Status Set if the receive buffer is full and an additional byte is received from an external I2C device. The I2C acknowledges this, but any data bytes received after the FIFO is full are lost. 0: No interrupt 1: Interrupt is asserting.
0	*RXUNDRI*	RO	0	Rx Underflow Raw Interrupt Status Set if the processor attempts to read the receive buffer when it is empty by reading from the **I2C_DATA_CMD** register. 0: No interrupt 1: Interrupt is asserting.

4.15.3.16 I2C_ISC - I2C INTERRUPT STATUS AND CLEAR REGISTER

Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt.

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

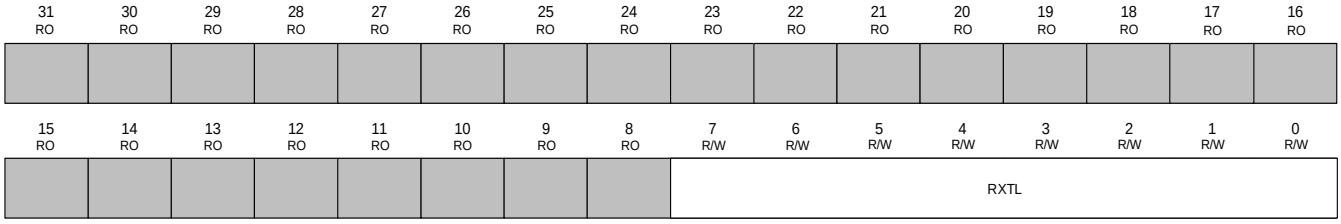
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
			TOIS	GCIS	STDETIS	STPDETIS	ACTIS	RXDONEIS	TXABRTIS	RDREQIS	TXEPTIS	TXOVIS	RXFULLIS	RXOVERIS	RXUNDIS

Offset: 0x003C

Bits	Name	Type	Reset	Description
31:13	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
12	*TOIS*	R/W1C	0	Timeout Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Timeout interrupt has been signaled.
11	*GCIS*	R/W1C	0	General Call Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
10	*STDETIS*	R/W1C	0	Start Detection Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
9	*STPDETIS*	R/W1C	0	Stop Detection Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
8	*ACTIS*	R/W1C	0	I2C Activity Detection Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
7	*RXDONEIS*	R/W1C	0	Rx Transmission Done Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
6	*TXABRTIS*	R/W1C	0	Tx Abort Raw Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
5	*RDREQIS*	R/W1C	0	Rx Request Data Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
4	*TXEPTIS*	R/W1C	0	Tx Empty Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
3	*TXOVIS*	R/W1C	0	Tx Overflow Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
2	*RXFULLIS*	R/W1C	0	Rx Full Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.
1	*RXOVERIS*	R/W1C	0	Rx Overflow Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.

Bits	Name	Type	Reset	Description
0	*RXUNDIS*	R/W1C	0	Rx Underflow Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Interrupt has been signaled.

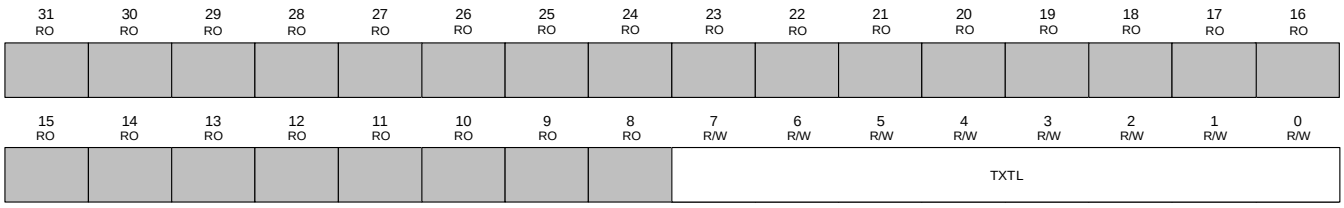
4.15.3.17 I2C_RX_TL - I2C RECEIVE FIFO THRESHOLD REGISTER



Offset: 0x0040

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*RXTL*	R/W	0x0	Receive FIFO Threshold Level This field controls the level of entries that triggers the RX FULL interrupt.

4.15.3.18 I2C_TX_TL - I2C TRANSMIT FIFO THRESHOLD REGISTER

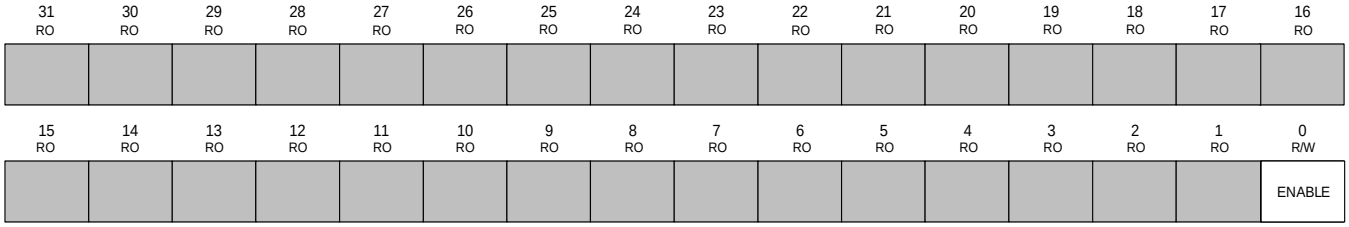


Offset: 0x004C

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*TXTL*	R/W	0x0	Transmit FIFO Threshold Level This field controls the level of entries that triggers the TX EMPTY interrupt.

4.15.3.19 I2C_ENABLE - I2C ENABLE REGISTER

Enable and disable i2c operation

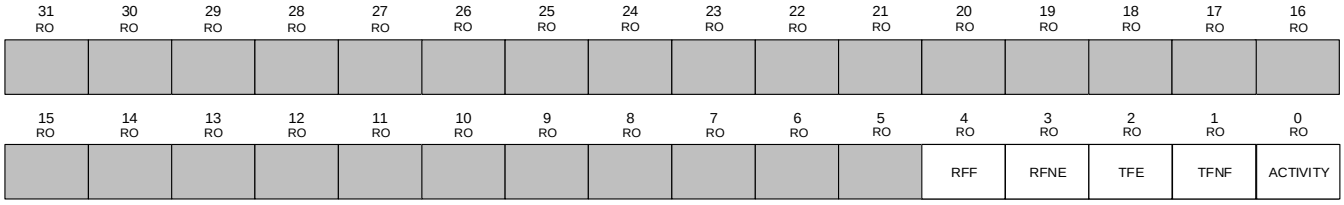


Offset: 0x006C

Bits	Name	Type	Reset	Description
31:1	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*ENABLE*	R/W	0	I2C Enable 0: I2C is disabled. When the I2C is disabled, the following occurs: The TX FIFO and RX FIFO get flushed. 1: I2C is enabled.

4.15.3.20 I2C_STATUS - I2C STATUS REGISTER

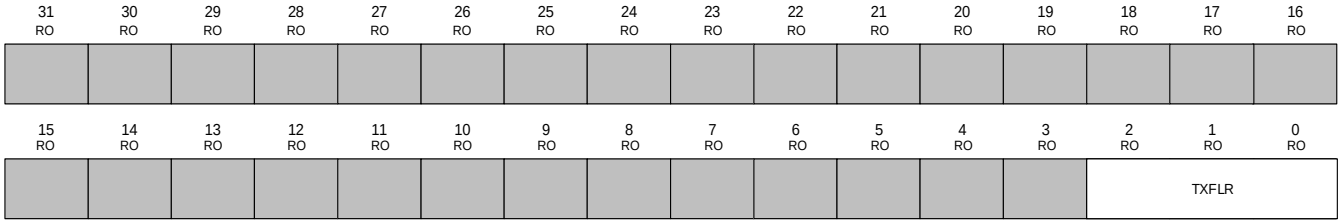
This is a read-only register used to indicate the current transfer status and FIFO status.



Offset: 0x0070

Bits	Name	Type	Reset	Description
31:5	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4	*RFF*	RO	0	Receive FIFO Completely Full 0: Receive FIFO is not full. 1: Receive FIFO is full.
3	*RFNE*	RO	0	Receive FIFO Not Empty 0: Receive FIFO is empty. 1: Receive FIFO is not empty.
2	*TFE*	RO	0	Transmit FIFO Completely Empty 0: Transmit FIFO is not empty. 1: Transmit FIFO is empty.
1	*TFNF*	RO	0	Transmit FIFO Not Full 0: Transmit FIFO is full. 1: Transmit FIFO is not full.
0	*ACTIVITY*	RO	0	I2C Activity Status 0: I2C is idle 1: I2C is activated.

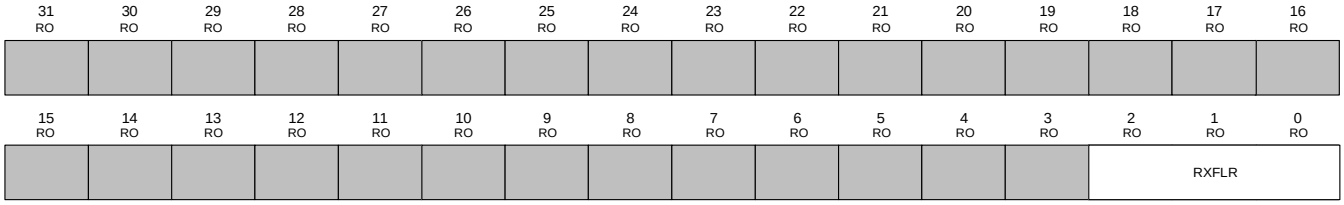
4.15.3.21 I2C_TXFLR - I2C TRANSMIT FIFO LEVEL REGISTER



Offset: 0x0074

Bits	Name	Type	Reset	Description
31:3	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*TXFLR*	RO	0x0	Transmit FIFO Level This field contains the number of valid data entries in the transmit FIFO.

4.15.3.22 I2C_RXFLR - I2C RECEIVE FIFO LEVEL REGISTER

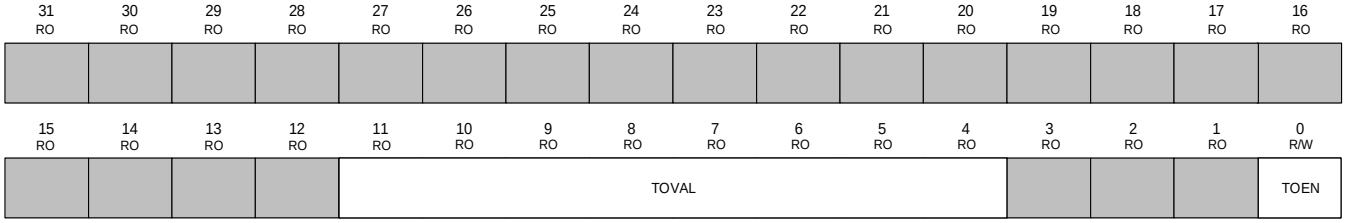


Offset: 0x0078

Bits	Name	Type	Reset	Description
31:3	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*RXFLR*	RO	0x0	Receive FIFO Level This field contains the number of valid data entries in the receive FIFO.

4.15.3.23 I2C_TIMEOUT - I2C TIMEOUT ENABLE REGISTER

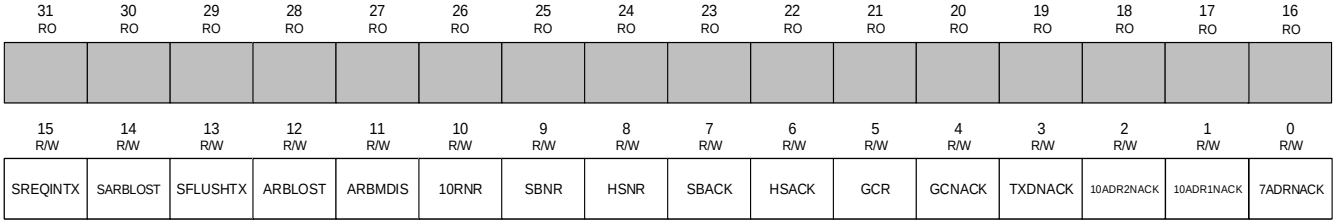
This register contains the timeout function enable and the timeout counter value.



Offset: 0x007C

Bits	Name	Type	Reset	Description
31:12	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11:4	*TOVAL*	RO	0x0	Timeout Counter Value
3:1	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*TOEN*	R/W	0	Timeout Enable This is only functional in master mode. When timeout is enabled, the master can wait for 256 period and reset itself, if the slave is not ready. 0: Timeout is disabled. 1: Timeout is enabled.

4.15.3.24 I2C_TX_ABRT - I2C TRANSMIT ABORT STATUS REGISTER



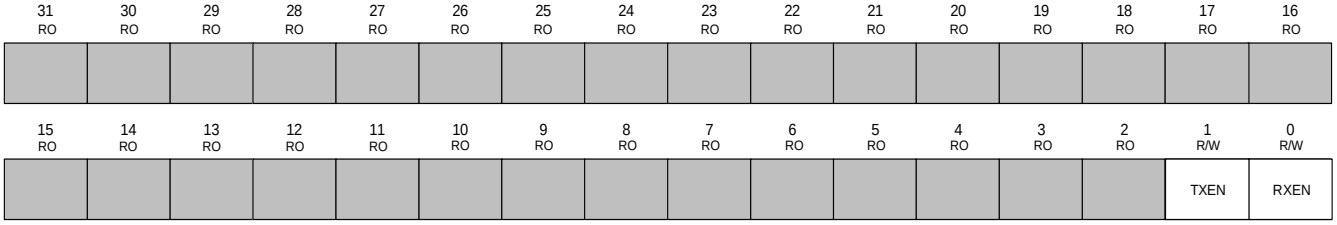
Offset: 0x0080

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15	*SREQINTX*	R/W	0	Slave Requesting Data to TX 1: Slave requesting data to Tx and the user wrote a read command into the Tx FIFO.
14	*SARBLOST*	R/W	0	Slave Arbitration Lost 1: Slave lost the bus while it is transmitting data to remote master/
13	*SFLUSHTX*	R/W	0	Slave Flush Tx 1: Slave has received a read command and some data exists in the Tx FIFO so the slave issues a Tx arbitration to flush old data in Tx FIFO.
12	*ARBLOST*	R/W	0	Arbitration Lost 1: Master has lost arbitration, or if SARLO is also set, then the slave transmitter has lost arbitration.
11	*ARBMDIS*	R/W	0	Arbitration Master Disable 1: User attempted to use disabled Master
10	*10RNR*	R/W	0	10-bit Addressing Read and No Restart 1: The restart is disabled and the Master sends a read command in 10-bit addressing.
9	*SBNR*	R/W	0	Start Byte and No Restart 1: The restart is disabled and the user is trying to send a Start Byte.
8	*HSNR*	R/W	0	High Speed and No Restart 1: The restart is disabled and the user is trying to use the master to send data in High Speed mode.
7	*SBACK*	R/W	0	Start Byte Acknowledge 1: Master has sent a Start Byte and the Start Byte was acknowledged (Wrong Behavior)
6	*HSACK*	R/W	0	High Speed Acknowledge 1: Master is in High Speed mode and the High Speed Master code was acknowledged. (Wrong Behavior)
5	*GCR*	R/W	0	General Call Read 1: Master sent a general call but the user programmed the byte following the general call to be read from the bus.
4	*GCNACK*	R/W	0	General Call Not Acknowledge 1: Master sent a general call and no slave on the bus responded with an acknowledgement.
3	*TXDNACK*	R/W	0	Tx Data Not Acknowledge 1: Master has received an acknowledgement for the address, but when it sent data byte(s) following the address, it did not receive an acknowledgement from the remote slave.

Bits	Name	Type	Reset	Description
2	*10ADR2NACK*	R/W	0	10-bit Addressing and 2nd address Not Acknowledge 1: Master is in 10-bit address mode and the 2nd address byte of the 10-bit address was not acknowledged by any slave.
1	*10ADR1NACK*	R/W	0	10-bit Addressing and 1st address Not Acknowledge 1: Master is in 10-bit address mode and the 1st 10-bit address byte was not acknowledged by any slave.
0	*7ADRNACK*	R/W	0	7-bit Addressing No Acknowledge 1: Master is in 7-bit addressing mode and the address sent was not acknowledged by any slave.

4.15.3.25 I2C_DMA_CTRL - I2C DMA CONTROL REGISTER

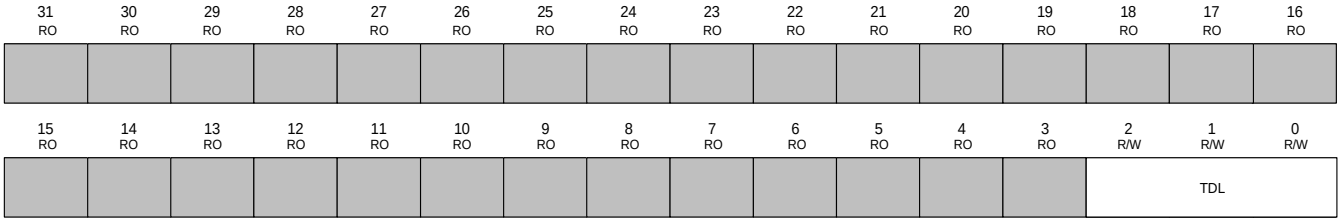
The register is used to enable the DMA Controller interface operation. There is a separate bit for transmitting and receiving.



Offset: 0x0088

Bits	Name	Type	Reset	Description
31:2	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*TXEN*	R/W	0	DMA Transmit Enable 0: Disable DMA transmit 1: Enable DMA transmit
0	*RXEN*	R/W	0	DMA Receive Enable 0: Disable DMA receive 1: Enable DMA receive

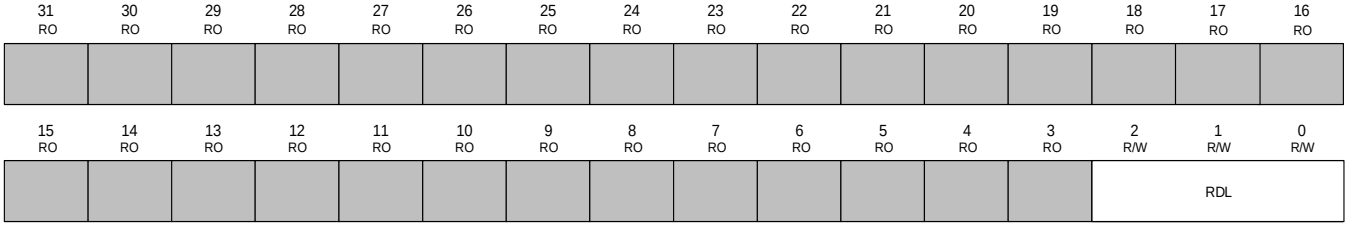
4.15.3.26 I2C_DMA_TDLR - I2C DMA TRANSMIT THRESHOLD REGISTER



Offset: 0x008C

Bits	Name	Type	Reset	Description
31:3	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*TDL*	R/W	0	DMA Transmit Data Level This bit field controls the level at which a DMA request is made by the transmit logic.

4.15.3.27 I2C_DMA_RDLR - I2C DMA RECEIVE THRESHOLD REGISTER



Offset: 0x0090

Bits	Name	Type	Reset	Description
31:3	*reserved*	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*RDL*	R/W	0	DMA Receive Data Level This bit field controls the level at which a DMA request is made by the receive logic.

4.16 SERIAL PERIPHERAL INTERFACE (SPI)

The Serial Peripheral Interface (SPI) provides an SPI protocol data transmit and receive functions in both master or slave mode. SPI supports full-duplex, half-duplex and simple synchronous, serial communication with external devices. On the principle of full duplex data transmission, 4bit to 16bit data sent by the master to the slave or the slave sent to the master.

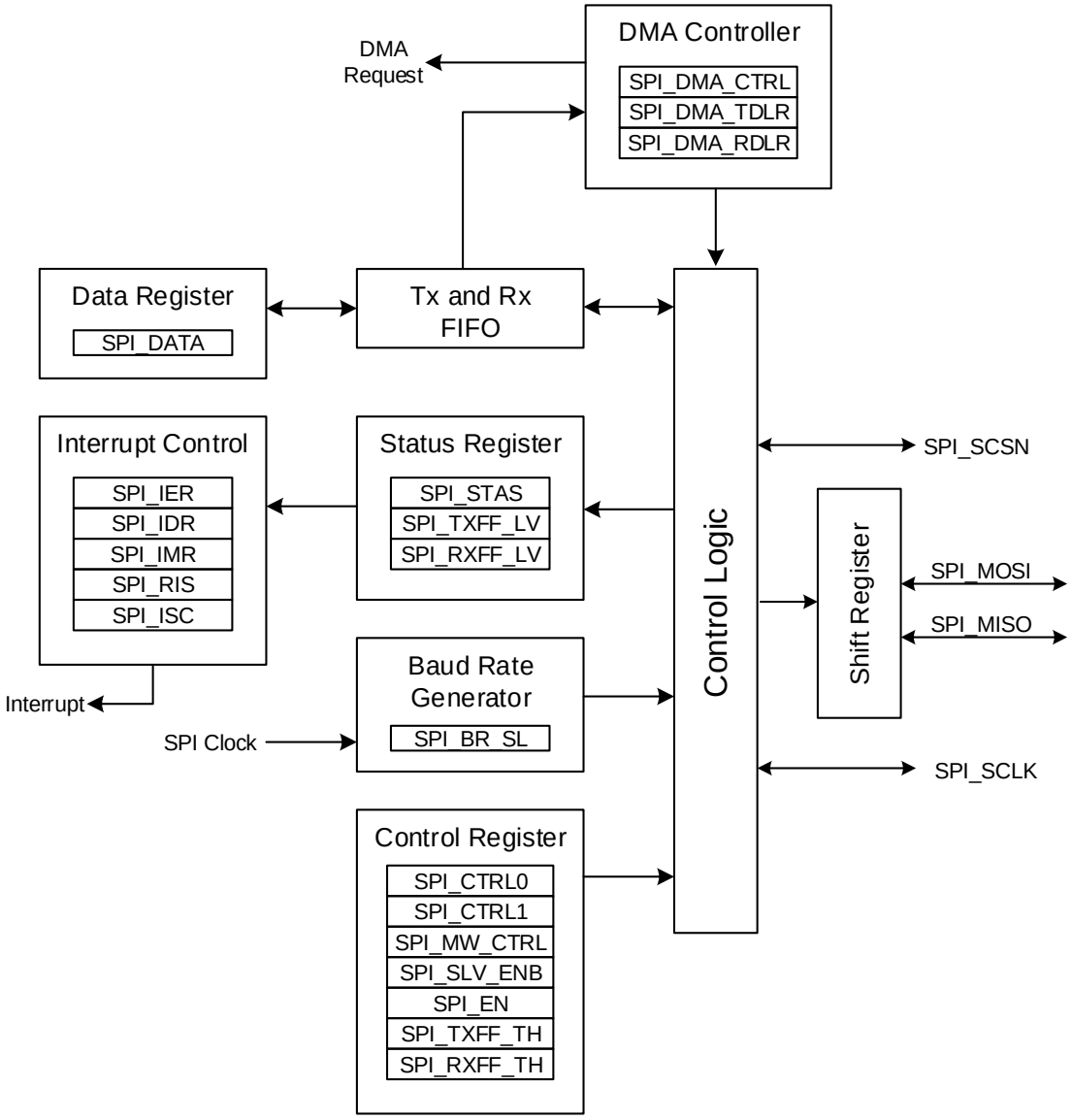
Note: In PT32C302, as I2S can be configured as SPI operating mode, so there are up to three set of SPI modules depends on user's setting in I2S. Refer to I2S Block Diagram*to understand more about I2S' SPI configuration.*

SPI Main Features

- Master / Slave operation
- Serial-master or serial-slave operation - Enables serial communication with serial-master or serial-slave peripheral devices.
- Serial Interface operation - Programmable choice of the following protocols:
  - Motorola SPI protocol
  - Texas Instruments Synchronous Serial Protocol
  - National Semiconductor Mircowire
- DMA handshaking interface
- Synchronous serial communication.
- Single data direction operation allows alternate function on MISO or MOSI pin
- 4 - 16 bits per frame.

4.16.1 BLOCK DIAGRAM

Figure 4.16-1: SPI Block Diagram



- SPIx_MOSI: Master In/Slave Out data. In the general case, this pin is used to transmit data in slave mode and receive data in master mode.
- SPIx_MISO: Master Out/Slave In data. In the general case, this pin is used to transmit data in master mode and receive data in slave mode.
- SPIx_SCLK: Serial bit rate clock output pin for SPI masters and input pin for SPI slaves.
- SPIx_SCSN: Slave select pin. Depending on the SPI and SCSN setting, this pin can be used to either:
  - Select an individual slave device for communication
  - Synchronize the data frame
  - Detect a conflict between multiple masters

4.16.2 SPI FUNCTION DESCRIPTION

4.16.2.1 SERIAL BIT-RATE CLOCKS

The maximum frequency of the SPI master bit-rate clock (SPI_SCLK) is a quarter of the frequency of the SPI master clock (SPI clock).

The frequency of the SPI_SCLK is defined by:

F SPI_SCLK = FSPI Clock / SCKDV

SCKDV is a bit field in the register I2C_BR_SL, holding any even value in the range 4 to 65534. If SCKDV is equal or smaller than 3, then SPI_SCLK is disabled.

4.16.2.2 TRANSFER MODES

When transferring data on the serial bus, the SPI controller operates one of several modes. The transfer mode (TMOD) is set by writing to the SPI_CTRL0.TMOD field in control register 0 SPI_CTRL0.

Note: The transfer mode setting does not affect the duplex of the serial transfer. SPI_CTRL0.TMOD is ignored for Microwire transfers, which are controlled by the SPI_MW_CTRL register.

Note: In Master mode, SPI_MOSI line is denoted as txd line; SPI_MISO line is denoted as rxd line. In Slave mode, SPI_MOSI line is denoted as rxd line; SPI_MISO line is denoted as txd line.

TRANSMIT AND RECEIVE

When **SPI_CTRL0.***TMOD* = 0, both transmit and receive logic are valued. The data transfer occurs as normal according to the selected format (serial protocol). Transmit data are popped from the txd to the target device, which replies with data on the rxd line. The receive data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame.

TRANSMIT ONLY

When **SPI_CTRL0.***TMOD* = 1, the receive data are invalid and should not be stored in the receive FIFO. The data transfer occurs as normal, according to the selected frame format (serial protocol). Transmit data are popped from the transmit FIFO and sent through the txd line to the target device, which replies with data on the rxd line. At the end of the data frame, the receive shift register does not load its newly received data into the receive FIFO. The data in the receive shift register is overwritten by the next transfer. Any interrupts originating from the receive logic should be masked when this mode is entered.

RECEIVE ONLY

When **SPI_CTRL0.***TMOD* = 2, the transmit data are invalid. When configured as a slave, the transmit FIFO is never popped in Receive Only mode. The txd output remains at a constant logic level during the transmission. The data transfer occurs as normal according to the selected frame format (serial protocol). The receive data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame. Any interrupts originating from the transmit logic should be masked when this mode is entered.

EEPROM READ

Note: This transfer mode is only valid for serial masters.

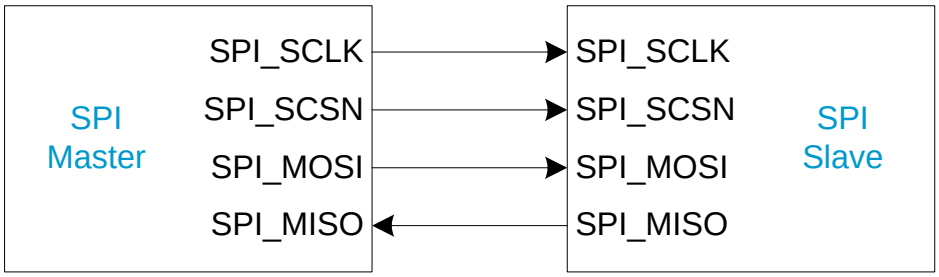
When **SPI_CTRL0.***TMOD* = 3, the transmit data is used to transmit an opcode and/or an address to the EEPROM device. Typically this takes three data frames (8-bit opcode followed by 8-bit upper address and 8-bit lower address). During the transmission of the opcode and address, no data is captured by the receive logic (as long as the SPI master is transmitting data on its txd line, data on the rxd line is ignored). The SPI master continues to transmit data until the transmit FIFO is empty. Therefore, you should ONLY have enough data frames in the transmit FIFO to supply the opcode and address to the EEPROM. If more data frames are in the transmit FIFO than are needed, then read data is lost. When the transmit FIFO becomes empty (all control information has been sent), data on the rxd is valid and is stored in the receive FIFO. The serial transfer continues until the number of data frames received by the SPI master matches the value of **SPI_CTRL1.***NDF* field plus one.

4.16.2.3 SPI MASTER AND SLAVE MODE

The SPI can be configured in the following two fundamental modes of operation:

- SPI Master mode
- SPI Slave mode

Figure 4.16-2:: The SPI Master connect to the Slave



SPI MASTER

The SPI master initiated and controls all serial transfers with serial-slave peripheral devices.

The serial bit-rate clock, generated and controlled by the SPI, is driven out on the SPI_SCLK line. When the SPI is disabled, no serial transfers can occur and SPI_SCLK is held in “inactive” state.

The SPI_SCSN pin is active during the full data transmission. The data stream will transmit or receive data in the shift register to the SPI_MOSI and SPI_MISO pin on the serial clock edge.

DATA TRANSFERS

The SPI master starts data transfers when all the following conditions are met:

- The SPI master operation is enabled
- There is at least one valid entry in the transmit FIFO buffer
- A slave device is selected

When actively transferring data, the busy flag (*BUSY*) in **SPI_STAS** register is set. You must wait until this flag is cleared before attempting a new serial transfer.

Note: The BUSY flag is not set when the data are written into the transmit FIFO buffer. This bit gets set only when the target slave has been selected and the transfer is underway. After writing data into the transmit FIFO buffer, the shift logic does not begin the serial transfer until a positive edge of the SPI_SCLK out signal is present. The delay in waiting for this positive edge depends on the baud rate of the serial transfer.

MASTER SPI AND SSP SERIAL TRANSFERS

For detailed description on SPI and SSP protocols, refer to Motorola Serial Peripheral Interface (SPI) and Texas Instruments Synchronous Serial Protocol (SSP) section.

When the transfer mode is “Transmit and Receive” (**SPI_CTRL0.***TMOD* = 0) or “Transmit only” (**SPI_CTRL0.***TMOD* = 1), transfer are terminated by the shift control logic when the transmit FIFO buffer is empty. For continuous data transfers, you must ensure that the transmit FIFO buffer does not become empty before all the data have been transmitted. The transmit FIFO threshold level register - **SPI_TXFF_TH** can be used to early interrupt (Transmit FIFO Empty Interrupt - **SPI_RIS**.*TXERI*) the processor indicating that the transmit FIFO buffer is nearly empty.

When the DMA is used in conjunction with the SPI master, the transmit data level (**SPI_DMA_TDLR**) can be used to early request the DMA Controller, indicating that the transmit FIFO buffer is nearly empty. The FIFO buffer can then be refilled with data to continue the serial transfer.

When the transfer mode is “Receive only” (**SPI_CTRL0.***TMOD*=2), a serial transfer is started by writing one “dummy” data into the transmit FIFO buffer when a serial slave is selected. If the serial transfer is continuous, this same data word is retransmitted until the serial transfer is completed. The transmit FIFO is popped only once at the beginning and may remain empty for the duration of the serial transfer. The end of the serial transfer is controlled by the “number of data frames” (*NDF*) field in control register 1 (**I2C_CTRL1**).

When the transfer mode is “EEPROM read” (**SPI_CTRL0.***TMOD*=3), a serial transfer is started by writing the opcode and/or address into the transmit FIFO buffer when a serial slave (EEPROM) is selected. The opcode and address are transmitted to the EEPROM device, after which read data is received from the EEPROM device and stored in the receive FIFO buffer. The end of the serial transfer is controlled by the *NDF* field in the control register 1 (**I2C_CTRL1**).

MASTER MICROWIRE SERIAL TRANSFERS

For detailed description on Microwire protocols, refer to National Semiconductor Microwire section.

Microwire serial transfers from the SPI serial master are controlled by the Microwire Control Register (**SPI_MW_CTRL**). The **SPI_MW_CTRL**.*MHS* bit field enables and disables the Microwire handshaking interface. The **SPI_MW_CTRL**.*MDD* bit field controls the direction of the data frame (the control frame is always transmitted by the master and received by the slave). The **SPI_MW_CTRL**.*MWMOD* bit field defines whether the transfer is sequential or nonsequential.

All Microwire transfers are started by the SPI serial master when there is at least one control word in the transmit FIFO buffer and a slave is enabled. When the SPI master transmits the data frame (**SPI_MW_CTRL**.*MDD* = 1), the transfer is terminated by the shift logic when the transmit FIFO buffer is empty. When the SPI master receives the data frame (**SPI_MW_CTRL**.*MDD* = 0), the termination of the transfer depends on the setting of the *MWMOD* bit field. If the transfer is nonsequential (**SPI_MW_CTRL**.*MWMOD* = 0), it is terminated when the transmit FIFO buffer is empty after shifting in the data frame from the slave. When the transfer is sequential (**SPI_MW_CTRL**.*MWMOD* = 1), it is terminated by the shift logic when the number of data frames (*NDF*) received is equal to the value in the **SPI_CTRL1** register plus one.

When the handshaking interface on the SPI master is enabled (**SPI_MW_CTRL**.*MHS* = 1), the status of the target slave is polled after transmission. Only when the slave reports a ready status does the SPI master complete the transfer and clear its *BUSY* status. If the transfer is continuous, the next control/data frame is not sent until the slave device returns a ready status

SPI SLAVE

The SPI slave handles serial communication with transfer initiated and controlled by serial master peripheral devices.

The SPI_SCLK pin acts as an input pin and the serial clock will be derived from the external master device. The SPI_SCSN pin also acts as an input. The data stream will transmit or receive data in the shift register to the SPI_MOSI and SPI_MISO pin on the serial clock edge.

SLAVE SPI AND SSP SERIAL TRANSFERS

For detailed description on SPI and SSP protocols, refer to Motorola Serial Peripheral Interface (SPI) and Texas Instruments Synchronous Serial Protocol (SSP) section.

It the SPI slave transmits data to the master, when the transfer mode is “Transmit and Receive” (**SPI_CTRL0.***TMOD* = 0) or “Transmit only” (**SPI_CTRL0.***TMOD* = 1), you must ensure that data exists in the transmit FIFO before a transfer is initiated by the serial-master device. If the master transfer to the SPI slave when no data exists in the transmit FIFO, an error flag (**SPI_STAS**.*TXE*) is set, and the previous transmitted data frame is resent on SPI_MISO line. For continuous data transfers, you must ensure that the transmit FIFO buffer does not become empty before all the data have been transmitted. The transmit FIFO threshold level register - **SPI_TXFF_TH** can be used to early interrupt (Transmit FIFO Empty Interrupt - **SPI_RIS**.*TXERI*) the processor indicating that the transmit FIFO buffer is nearly empty.

When the DMA is used in conjunction with the SPI master, the transmit data level (**SPI_DMA_TDLR**) can be used to early request the DMA Controller, indicating that the transmit FIFO buffer is nearly empty. The FIFO buffer can then be refilled with data to continue the serial transfer.

If the SPI slave is receive only (**SPI_CTRL0.***TMOD* = 2), the transmit FIFO need not contain valid data because the data currently in the transmit shift register is resent each time the slave device is selected.

SLAVE MICROWIRE SERIAL TRANSFERS

For detailed description on SPI and SSP protocols, refer to National Semiconductor Microwire section.

When the SPI is configured as a slave device, the Microwire protocol operates in much the same way as the SPI protocol. There is no decode of the control frame by the SPI slave device

4.16.2.4 PARTNER CONNECTION INTERFACES

In order for the SPI connect to a serial-master or serial-slave peripheral device, the peripheral must have a least one of the following interfaces:

- Motorola SPI protocol
- Texas Instruments Synchronous Serial Protocol
- National Semiconductor Mircowire

The serial protocols supported by the SPI controller allow for serial slaves to be selected or addressed using hardware. Serial slaves are selected under the control of dedicated hardware select lines. The number of select lines generated from the serial master is equal to the number of serial slaves present on the bus. The serial-master device asserts the select line of the target serial slave before data transfer begins.

SPI SIGNAL DESCRIPTION

This table describes the alternate function of SPI pins under different interfaces.

Table 4.16-1: SPI/I2S Pin Description in Different interface

Symbol	Interface Name / Function		Description
	SSP	MicroWire	
SCLK	CLK	SK	Serial clock. SCLK is a clock signal used to synchronize data transmission. When using the SPI interface, the clock programming can be active high or active low, otherwise, it is active high. SCLK only during data transmission hopping.
SCSN	FS	CS	Frame synchronization / Slave selection. When SPI is the master, it will be driven to active state before starting transmission, and release the signal into the inactive state after send the information. This signal is active high or active low is depending on the selected bus mode. When the SPI interface is the slave, the signal is sent from the master. When there is only one bus master and a bus slave, frame sync from the master or from a selection signal can be directly connected to the slave. When there is more than one slave, it is necessary to further limit its frame selection / slave select signal, in order to avoid multiple slaves to respond to the transmission on the bus.
MISO	DR(M) DX(S)	SI (M) SO (S)	Master Input Slave Output. MISO serial data will be transmitted from the slave to the master. When SPI is a slave, serial data output from the signal. When SPI is the master, it records the serial data sent from the signal. When SPI is not slave and FS / SSEL not be selected, it does not drive the signal (it is in a high impedance state)
MOSI	DX(M) DR(S)	SO(M) SI(S)	Master Out Slave input. MOSI serial data signals transmitted from master to slave. When SPI is the master, serial data output from the pin. When the SPI is the slave, this pin receives input data from the master.

MOTOROLA SERIAL PERIPHERAL INTERFACE (SPI)

A four-wire, full-duplex serial protocol. Main characteristics of SPI format is that SCLK signal polarity and phase can be controlled by **SPI_CTRL0** register's bits *SCPOL* and *SCPH*.

Clock Polarity (SCPOL) and Phase (SCPH) Control

SCPOL control the clock polarity. When the device is in idle state, if the *SCPOL* is low, it will produce a stable low-level at the SCLK pin; if the SCPOL is high, it will produce a stable high-level at the SCLK pin.

SCPH control bits select the capture data and allows data to change the status of the clock edge. When *SCPH* is low, the data will be captured by the first clock edge transition. If *SCPH* is high, the data will be captured by the second clock edge transition.

Case 1: SCPOL = 0 and SCPH = 0

When *SCPOL* = 0 and *SCPH* = 0:

Figure 4.16-3: SPI Serial Format in Case 1

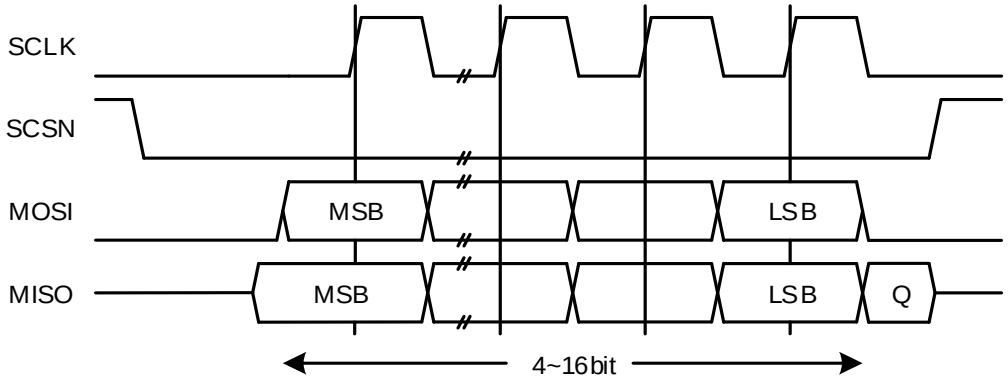
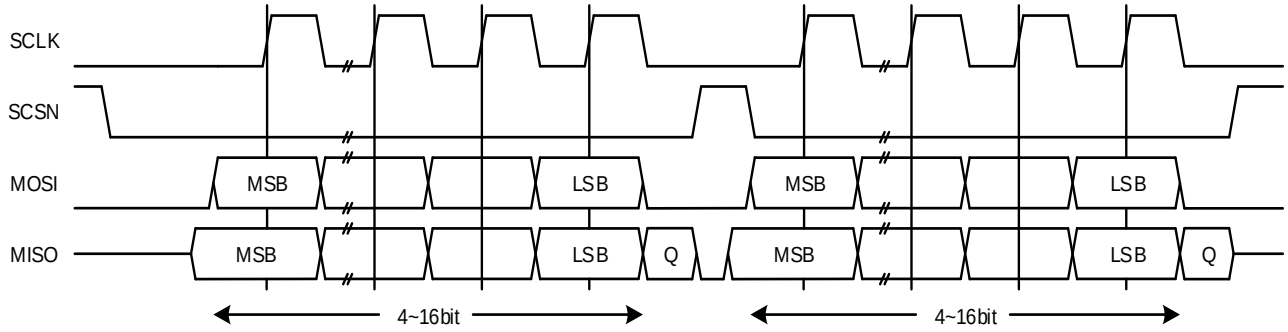


Figure 4.16-4: SPI Serial Format Continuous Transfer in Case 1



In this configuration, during idle periods:

- CLK signal is forced low.
- SSEL signal is forced high.
- MOSI/MISO pin is in high impedance state.

When the configuration parameter *SCPH* = 0, data transmission begins on the falling edge of the slave select signal. The first data bit is captured by the master and slave peripherals on the first edge of the serial clock; therefore, valid data must be present on the txd and rxd lines prior to the first serial clock edge. As data transmission starts on the falling edge of the slave select signal when *SCPH* = 0, continuous data transfers require the slave select signal to toggle before beginning the next data frame.

Case 2: SCPOL= 0 and SCPH = 1

When *SCPOL* = 0 and *SCPH* = 1:

Figure 4.16-5: SPI Serial Format in Case 2

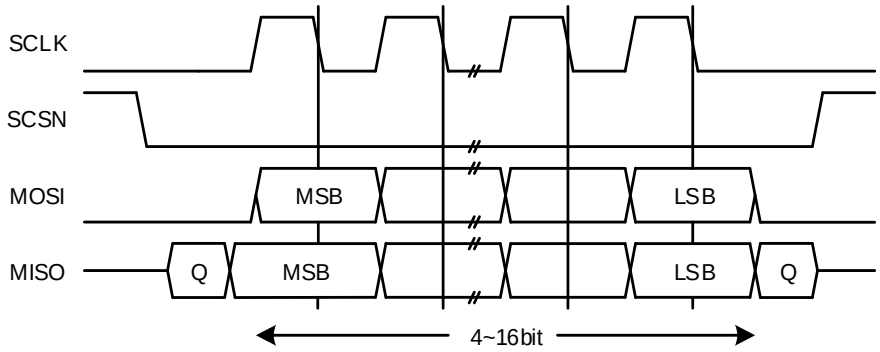
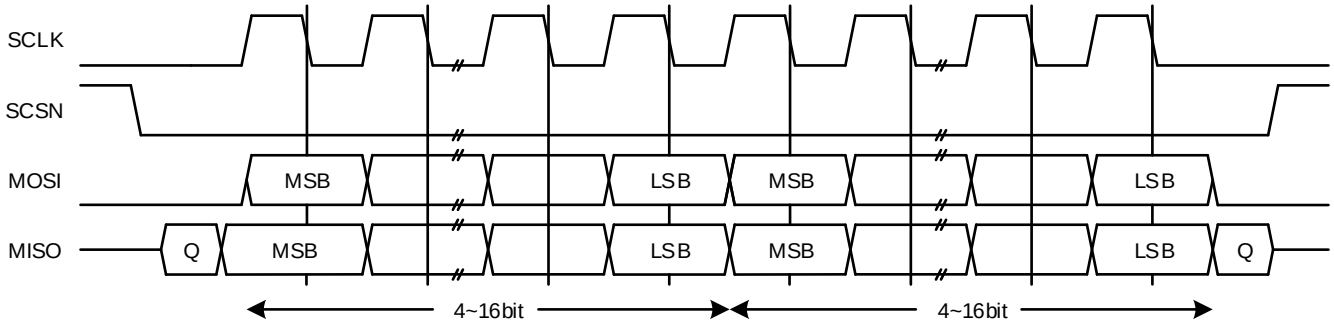


Figure 4.16-6: SPI Serial Format Continuous Transfer in Case 2



In this configuration, during idle periods:

- CLK signal is forced low.
- SSEL signal is forced high.
- MOSI/MISO pin is in high impedance state.

When the configuration parameter *SCPH* = 1, both master and slave peripherals begin transmitting data on the first serial clock edge after the slave select line is activated. The first data bit is captured on the second (trailing) serial clock edge. Data are propagated by the master and slave peripherals on the leading edge of the serial clock. During continuous data frame transfers, the slave select line may beheld active-low until the last bit of the last frame has been captured. Continuous data frames are transferred in the same way as single frames, with the MSB of the next frame following directly after the LSB of the current frame. The slave select signal is held active for the duration of the transfer.

Case 3 : SCPOL = 1 and SCPH = 0

When *SCPOL* = 1 and *SCPH* = 0:

Figure 4.16-7: SPI Serial Format in Case 3

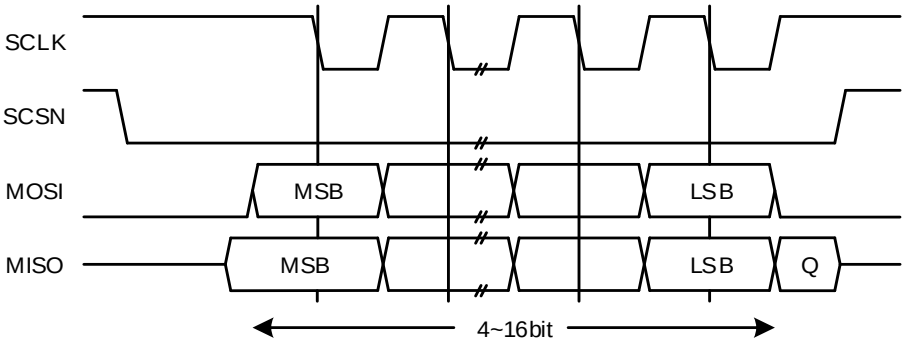
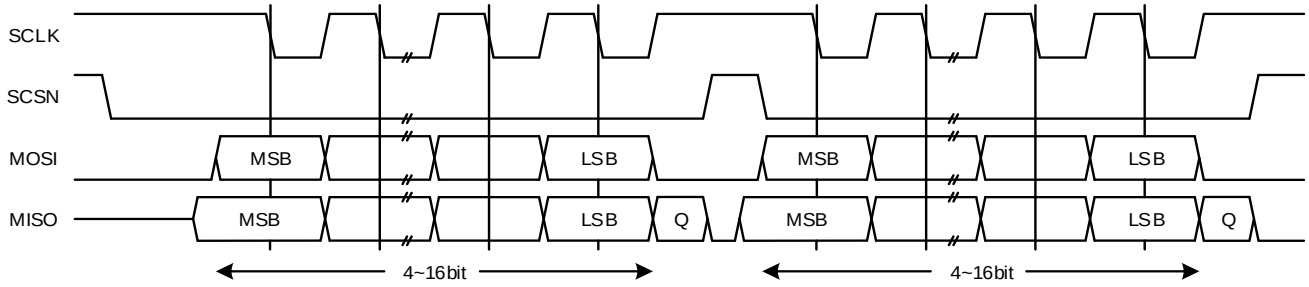


Figure 4.16-8: SPI Serial Format Continuous in Case 3



In this configuration, during idle periods:

- CLK signal is forced high.
- SSEL signal is forced high.
- MOSI/MISO pin is in high impedance state.

When the configuration parameter *SCPH* = 0, data transmission begins on the falling edge of the slave select signal. The first data bit is captured by the master and slave peripherals on the first edge of the serial clock; therefore, valid data must be present on the txd and rxd lines prior to the first serial clock edge. As data transmission starts on the falling edge of the slave select signal when *SCPH* = 0, continuous data transfers require the slave select signal to toggle before beginning the next data frame.

Case 4 : SCPOL = 1 and SCPH = 1

When *SCPOL* = 1 and *SCPH* = 1:

Figure 4.16-9: SPI Serial Format in Case 4

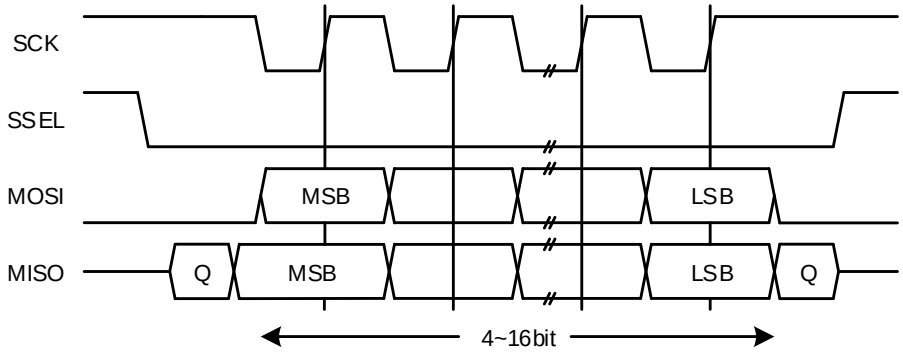
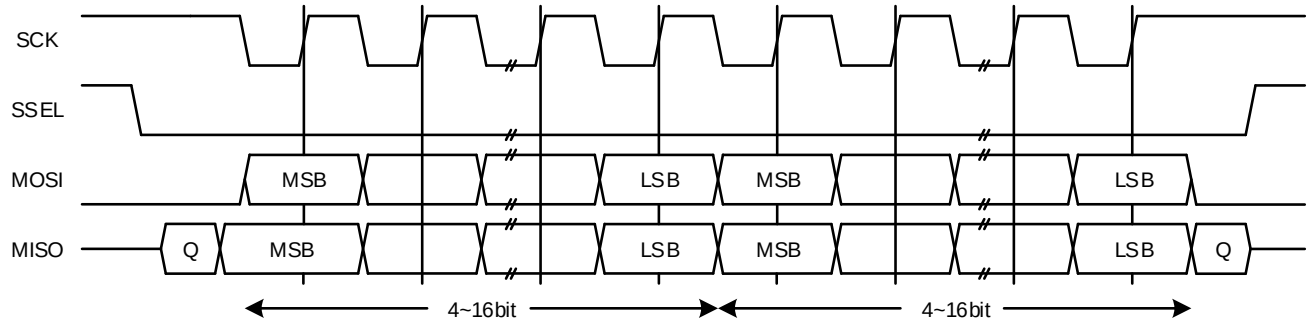


Figure 4.16-10: SPI Serial Format Continuous Transfer in Case 4



In this configuration, during idle periods:

- CLK signal is forced high.
- SSEL signal is forced high.
- MOSI/MISO pin is in high impedance state.

When the configuration parameter *SCPH* = 1, both master and slave peripherals begin transmitting data on the first serial clock edge after the slave select line is activated. The first data bit is captured on the second (trailing) serial clock edge. Data are propagated by the master and slave peripherals on the leading edge of the serial clock. During continuous data frame transfers, the slave select line may beheld active-low until the last bit of the last frame has been captured. Continuous data frames are transferred in the same way as single frames, with the MSB of the next frame following directly after the LSB of the current frame. The slave select signal is held active for the duration of the transfer.

TEXAS INSTRUMENTS SYNCHRONOUS SERIAL PROTOCOL (SSP)

Texas Instruments Serial Protocol (SSP) is a four-wire, full duplex serial protocol.

Figure 4.16-11: SSP Serial Format

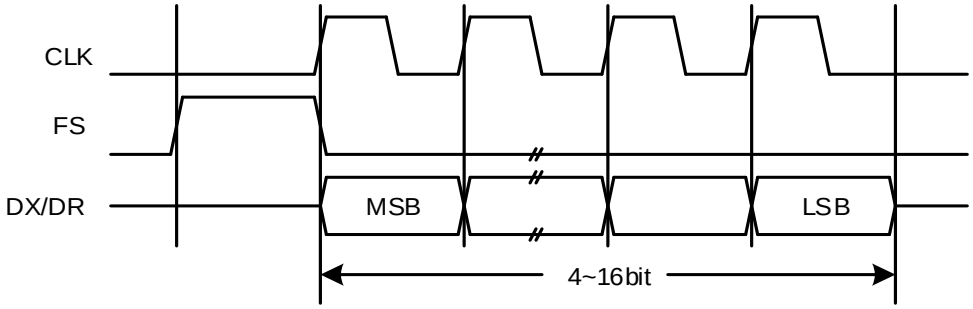
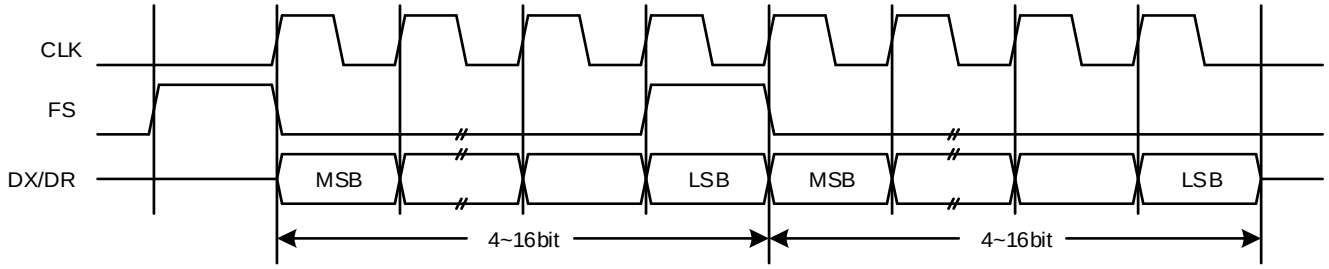


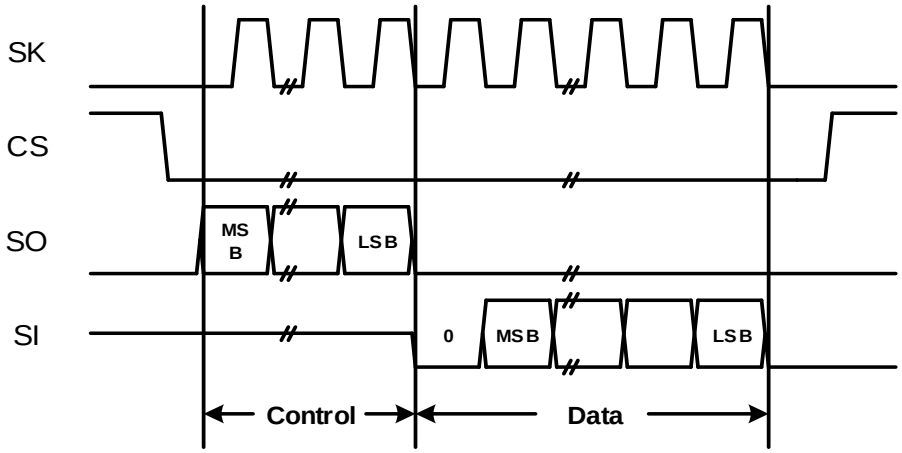
Figure 4.16-12: SSP Serial Format Continuous Transfer



Data transfers begin by asserting the frame indicator line (FS) for one serial clock period. Data to be transmitted are driven onto the DX line one serial clock cycle later; similarly data from the slave are driven onto the DR line. Data are propagated on the rising edge of the serial clock and captured on the falling edge. The length of the data frame ranges from 4 to 16bits.Continuous data frames are transferred in the same way as single data frames. The frame indicator is asserted for one clock period during the same cycle as the LSB from the current transfer, indicating that another data frame follows.

NATIONAL SEMICONDUCTOR MICROWIRE

Figure 4.16-13: Microwire Serial Format



When the SPI is configured as a serial master, data transmission begins with the falling edge of the slave-select signal (CS). One-half serial clock (SK) period later, the first bit of the control is sent out on the SO line. The length of the control word can be in the range 1 to 16 bits and is set by writing bit field *CFS* (bits 15:12) in **SPI_CTRL0**. The remainder of the control word is transmitted (propagated on the falling edge of sclk_out) by the SPI serial master. During this transmission, no data are present (high impedance) on the serial master's SI line.

The direction of the data word is controlled by the *MDD* bit field (bit 1) in the Microwire Control Register (**SPI_MW_CTRL**). When *MDD*=0, this indicates that the SPI serial master receives data from the external serial slave. One clock cycle after the LSB of the control word is transmitted, the slave peripheral responds with a dummy 0 bit, followed by the data frame, which can be 4 to 16 bits in length. Data are propagated on the falling edge of the serial clock and captured on the rising edge.

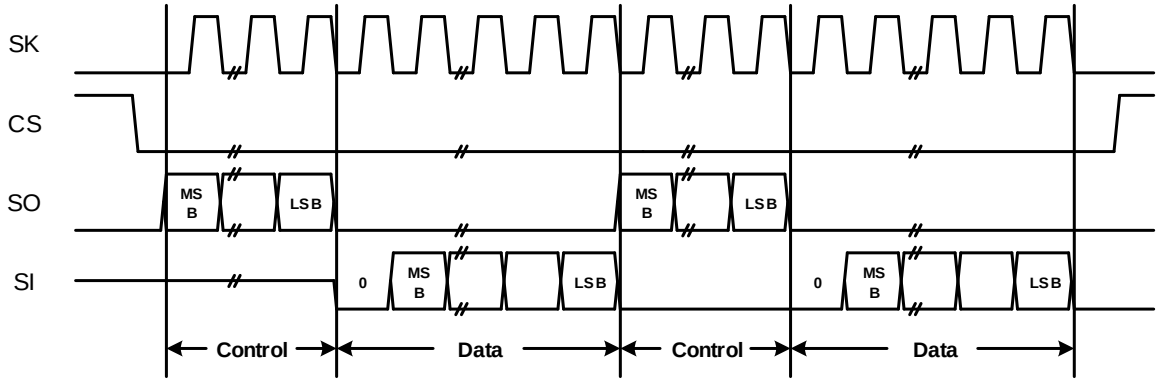
The slave-select signal is held active-low during the transfer and is de-asserted one-half clock cycle later, after the data are transferred.

In the Microwire mode, you can set the control register to change transmission format, whether to open the transfer mode (*MDD*), whether the continuous transmission of data transmission (*MWMOD*), and whether to open handshake function (*MHS*) and so on. There are several configurations:

Case 1 : MHS = 0, MDD = 0 and MWMOD = 0

When handshake function (*MHS*) is disable, transfer Mode Select (*MDD*) receive and continuous transmission of data (*MWMOD*) is disable, the Microwire format show as follow figure:

Figure 4.16-14: Microwire Serial Format in Case 1

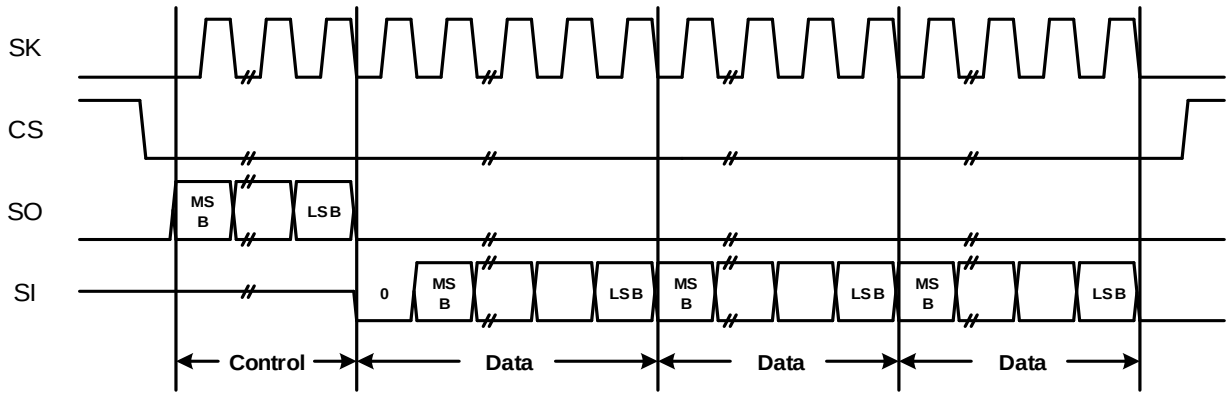


In this case of continuous transmission of this configuration, the transfer is the same with the single format. After the current data LSB byte is received, the control byte of the next frame will be sent immediately. After a frame LSB byte is latched into the SPI shift register, each received data is transmitted to the receiver FIFO on the falling edge of SK.

Case 2: MHS = 0, MDD = 0 and MWMOD = 1

When handshake function (*MHS*) is disable, transfer Mode Select (*MDD*) receive and continuous transmission of data (*MWMOD*) is enable, the Microwire format show as follow figure:

Figure 4.16-15: Microwire Serial Format in Case 2

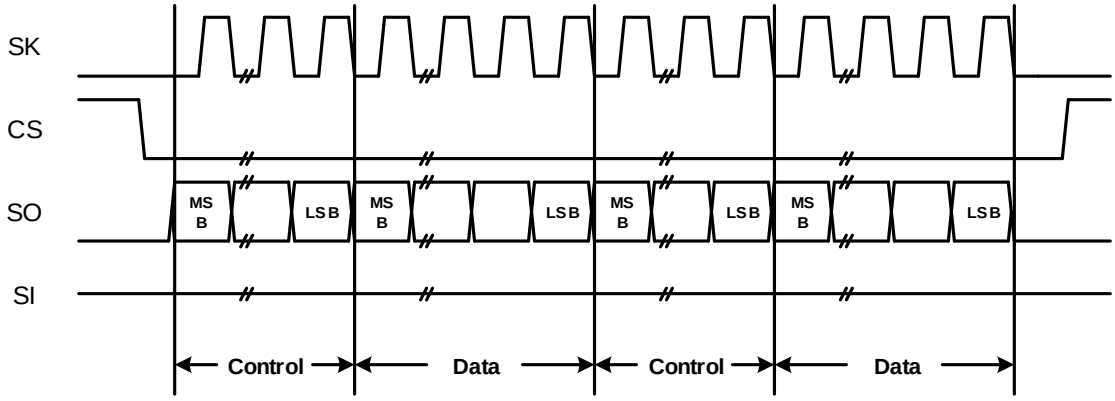


In this case of continuous transmission of this configuration, the transfer is the same with the single format. After the current data LSB byte is received, the control byte of the next frame will be sent immediately. The number of receive data bytes can setting by **SPI_CTRL1** register. After a frame LSB byte is latched into the SPI shift register, each received data is transmitted to the receiver FIFO on the falling edge of SK.

Case 3: MHS = 0, MDD = 1 and MWMOD = 0

When handshake function (*MHS*) is disable, transfer Mode Select (*MDD*) transmit and continuous transmission of data (*MWMOD*) is disable, the Microwire format show as follow figure:

Figure 4.16-16: Microwire Serial Format in Case 3

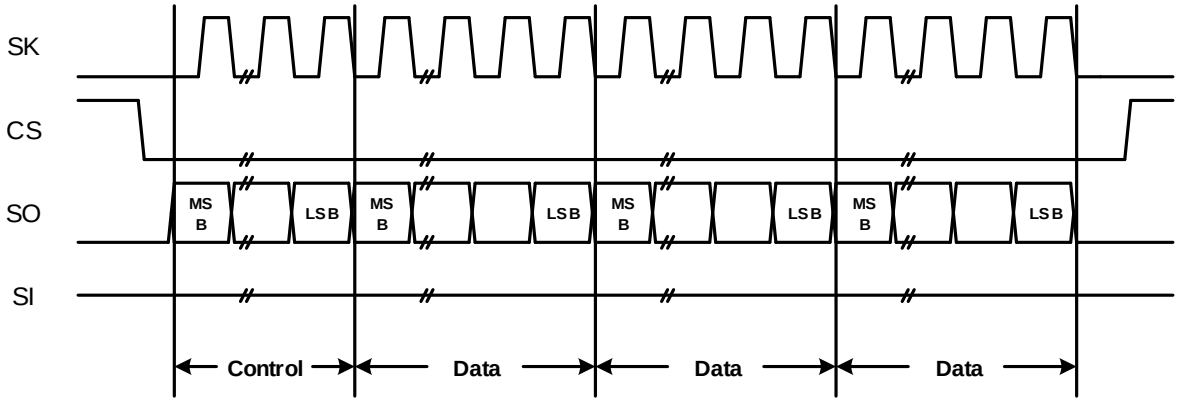


In this case of continuous transmission of this configuration, after the LSB current source frame has been transmitted, the next control byte MSB would then transmitted. At this condition is not open receive.

Case 4: MHS = 0, MDD = 1 and MWMOD = 1

When handshake function (*MHS*) is disable, transfer Mode Select (*MDD*) transmit and continuous transmission of data (*MWMOD*) is enable, the Microwire format show as follow figure:

Figure 4.16-17: Microwire Serial Format in Case 4



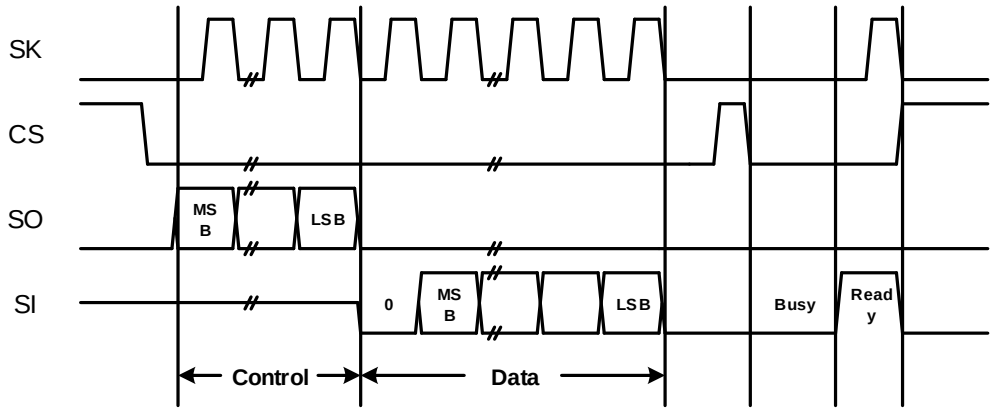
In this case of continuous transmission of this configuration, after the LSB current source frame has been transmitted, the next control byte MSB would then transmitted. At this condition is not open receive.

Case 5 & 6: MHS = 1, MDD = 0 and MWMOD = x

When handshake function (*MHS*) is enable, transfer Mode Select (*MDD*) receive and continuous transmission of data (*MWMOD*) is enable / disable, the Microwire format show as follow figure:

- Only valid in Master Mode.

Figure 4.16-18: Microwire Serial Format in Case 5&6



In this case of continuous transmission of this configuration, the transfer is the same with the single format. After the current data byte LSB is received, it toggle the CS, and began waiting outside the machine to respond to Ready. After receiving the response, end of the transfer.

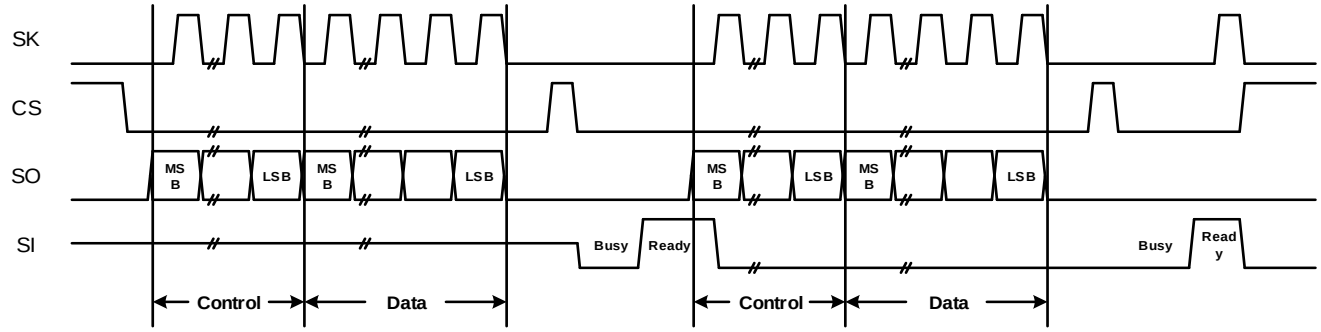
Note: If set *MWMOD* = 1, the handshake signal issuing after receive data transmission completely (in the **SPI_CTRL1** set the number of data).

Case 7: MHS = 1, MDD = 1 and MWMOD = 0

When handshake function (*MHS*) is enable, transfer Mode Select (*MDD*) transmit and continuous transmission of data (*MWMOD*) is disable, the Microwire format show as follow figure:

- Only valid in Master Mode.

Figure 4.16-19: Microwire Serial Format in Case 7



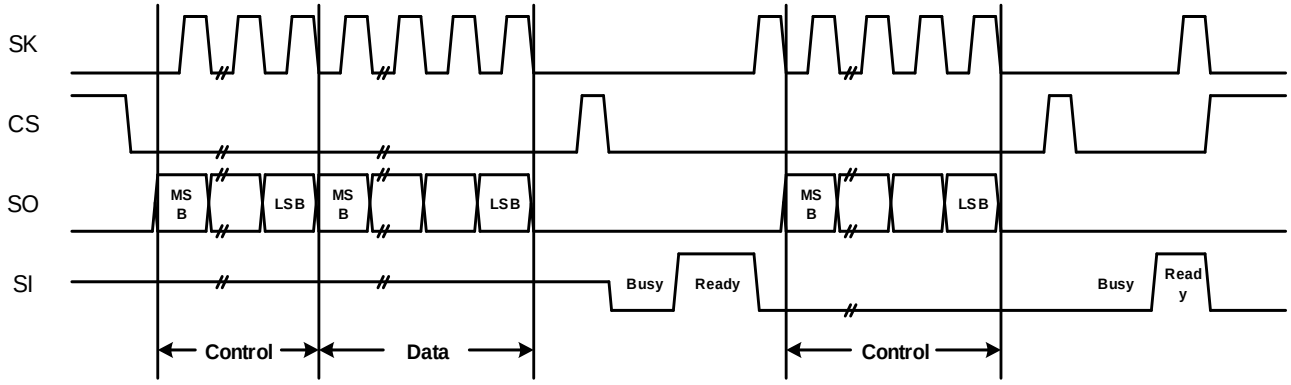
In this case of continuous transmission of this configuration, after the current data byte LSB is received, it toggle the CS, and began waiting outside the machine to respond to Ready. After receiving the response, will continue to send the control and data byte until all the data are be send. At this time, end of the transmission.

Case 8: MHS = 1, MDD = 1 and MWMOD = 1

When handshake function (MHS) is enable, transfer Mode Select (MDD) transmit and continuous transmission of data (MWMOD) is enable, the Microwire format show as follow figure:

- Only valid in Master Mode.

Figure 4.16-20: Microwire Serial Format in Case 8



In this case of continuous transmission of this configuration, after the current data byte LSB is received, it toggle the CS, and began waiting outside the machine to respond to Ready. After receiving the response, it will continue to send the next data's MSB, each time sending a data handshake signals are there until all the data are sent and end of the transmission.

4.16.2.5 DMA CONTROLLER

SPI in PT32C302 supports DMA signaling to indicate when data is ready to be read or when the transmit FIFO needs data. System software can choose the DMA burst mode by programming an appropriate value into the threshold registers.

To enable the DMA controller interface on the I2C controller, you must write to the DMA control register (**SPI_DMA_CTRL**) bits. Writing a 1 into the *TXEN* bit field of **SPI_DMA_CTRL** register enables the SPI controller transmit handshaking interface. Writing a 1 into the *RXEN* bit field of the **SPI _DMA_CTRL** register enables the SPI controller receive handshaking interface.

4.16.3 SPI PROGRAMMING MODEL

This section describes the programming model for the SPI

- Master SPI and SSP serial transfers
- Master Microwire serial transfers
- Slave SPI and SSP serial transfers
- Slave Microwire serial transfers

4.16.3.1 MASTER SPI AND SSP SERIAL TRANSFERS

To use SPI as a master to perform SPI or SSP serial transfers, perform the following steps:

1. Disabled the SPI by writing a 0 to SPIEN field of the SPI_ENABLE register.
2. Set up the SPI control register - SPI_CTRL0 and SPI_CTRL1 for the transfer. You can set these registers in any order.
   a. Write Control Register 0 - SPI_CTRL0. For SPI transfers, set the serial clock polarity (SPI_CTRL0.SCPOL) and serial clock phase - SPI_CTRL0.SCPH parameters identical to the target slave device.
   b. If the transfer mode is receive only, write Control Register 1 - SPI_CTRL1) with the number of frames - SPI_CTRL1.NDF in the transfer minus 1. For example, if you want to receive 5 data frames, write 4 to SPI_CTRL1.NDF.
   c. Write the Baud Rate Select Register - SPI_BR_SL to set the baud rate for the transfer.
   d. Write the Transmit and Receive FIFO Threshold Level registers - SPI_TXFF_TH and SPI_RXFF_TH to set FIFO buffer threshold levels.
   e. Write interrupt enable and disable register - SPI_IER and SPI_IDR to control the masking of interrupts.
   f. Write the SPI Slave Enable Register - SPI_SLV_ENB to enable the target slave for selection. If a slave is enabled at this time, the transfer begins as soon as one valid data entry is present in the transmit FIFO buffer. If no slaves are enabled prior to writing to the SPI Data Register - SPI_DATA, the transfer does not begin until a slave is enabled.
3. Enable the SPI - SPI_ENABLE.SPIEN = 1.
4. Write data for transmission to the target slave into the transmit FIFO buffer (write SPI_DATA). If no slaves were enabled in the SPI_SLV_ENB at this point, enable it now to begin the transfer.
5. Poll the SPI_STAS.BUSY status to wait for the transfer to complete. If a transmit FIFO empty interrupt – 6. SPI_RIS.TXERI = 1 is asserted, write the transmit FIFO buffer (write SPI_DATA). If a receive FIFO full interrupt - SPI_RIS.RXFRI is made, read the receive FIFO buffer (read SPI_DATA).
6. The shift control logic stops the transfer when the transmit FIFO buffer is empty. If the transfer mode is receive only (SPI_CTRL0.TMOD = 2), the shift control logic stops the transfer when the specified number of frames have been received. When the transfer is done, the SPI_STAS.BUSY status is reset to 0.
7. If the transfer mode is not transmit only (SPI_CTRL0.TMOD ≠ 1), read the receive FIFO buffer until it is empty.

4.16.3.2 MASTER MICROWIRE SERIAL TRANSFERS

To use SPI as a master to perform microwire serial transfers, perform the following steps:

1. Disabled the SPI by writing a 0 to SPIEN field of the SPI_ENABLE register.
2. Set up the SPI control register - SPI_CTRL0 and SPI_CTRL1 for the transfer. You can set these registers in any order.
   a. Write SPI_CTRL0 to set transfer parameters. If the transfer is sequential and the SPI master receives data, write SPI_CTRL1.NDF with the number of frames in the transfer minus 1. For example, if you want to receive 5 data frames, write 4 to SPI_CTRL1.NDF.
   b. Write SPI_BR_SL to set the baud rate for the transfer.
   c. Write SPI_TXFF_TH and SPI_RXFF_TH to set FIFO buffer threshold levels.
   d. Write interrupt enable and disable register - SPI_IER and SPI_IDR to control the masking of interrupts.
   e. Write the SPI Slave Enable Register - SPI_SLV_ENB to enable the target slave for selection. If a slave is enabled at this time, the transfer begins as soon as one valid data entry is present in the transmit FIFO buffer. If no slaves are enabled prior to writing to the SPI Data Register - SPI_DATA, the transfer does not begin until a slave is enabled.
   f. Configure the SPI_MW_CTRL register.
3. Enable the SPI - SPI_ENABLE.SPIEN = 1.
4. If the SPI transmits data, write the control and data words into the transmit FIFO (write SPI_DATA). If the SPI master receives data, write the control word(s) into the transmit FIFO. If no slaves were enabled in the SPI_SLV_ENB at this point, enable it now to begin the transfer.
5. Poll the SPI_STAS.BUSY status to wait for the transfer to complete. If a transmit FIFO empty interrupt – 6. SPI_RIS.TXERI = 1 is asserted, write the transmit FIFO buffer (write SPI_DATA). If a receive FIFO full interrupt - SPI_RIS.RXFRI is made, read the receive FIFO buffer (read SPI_DATA)
6. The shift control logic stops the transfer when the transmit FIFO buffer is empty. If the transfer mode is sequential - SPI_MW_CTRL.MWMOD = 1 and the SPI receives data, the shift control logic stops the transfer when the specified number of frames have been received. When the transfer is done, the SPI_STAS.BUSY status is reset to 0.
7. If the transfer mode is not transmit only (SPI_CTRL0.TMOD ≠ 1), read the receive FIFO buffer until it is empty.

4.16.3.3 SLAVE SPI AND SSP SERIAL TRANSFERS

To use SPI as a slave to perform SPI or SSP serial transfers, perform the following steps:

1. Disabled the SPI by writing a 0 to SPIEN field of the SPI_ENABLE register.
2. Set up the SPI control register - SPI_CTRL0 and SPI_CTRL1 for the transfer. You can set these registers in any order.
   a. Write Control Register 0 - SPI_CTRL0. For SPI transfers, set the serial clock polarity (SPI_CTRL0.SCPOL) and serial clock phase - SPI_CTRL0.SCPH parameters identical to the master device.
   b. Write the Transmit and Receive FIFO Threshold Level registers - SPI_TXFF_TH and SPI_RXFF_TH to set FIFO buffer threshold levels.
   c. Write interrupt enable and disable register - SPI_IER and SPI_IDR to control the masking of interrupts.
3. Enable the SPI - SPI_ENABLE.SPIEN = 1.
4. If the transfer mode is transmit and receive (SPI_CTRL0.TMOD = 0) or transmit only (SPI_CTRL0.TMOD = 1), write data for transmission to the master into the transmit FIFO buffer (write SPI_DATA). If the transfer mode is receive only (SPI_CTRL0.TMOD = 2), you need not write data into the transmit FIFO buffer. The current value in the transmit shift register is retransmitted.
5. The SPI slave is now ready for the serial transfer. The transfer begins when a serial-master device selects the SPI slave.
6. When the transfer is underway, the BUSY status can be polled to return the transfer status. If a transmit FIFO empty interrupt request is made, write the transmit FIFO buffer (write SPI_DATA). If a receive FIFO full interrupt request is made, read the receive FIFO buffer (read SPI_DATA).
7. The transfer ends when the serial master removes the select input to the SPI slave. When the transfer is completed, the BUSY status is reset to 0.
8. If the transfer mode is not transmit only (SPI_CTRL0.TMOD ≠ 1), read the receive FIFO buffer until empty.

4.16.3.4 SLAVE MICROWIRE SERIAL TRANSFERS

For the SPI slave, the Microwire protocol operates in much the same way as the SPI protocol. The SPI slave does not decode the control frame.

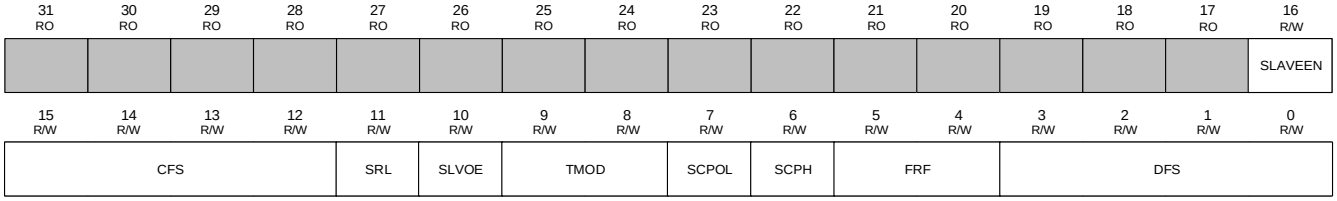
4.16.4 SPI REGISTER MAP

Base address: 0x4800_4000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	SPI_CTRL0	R/W	0x0000_0007	SPI Control Register 0	409
0x0004	SPI _CTRL1	R/W	0x0000_0000	SPI Control Register1	411
0x0008	SPI _ENABLE	R/W	0x0000_0000	SPI Enable Register	412
0x000C	SPI _MW_CTRL	R/W	0x0000_0000	Microwire Control Register	413
0x0010	SPI_SLV_ENB	R/W	0x0000_0000	SPI Slave Enable Register	414
0x0014	SPI_BR_SL	R/W	0x0000_0000	SPI Baud Rate Select	414
0x0018	SPI_TXFF_TH	R/W	0x0000_0000	SPI Transmit FIFO Threshold Level	415
0x001C	SPI_RXFF_TH	R/W	0x0000_0000	SPI Receive FIFO Threshold Level	416
0x0020	SPI_TXFF_LV	RO	0x0000_0000	SPI Transmit FIFO Level Register	417
0x0024	SPI_RXFF_LV	RO	0x0000_0000	SPI Receive FIFO Level Register	417
0x0028	SPI_STAS	RO	0x0000_0006	SPI Status Register	418
0x002C	SPI_IER	WO	0x0000_0000	SPI Interrupt Enable Register	419
0x0030	SPI_IDR	WO	0x0000_0000	SPI Interrupt Disable Register	420
0x0034	SPI_IMR	RO	0x0000_0000	SPI Interrupt Mask Status Register	421
0x0038	SPI_RIS	RO	0x0000_0000	SPI Raw Interrupt Status Register	422
0x003C	SPI_ISC	R/W1C	0x0000_0000	SPI Interrupt Status and Clear Register	423
0x0040	reserved	N/A	0x0000_0000	Reserved	-
0x0044	reserved	N/A	0x0000_0000	Reserved	-
0x0048	reserved	N/A	0x0000_0000	Reserved	-
0x004C	SPI_DMA_CTRL	R/W	0x0000_0000	SPI DMA Control Register	424
0x0050	SPI_DMA_TDLR	R/W	0x0000_0000	SPI DMA Transmit Threshold Register	424
0x0054	SPI_DMA_RDLR	R/W	0x0000_0000	SPI DMA Receive Threshold Register	425
0x0058	reserved	N/A	0x0000_0000	Reserved	-
0x005C	reserved	N/A	0x0000_0000	Reserved	-
0x0060	SPI_DATA	R/W	0x0000_0000	SPI Data Register	425

4.16.4.1 SPI _CTRL0 - SPI CONTROL REGISTER 0

This register controls the serial data transfer. It is impossible to write to this register when the SPI Master is enabled.



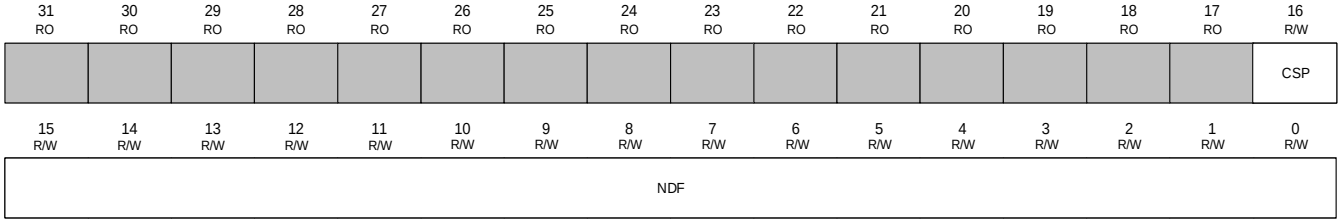
Offset: 0x0000

Bits	Name	Type	Reset	Description
31:17	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	*SLAVEEN*	R/W	0	Slave Mode Enable 0: Master Mode 1: Slave Mode
15:12	*CFS*	R/W	0x0	Control Frame Size This field selects the length of control word for the Microwire frame format. 0x0: 1-bit control word 0x1: 2-bit control word 0x2: 3-bit control word 0x3: 4-bit control word 0x4: 5-bit control word 0x5: 6-bit control word 0x6: 7-bit control word 0x7: 8-bit control word 0x8: 9-bit control word 0x9: 10-bit control word 0xA: 11-bit control word 0xB: 12-bit control word 0xC: 13-bit control word 0xD: 14-bit control word 0xE: 15-bit control word 0xF: 16-bit control word
11	*SRL*	R/W	0	Shift Register Loop This bit is used for testing only. When internally active, connects the transmit shift register output to the receive shift register input. 0: Normal Moe Operation 1: Test Mode Operation
10	*SLVOE*	R/W	0	Slave Output Enable This bit is functional only when the SPI is configured as serial-slave device. 0: Slave Transmit data signal (MISO) is enabled. 1: Slave Transmit data signal (MISO) is disabled.
9:8	*TMOD*	R/W	0x0	Transfer Mode This field selects the mode of transfer for serial communication. 00: Transmit & Receive. 0x1: Transmit Only. 0x2: Receive Only. 0x3: EEPROM Read.

Bits	Name	Type	Reset	Description
7	*SCPOL*	R/W	0	Serial Clock Polarity Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the SPI Master is not actively transferring data on the serial bus. 0: Inactive state of serial clock is low. 1: Inactive state of serial clock is high.
6	*SCPH*	R/W	0	Serial Clock Phase Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relationship of the serial clock with the slave select signal. 0: Serial clock toggles in middle of first data bit. 1: Serial clock toggles at start of first data bit.
5:4	*FRF*	R/W	0x0	Frame Format Selects which serial protocol transfer the data. 00: Motorola SPI. 01: Texas Instruments SSP. 10: National Semiconductors Microwire. 11: Reserved.
3:0	*DFS*	R/W	0x7	Data Frame Size Selects the data frame length. Not Support and used value range is 0x0 - 0x2. The length is (DFS+1) bits. 0x0 - 0x2: Reserved. 0x3: 4-bit serial data transfer 0x4: 5-bit serial data transfer 0x5: 6-bit serial data transfer 0x6: 7-bit serial data transfer 0x7: 8-bit serial data transfer 0x8: 9-bit serial data transfer 0x9: 10-bit serial data transfer 0xA: 11-bit serial data transfer 0xB: 12-bit serial data transfer 0xC: 13-bit serial data transfer 0xD: 14-bit serial data transfer 0xE: 15-bit serial data transfer 0xF: 16-bit serial data transfer

4.16.4.2 SPI _CTRL1 - SPI CONTROL REGISTER 1

This register controls the end of serial transfers when in receive-only mode.

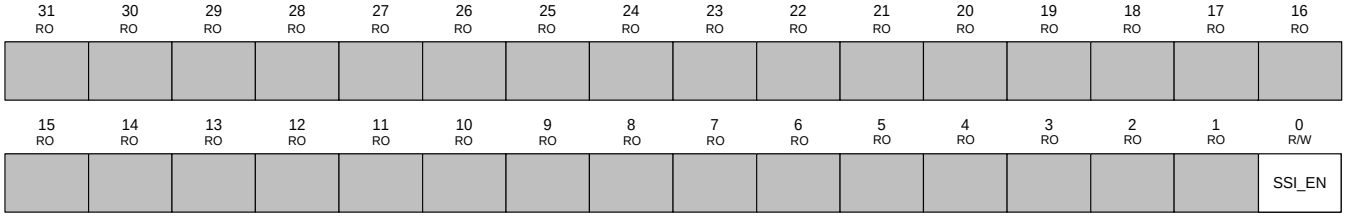


Offset: 0x0004

Bits	Name	Type	Reset	Description
31:17	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	*CSP*	R/W	0	Chip Select Pulse Management This bit is used in master mode only. It does not allow the SPI to generate a CS pulse between two consecutive data when doing continuous transfers. In the case of a single data transfer, it forces the CS pin high level after the transfer. It has no meaning if **SPI_CTRL0**.*SCPH* = '1', and only used in **SPI_CTRL0**.*FRF* = '00'. 0: NSS pulse generated 1: No NSS pulse
15:0	*NDF*	R/W	0	Number of Data Frames When *TMOD* = 10, this register field sets the number of data frames to be continuously received by the SPI. The SPI continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64 KB of data in a continuous transfer. When the SPI is configured as a serial slave, the transfer continues for as long as the slave is selected. Therefore, this register serves no purpose and is not present when the SPI is configured as a serial slave.

4.16.4.3 SPI_ENABLE - SPI ENABLE REGISTER

This register is used to enable SPI.

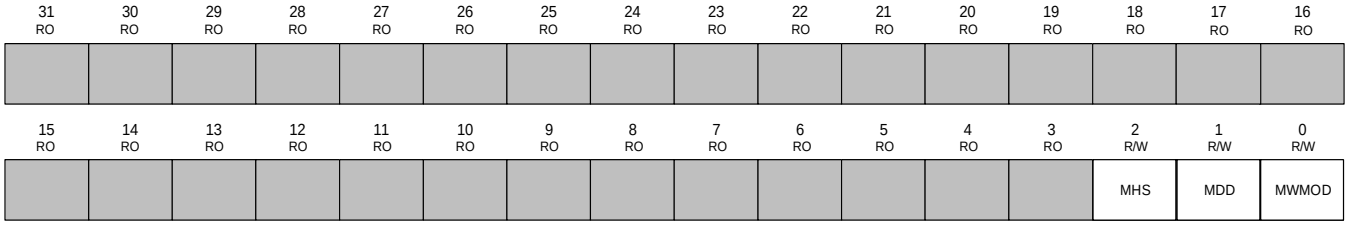


Offset: 0x0008

Bits	Name	Type	Reset	Description
31:1	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*SSI_EN*	R/W	0	SSI Enable Enables and disables all SPI operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. It is impossible to program some of the SPI control registers when enabled. 0: Disable SPI 1: Enable SPI

4.16.4.4 SPI_MW_CTRL - MICROWIRE CONTROL REGISTER

This register controls the direction of the data word for the half-duplex Microwire serial protocol.

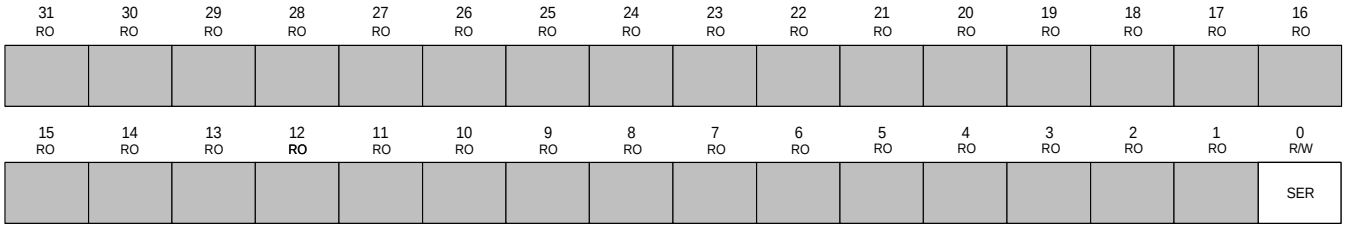


Offset: 0x000C

Bits	Name	Type	Reset	Description
31:3	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*MHS*	R/W	0	Microwire Handshaking This bit is only functional when the SPI is configured as serial-master device. Used to enable and disable the “busy/ready” handshaking interface for the Microwire protocol. When enabled, the SPI checks for a ready status from the target slave, after the transfer of the last data/control bit, before clearing the BUSY status in the SR register. 0: handshaking interface is disabled. 1: handshaking interface is enabled.
1	*MDD*	R/W	0	Microwire Control Defines the direction of the data word when the Microwire serial protocol is used.
0	*MWMOD*	R/W	0	Microwire Transfer Mode Defines whether the Microwire transfer is sequential or non-sequential. When sequential mode is used, only one serial data transfer is needed to transmit or receive a block of data words. When non-sequential mode is used, there must be a control word for each data word that is transmitted or received. 0: non-sequential transfer. 1: sequential transfer.

4.16.4.5 SPI_SLV_ENB - SPI SLAVE ENABLE REGISTER

This register is valid only when the SPI is configured as a master device. The register enables the individual slave select output lines from the SPI Master.

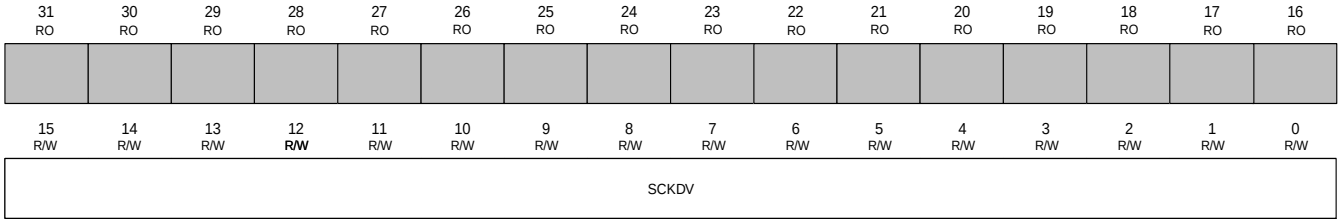


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:1	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*SER*	R/W	0x0	Slave Select Enable Flag When the bit is set, the corresponding slave select line from the master is activated when a serial transfer begins. It should be noted that setting or clearing bits in this register have no effect on the corresponding slave select outputs until a transfer is started. Before beginning a transfer, you should enable the bit in this register that corresponds to the slave device with which the master wants to communicate. 0: Slave is not selected 1: Slave is selected.

4.16.4.6 SPI_BR_SL - SPI BAUD RATE SELECT

This register is valid only when the SPI is configured as a master device. This register derives the frequency of the serial clock that regulates the data transfer.

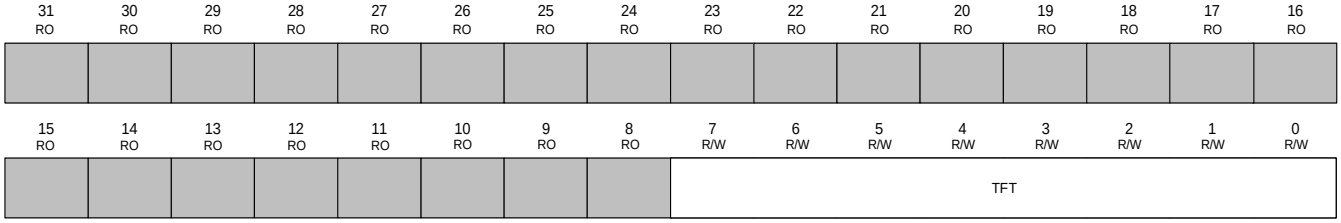


Offset: 0x0014

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*SCKDV*	R/W	0x0	SPI Clock Divider The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is below 3, the serial output clock (SPI_SCLK) is disabled. The frequency of the SPI_SCLK is derived from the following equation: $F_{SPI_SCLK} = F_{SPI\ Clock} / SCKDV$, where SCKDV is any even value between 4 and 65534. For example: $F_{SPI\ Clock}$ = 16 MHz, then SCKDV = 4, F_{SPI_SCLK} = 16 / 4 = 4 MHz

4.16.4.7 SPI_TXFF_TH - SPI TRANSMIT FIFO THRESHOLD LEVEL

This register controls the threshold value for the transmit FIFO memory.

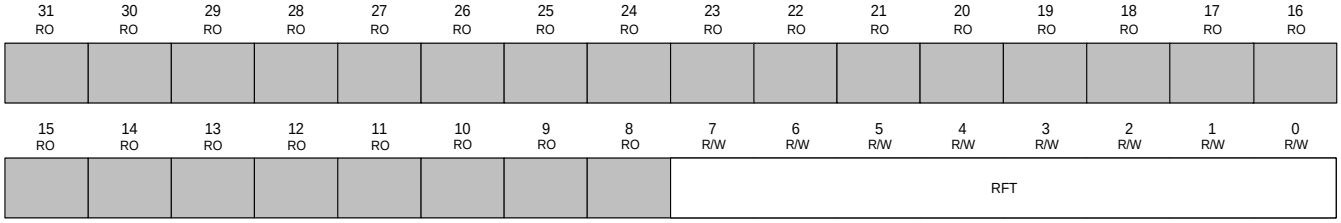


Offset: 0x0018

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*TFT*	R/W	0x0	Transmit FIFO Threshold Controls the level of entries (or below) at which the transmit FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2-8; this register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered. 000: Interrupt is asserted when 0 data entries are present in transmit FIFO. 001: Interrupt is asserted when 1 data entries are present in transmit FIFO. 010: Interrupt is asserted when 2 data entries are present in transmit FIFO. 011: Interrupt is asserted when 3 data entries are present in transmit FIFO. 100: Interrupt is asserted when 4 data entries are present in transmit FIFO. 101: Interrupt is asserted when 5 data entries are present in transmit FIFO. 110: Interrupt is asserted when 6 data entries are present in transmit FIFO. 111: Interrupt is asserted when 7 data entries are present in transmit FIFO.

4.16.4.8 SPI _RXFF_TH - SPI RECEIVE FIFO THRESHOLD LEVEL

The register controls the threshold value for the receive FIFO memory

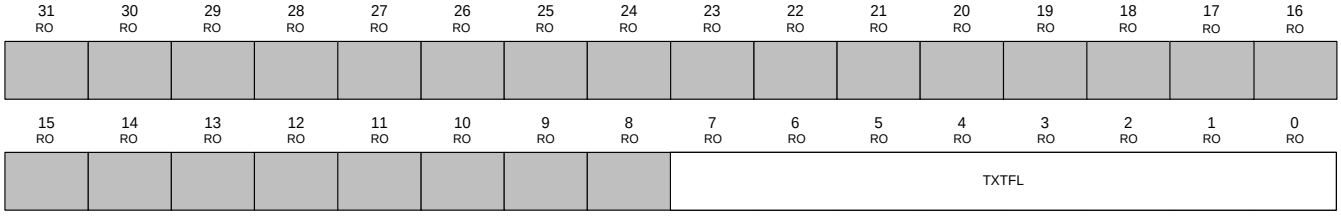


Offset: 0x001C

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*RFT*	R/W	0x0	Receive FIFO Threshold Controls the level of entries (or above) at which the receive FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2-8. This registers sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered. 000: Interrupt is asserted when 1 or more data entry is present in receive FIFO. 001: Interrupt is asserted when 2 or more data entry is present in receive FIFO. 010: Interrupt is asserted when 3 or more data entry is present in receive FIFO. 011: Interrupt is asserted when 4 or more data entry is present in receive FIFO. 100: Interrupt is asserted when 5 or more data entry is present in receive FIFO. 101: Interrupt is asserted when 6 or more data entry is present in receive FIFO. 110: Interrupt is asserted when 7 or more data entry is present in receive FIFO. 111: Interrupt is asserted when 8 or more data entry is present in receive FIFO.

4.16.4.9 SPI_TXFF_LV - SPI TRANSMIT FIFO LEVEL REGISTER

This register contains the number of valid data entries in the transmit FIFO memory.

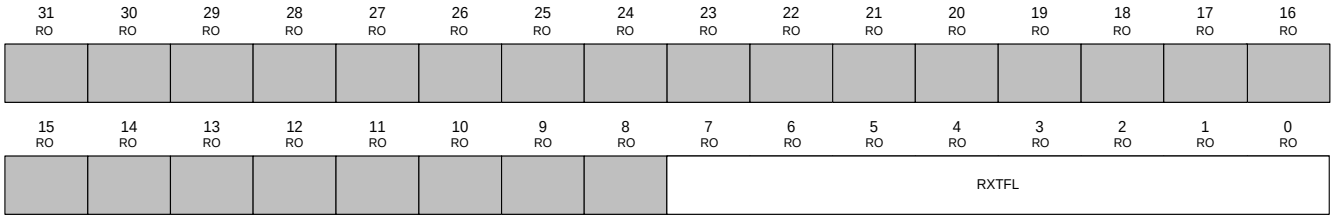


Offset: 0x0020

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*TXTFL*	RO	0x0	Transmit FIFO Level Contains the number of valid data entries in the transmit FIFO.

4.16.4.10 SPI _RXFF_LV - SPI RECEIVE FIFO LEVEL REGISTER

This register contains the number of valid data entries in the receive FIFO memory.

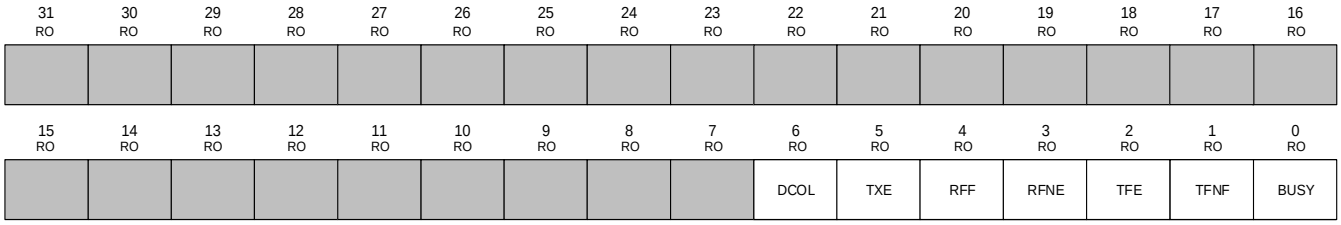


Offset: 0x0024

Bits	Name	Type	Reset	Description
31:8	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
7:0	*RXTFL*	RO	0x0	Receive FIFO Level Contains the number of valid data entries in the receive FIFO.

4.16.4.11 SPI _STAS - SPI STATUS REGISTER

This is a read-only register indicating the current transfer status, FIFO status, and any transmission/reception errors that may have occurred.

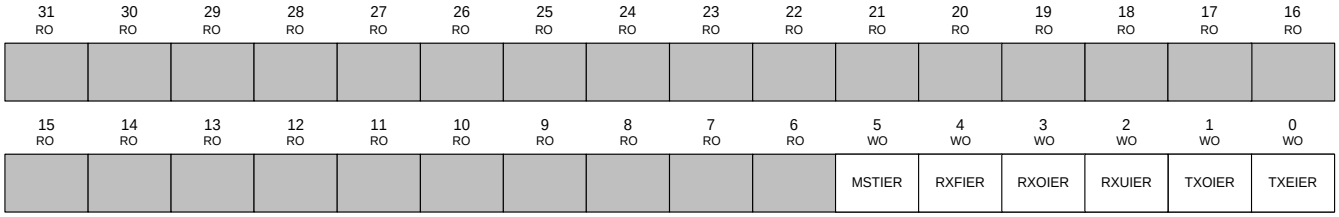


Offset: 0x0028

Bits	Name	Type	Reset	Description
31:7	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6	*DCOL*	RO	0	Data Collision Error Relevant only when the SPI is configured as a master device. This bit is set if the SPI master is actively transmitting when another master selects this device as a slave. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read. 0: No error. 1: Transmit data collision error.
5	*TXE*	RO	0	Transmission Error Set if the transmit FIFO is empty when a transfer is started. This bit can be set only when the SPI is configured as a slave device. Data from the previous transmission is resent on the txd line. This bit is cleared when read. 0: No error. 1: Transmission error.
4	*RFF*	RO	0	Receive FIFO Full When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0: Receive FIFO is not full. 1: Receive FIFO is full.
3	*RFNE*	RO	0	Receive FIFO Not Empty Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. 0: Receive FIFO is empty. 1: Receive FIFO is not empty
2	*TFE*	RO	1	Transmit FIFO Empty When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0: Transmit FIFO is not empty. 1: Transmit FIFO is empty
1	*TFNF*	RO	1	Transmit FIFO Not Full Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0: Transmit FIFO is full. 1: Transmit FIFO is not full

Bits	Name	Type	Reset	Description
0	*BUSY*	RO	0	SPI Busy Flag. When set, indicates that a serial transfer is in progress; when cleared indicates that the SPI is idle or disabled. 0: SPI is idle or disabled. 1: SPI is actively transferring data.

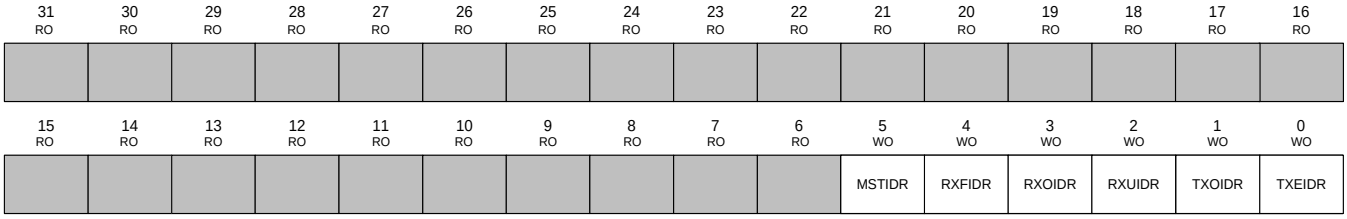
4.16.4.12 SPI _IER - SPI INTERRUPT ENABLE REGISTER



Offset: 0x002C

Bits	Name	Type	Reset	Description
31:6	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTIER*	WO	0	Multi-Master Contention Interrupt Enable 1: Interrupt is enabled.
4	*RXFIER*	WO	0	Receive FIFO Full Interrupt Enable 1: Interrupt is enabled.
3	*RXOIER*	WO	0	Receive FIFO Overflow Interrupt Enable 1: Interrupt is enabled.
2	*RXUIER*	WO	0	Receive FIFO Underflow Interrupt Enable 1: Interrupt is enabled.
1	*TXOIER*	WO	0	Transmit FIFO Overflow Interrupt Enable 1: Interrupt is enabled.
0	*TXEIER*	WO	0	Transmit FIFO Empty Interrupt Enable 1: Interrupt is enabled.

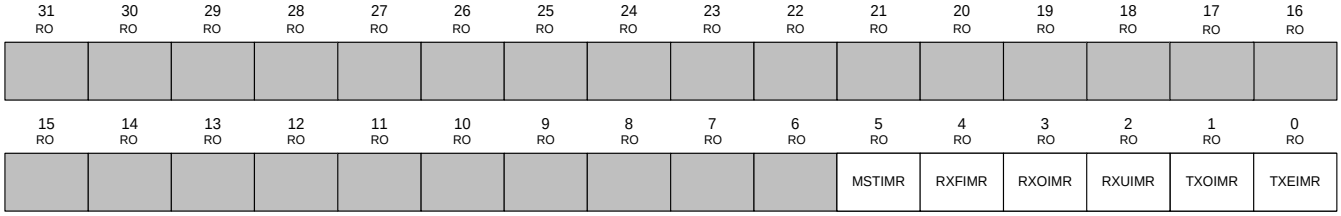
4.16.4.13 SPI_IDR - SPI INTERRUPT DISABLE REGISTER



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:6	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTIDR*	WO	0	Multi-Master Contention Interrupt Disable 1: Interrupt is disabled.
4	*RXFIDR*	WO	0	Receive FIFO Full Interrupt Disable 1: Interrupt is disabled.
3	*RXOIDR*	WO	0	Receive FIFO Overflow Interrupt Disable 1: Interrupt is disabled.
2	*RXUIDR*	WO	0	Receive FIFO Underflow Interrupt Disable 1: Interrupt is disabled.
1	*TXOIDR*	WO	0	Transmit FIFO Overflow Interrupt Disable 1: Interrupt is disabled.
0	*TXEIDR*	WO	0	Transmit FIFO Empty Interrupt Disable 1: Interrupt is disabled.

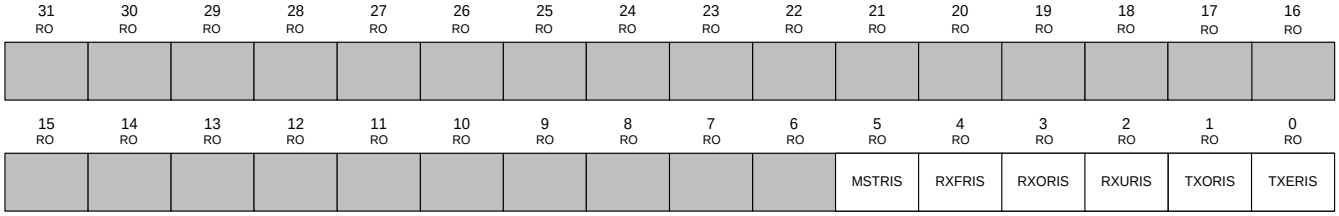
4.16.4.14 SPI _IMR - SPI INTERRUPT MASK STATUS REGISTER



Offset: 0x0034

Bits	Name	Type	Reset	Description
31:6	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTIMR*	RO	0	Multi-Master Contention Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
4	*RXFIMR*	RO	0	Receive FIFO Full Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
3	*RXOIMR*	RO	0	Receive FIFO Overflow Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
2	*RXUIMR*	RO	0	Receive FIFO Underflow Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
1	*TXOIMR*	RO	0	Transmit FIFO Overflow Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
0	*TXEIMR*	RO	0	Transmit FIFO Empty Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked

4.16.4.15 SPI_RIS - SPI RAW INTERRUPT STATUS REGISTER



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:6	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTRIS*	RO	0	Multi-Master Contention Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
4	*RXFRIS*	RO	0	Receive FIFO Full Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
3	*RXORIS*	RO	0	Receive FIFO Overflow Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
2	*RXURIS*	RO	0	Receive FIFO Underflow Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
1	*TXORIS*	RO	0	Transmit FIFO Overflow Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
0	*TXERIS*	RO	0	Transmit FIFO Empty Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.16.4.16 SPI _ISC - SPI INTERRUPT STATUS AND CLEAR REGISTER

Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt.

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

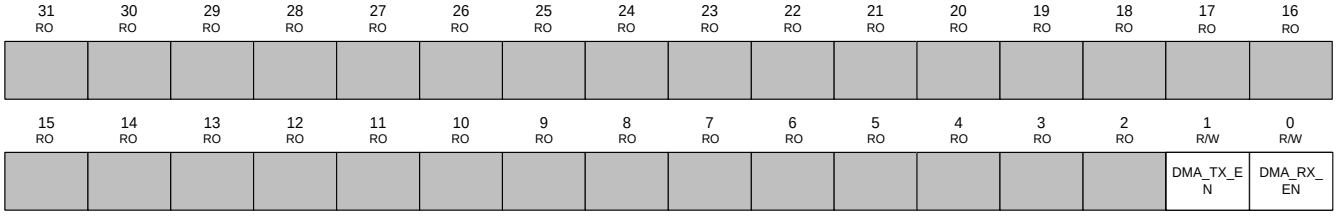
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
										MSTIC	RXFIC	RXOIC	RXUIC	TXOIC	TXEIC

Offset: 0x003C

Bits	Name	Type	Reset	Description
31:6	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTIC*	R/W1C	0	Multi-Master Contention Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
4	*RXFIC*	R/W1C	0	Receive FIFO Full Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
3	*RXOIC*	R/W1C	0	Receive FIFO Overflow Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
2	*RXUIC*	R/W1C	0	Receive FIFO Underflow Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
1	*TXOIC*	R/W1C	0	Transmit FIFO Overflow Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
0	*TXEIC*	R/W1C	0	Transmit FIFO Empty Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.

4.16.4.17 SPI_DMA_CTRL - SPI DMA CONTROL REGISTER

This register is used to enable the DMA Controller interface operation.

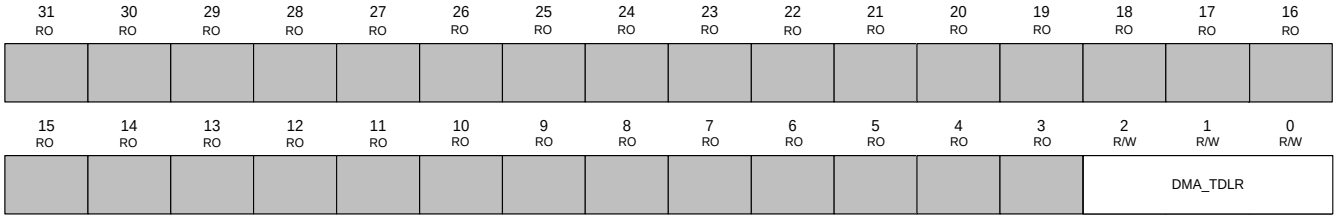


Offset: 0x004C

Bits	Name	Type	Reset	Description
31:2	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*DMA_TX_EN*	R/W	0	DMA Transmit Enable 0: Transmit DMA disabled 1: Transmit DMA enabled
0	*DMA_RX_EN*	R/W	0	DMA Receive Enable 0: Receive DMA disabled 1: Receive DMA enabled

4.16.4.18 SPI_DMA_TDLR - SPI DMA TRANSMIT THRESHOLD REGISTER

Controls the FIFO Level for a DMA transmit request.

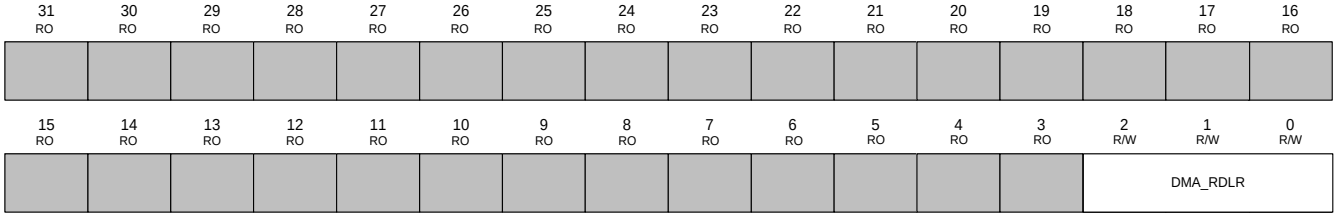


Offset: 0x0050

Bits	Name	Type	Reset	Description
31:3	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*DMA_TDLR*	R/W	0x0	DMA Transmit Data Level Controls the level of entries (or below) at which the transmit FIFO controller triggers a DMA transmit request. When the number of transmit FIFO entries is less than or equal to this value, the DMA transmit request is triggered.

4.16.4.19 SPI_DMA_RDLR - SPI DMA RECEIVE THRESHOLD REGISTER

Controls the FIFO Level for a DMA receive request.

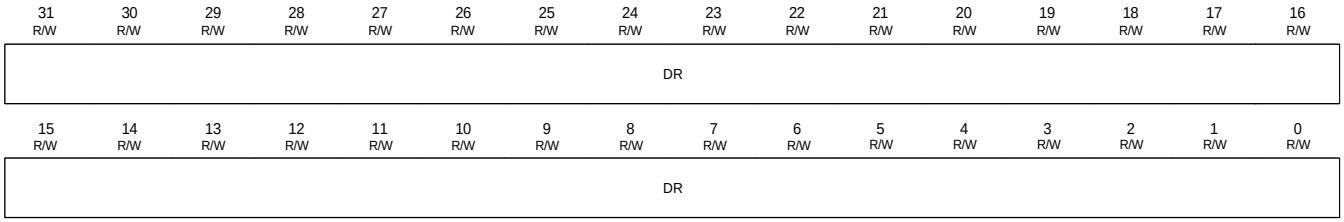


Offset: 0x0054

Bits	Name	Type	Reset	Description
31:3	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*DMA_RDLR*	R/W	0x0	DMA Receive Data Level Controls the level of entries (or above) at which the receive FIFO controller triggers a DMA receive request. When the number of receive FIFO entries is greater than or equal to this value + 1, the DMA receive request is triggered.

4.16.4.20 SPI _DATA - SPI DATA REGISTER

When the register is read, data in the receive FIFO buffer is accessed. When it is written to, data are moved into the transmit FIFO buffer.



Offset: 0x0060

Bits	Name	Type	Reset	Description
31:0	*DR*	R/W	0	Data Register When writing to this register, you must right-justify the data, while reading data are automatically right-justified. Read: Receive FIFO buffer. Write: Transmit FIFO buffer.

4.17 INTER-IC SOUND (I2S)

The Inter-IC Sound (I2S) protocol is a synchronous serial communication interface which can operate in slave or master mode to exchange data with other audio peripherals, such as ADC or DAC. I2S supports a variety of data formats, it can address four different audio standards including the Philips I2S standard, the MSB- and LSB- justified standards and the PCM standards. When the I2S operates in the master mode, then when using the integer fractional divider, it can provide an accurate sampling frequency output and support the rate control function and fine-tuning of the output frequency to avoid any system crash caused by the cumulative frequency error between different devices.

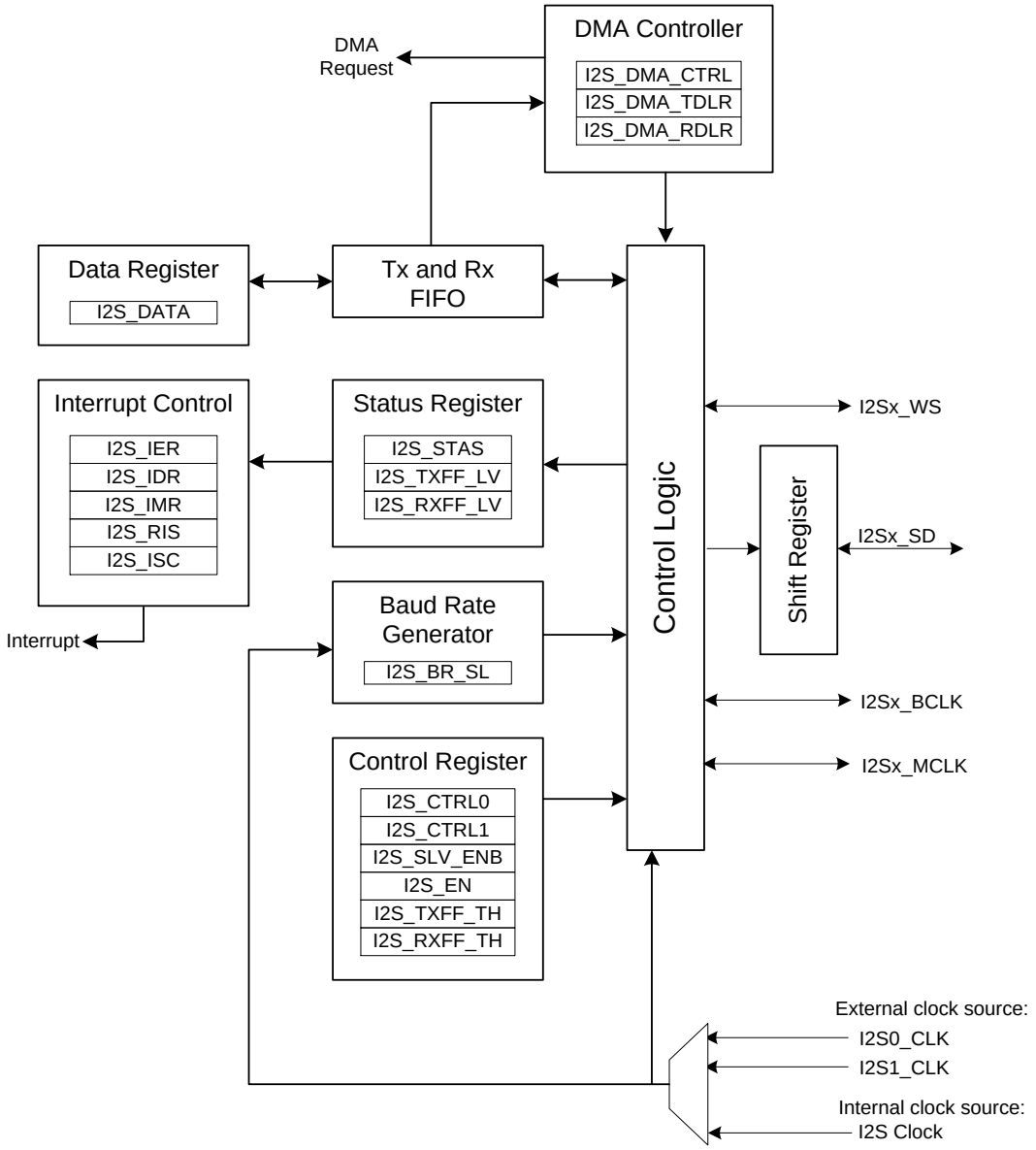
In PT32C302, I2S modules can be configured as Serial Peripheral Interface (SPI) providing SPI protocol.

I2S Main Features:

- Master/Slave Operation
- Half-duplex communication
- Configurable to SPI protocol
- 16-bit programmable linear prescaler to reach accurate audio sample frequencies (from 8kHz to 192kHz)
- 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
- Data direction is always MSB first
- Master clock can be output to drive an external audio component.
- Supported I^2S 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).
- One 32-bit embedded Rx or Tx FIFOs support.
- DMA capability for transmission and reception

4.17.1 BLOCK DIAGRAM

Figure 4.17-1: I2S Block Diagram



IMPORTANT: All I2S pins are configured as SPI pins by default. To use I2S function, user first set MODE bit in I2S_CTRL2 register (bit 9) to enable the I2S protocol.

In PT32C302, I2S shares four common pins with the SPI and I2S module is allowed to be configured as SPI module and perform any SPI functions.

Table 4.17-1: I2S & SPI Pin Mapping

I2S	Mapped SPI	Description
BCLK	SCLK	The serial clock output in master mode and serial clock input in slave mode
WS	SCSN	The data control signal output in master mode and input in slave mode.
SD	MISO	Transmit or receive the two time multiplexed data channels (in half-duplex mode only).
MCLK	MOSI	Master clock output (only when the I2S is configured in master mode)

An additional pin can be used to provide external clock source for I2S module in PT32C302:

- I2S0_CLK; I2S1_CLK

4.17.2 FUNCTIONAL DESCRIPTION

4.17.2.1 I2S MASTER AND SLAVE MODE

The I²S can operate in slave or master mode. Within the I²S module, the difference between these modes lies in the word select (WS) signal which determines the timing of data transmissions.

- In the master mode, the word select signal is generated internally by a clock rate generator.
- In the slave mode, the word select signal is input on the I2S_WS pin.
- When an I²S bus is enabled, the word select, bit clock signals are sent continuously by the bus master.

Figure 4.17-2: The I2S is Master and connect to Slave to transmit data

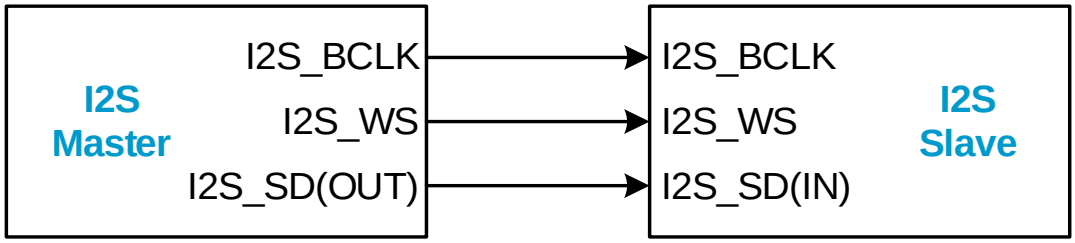


Figure 4.17-3: The I2S is Master and connect to Slave, and receive data

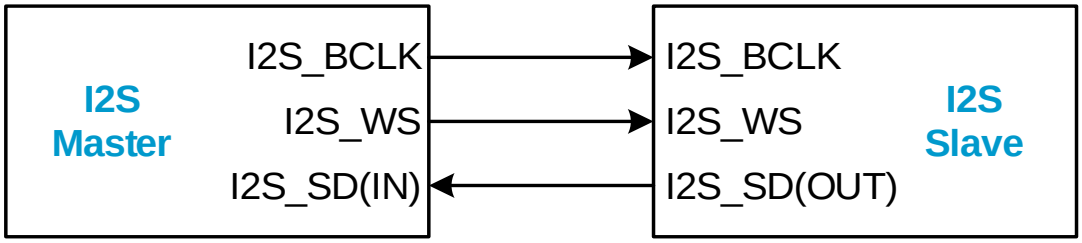
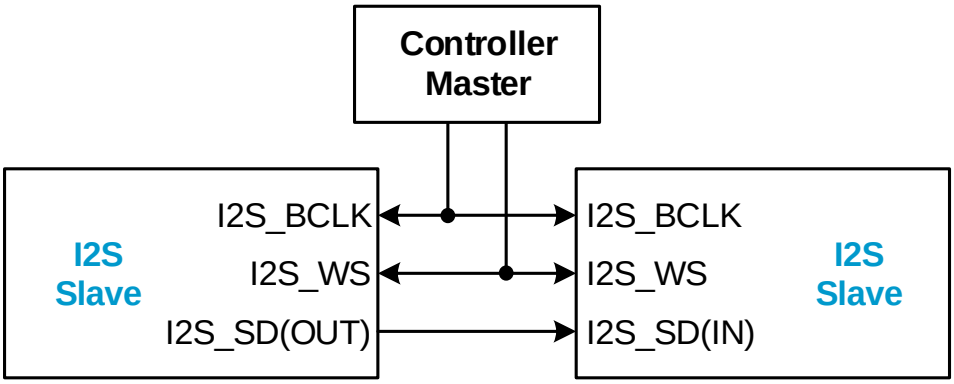


Figure 4.17-4: Slave to Slave communication with external Controller Master



4.17.2.2 I2S INTERFACE FORMAT

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 using 16-bit data extended on 32-bit packet, the first 16 bits (MSB) are the significant bits, the 16-bit LSB is forced to 0 without any need for software action or DMA request (only one read/write operation).

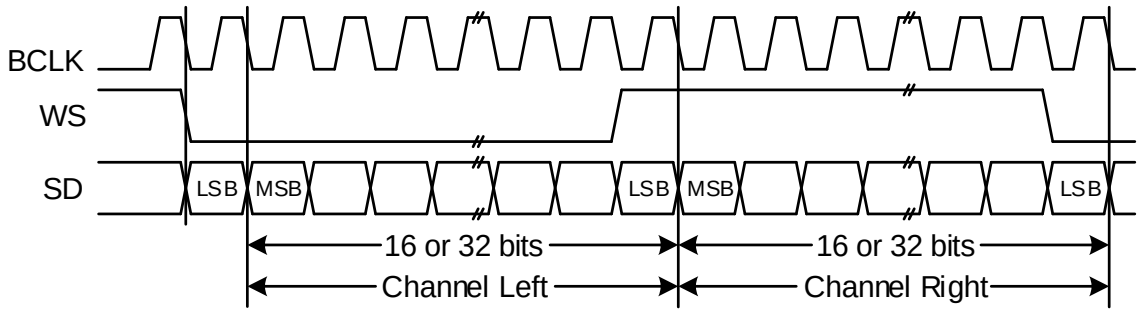
The 24-bit and 32-bit data frames need two CPU read or write operations to/from the **I2Sx_DATA** register or two DMA operations if the DMA is preferred for the application.

For all data formats and communication standards, the most significant bit is always sent first (MSB first).

I2S PHILIPS STANDARD

For this standard, the WS signal is used to indicate which channel is being transmitted. It is activated one BCLK clock cycle before the first bit (MSB) is available.

Figure 4.17-5: I2S Philips protocol waveforms (16/32-bit full accuracy, *CKPOL* = 0)



Data are latched on the falling edge of BCLK (for the transmitter) and are read on the rising edge (for the receiver). The WS signal is also latched on the falling edge of BCLK.

Figure 4.17-6: I2S Philips Standard Waveforms (24-bit frame with *CKPOL* = 0)

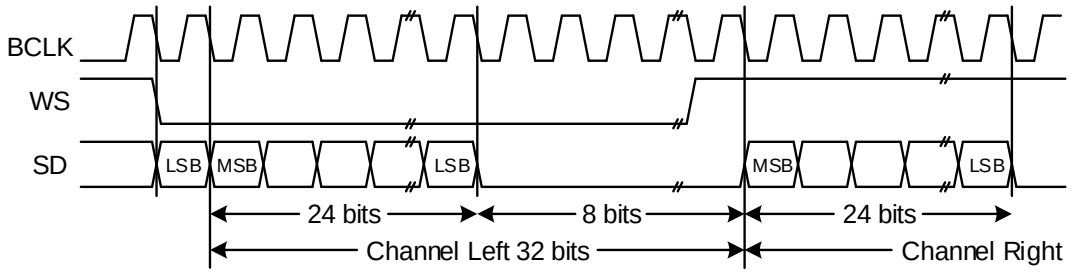
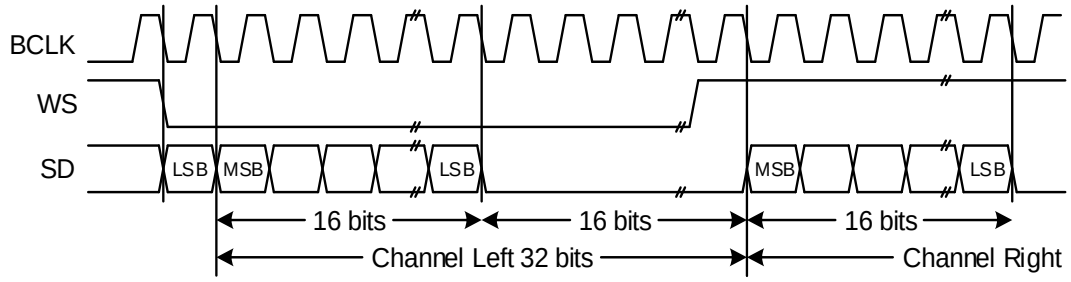


Figure 4.17-7: I2S Philips standard (16-bit extended to 32-bit packet frame with *CKPOL* = 0)



When 16-bit data frame extended to 32-bit channel frame is selected during the I2S configuration phase, only one access to the **I2Sx_DATA** register is required. The 16 remaining bits are forced by hardware to 0x0000 to extend the data to 32-bit format.

FIFO OPERATION

Figure 4.17-8: Two data words to transmit, *DIR* = 0

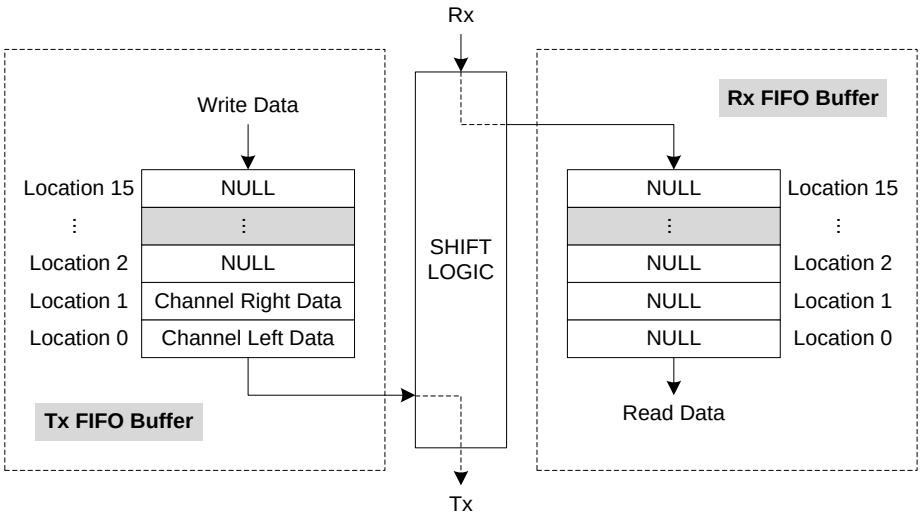


Figure 4.17-8 shows the FIFO levels prior to the beginning of a serial transfer and the FIFO levels on completion of the transfer. In this example, two data words (channel left and channel right data) are transmitted from the I2S to the external serial device in a continuous transfer. In this mode, the transfer data must be even to write to the TX FIFO. The transmission direction was controlled by *DIR* bit at **I2S_CTRL2** register.

Figure 4.17-9: Two data words to receive, *DIR* = 1

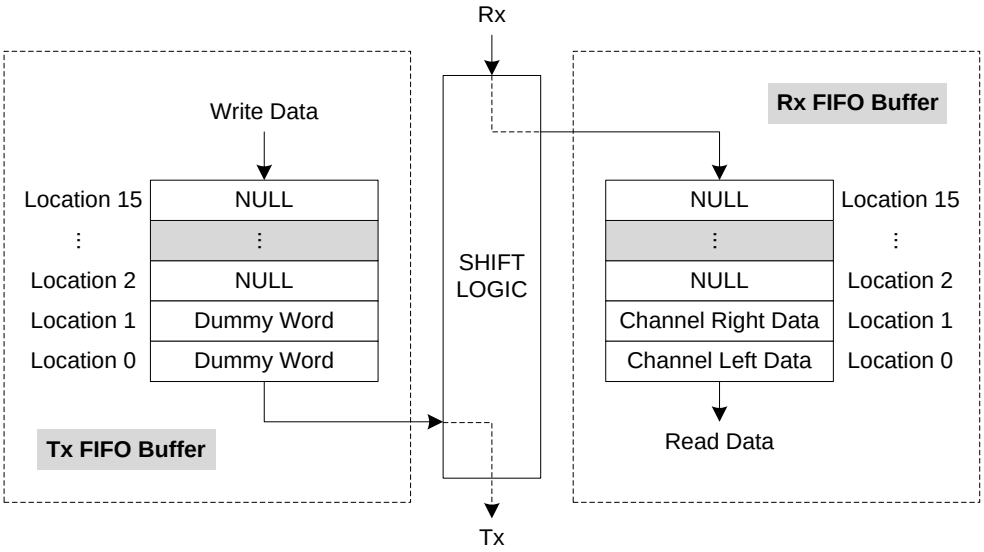
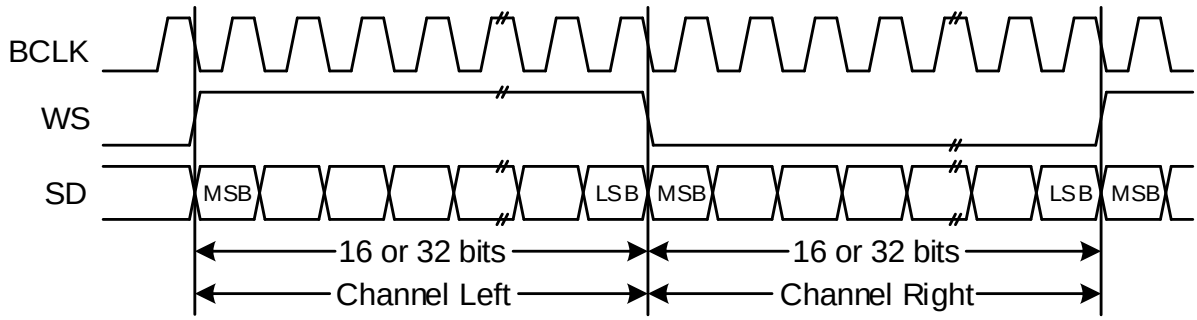


Figure 4.17-9 shows the FIFO levels prior to the beginning of a serial transfer and the FIFO levels on completion of the transfer. In this example, two data words are received by the I2S from the external serial device in a continuous transfer. Here have two dummy words write to TX FIFO, then received two data words (channel left and channel right data).

MSB-JUSTIFIED STANDARD

For this standard, the WS signal is generated at the same time as the first data bit, which is the MSB bit.

Figure 4.17-10: MSB Justified 16-bit or 32-bit full-accuracy length with *CKPOL* = 0



Data are latched on the falling edge of BCLK (for transmitter) and are read on the rising edge (for the receiver).

Figure 4.17-11: MSB justified 24-bit frame length with *CKPOL*= 0

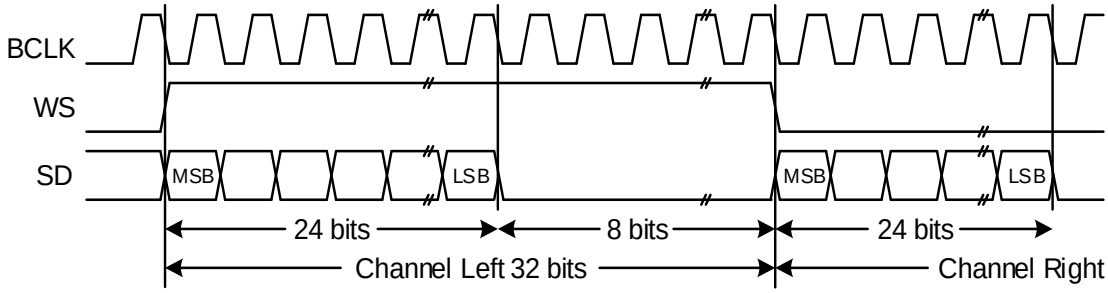
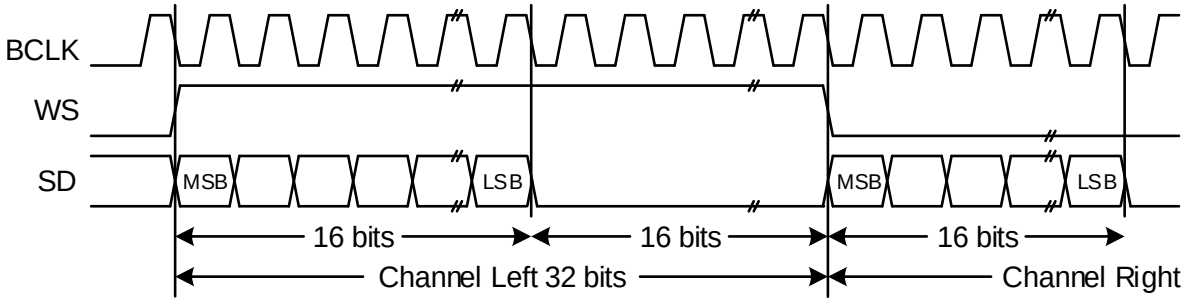


Figure 4.17-12: MSB justified 16-bit extended to 32-bit packet frame with *CKPOL*= 0



FIFO OPERATION

The FIFO's operation is the same with I²S Philips standard.

LSB-JUSTIFIED STANDARD

This standard is similar to the MSB justified standard (no difference for the 16-bit and 32-bit full-accuracy frame formats).

Figure 4.17-13: LSB justified 16-bit or 32-bit full-accuracy with *CKPOL* = 0

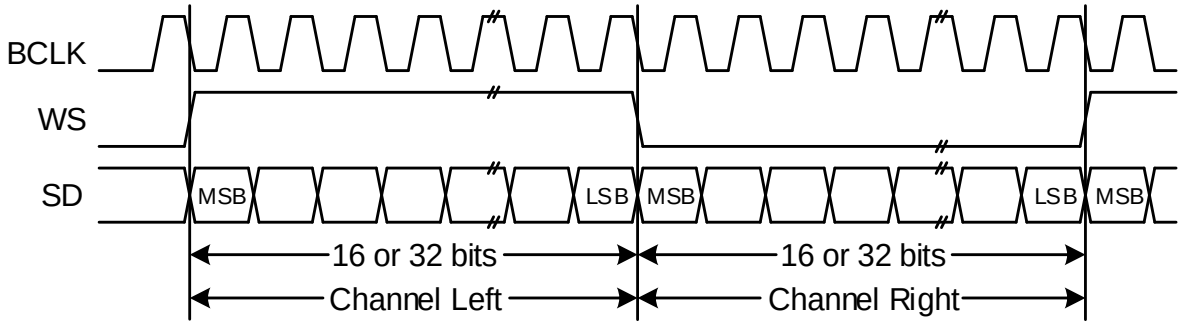


Figure 4.17-14: LSB justified 24-bit frame length with CKPOL= 0

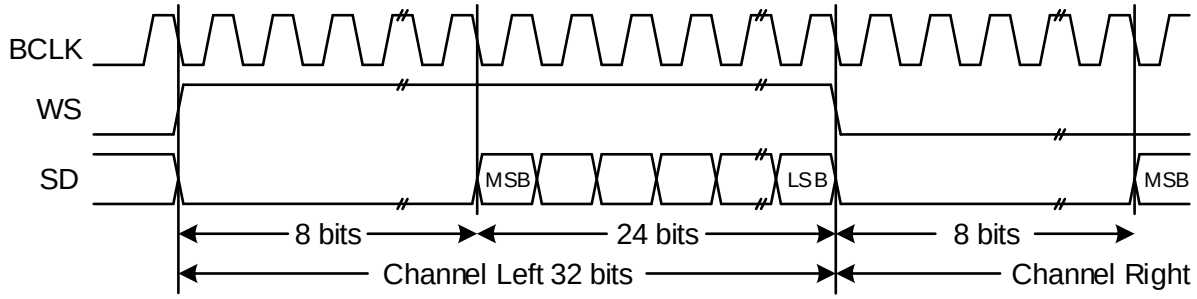
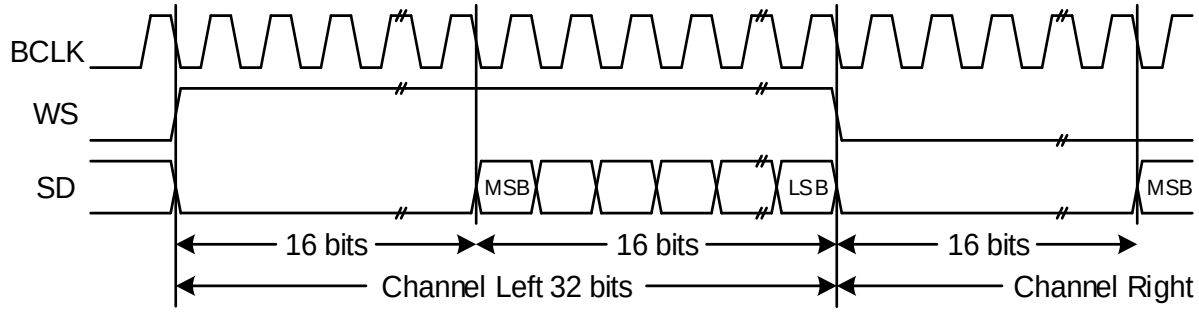


Figure 4.17-15: LSB justified 16-bit extended to 32-bit packet frame with CKPOL= 0



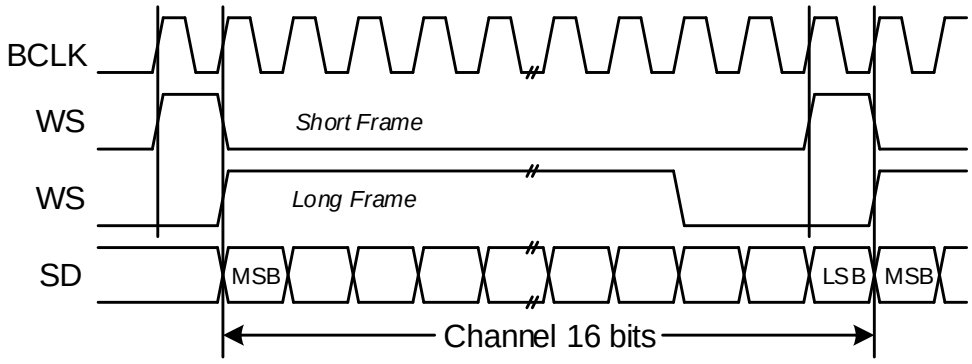
FIFO OPERATION

The FIFO's operation is the same with I^2S Philips standard.

PCM STANDARD

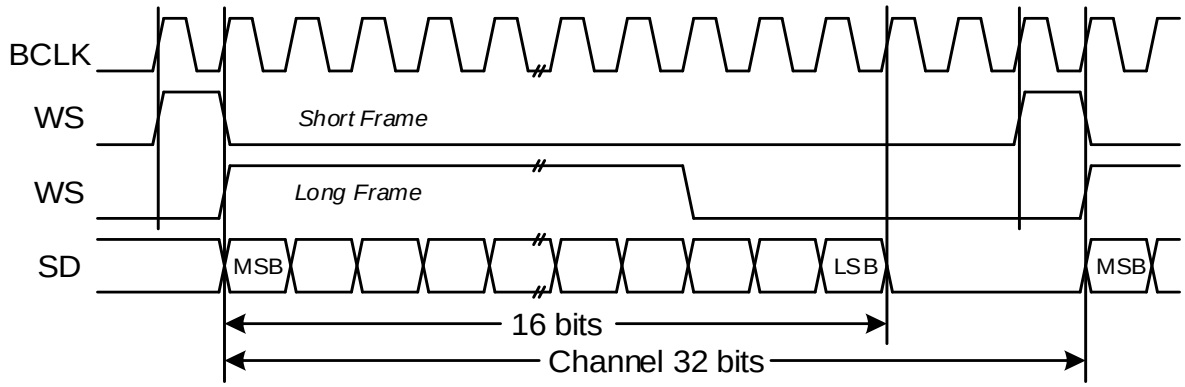
For the PCM standard, there is no need to use channel-side information. The two PCM modes (short and long frame) are available and configurable using the *PCMSYNC* bit in **I2S_CTRL1** register.

Figure 4.17-16: PCM standard waveforms (16-bit)



- For long frame synchronization, the WS signal assertion time is fixed to 13 bits in master mode.
- For short frame synchronization, the WS synchronization signal is only one cycle long.

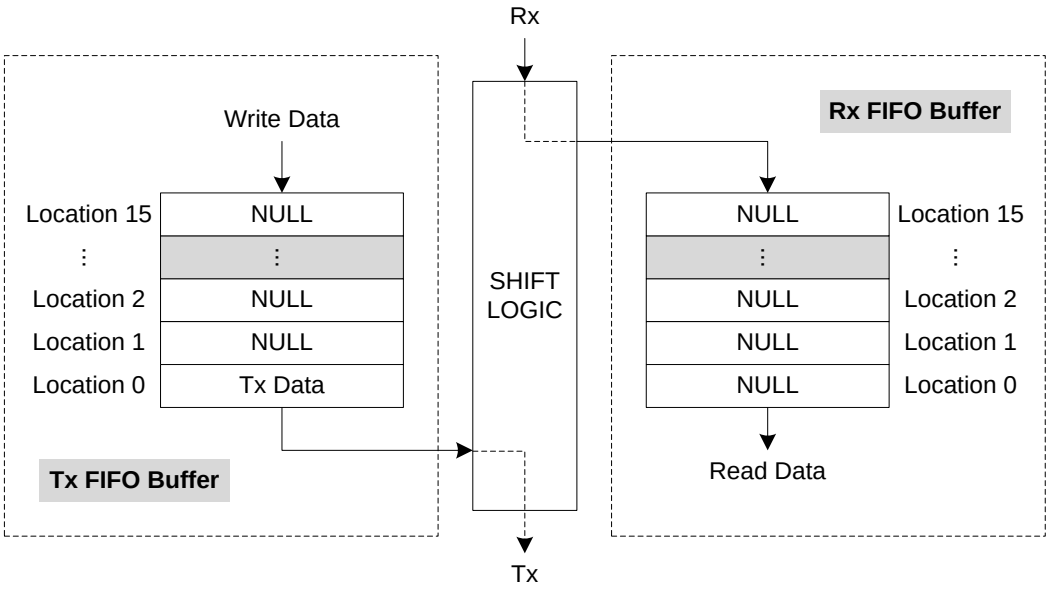
Figure 4.17-17: PCM standard waveforms (16-bit extended to 32-bit packet frame)



FIFO OPERATION

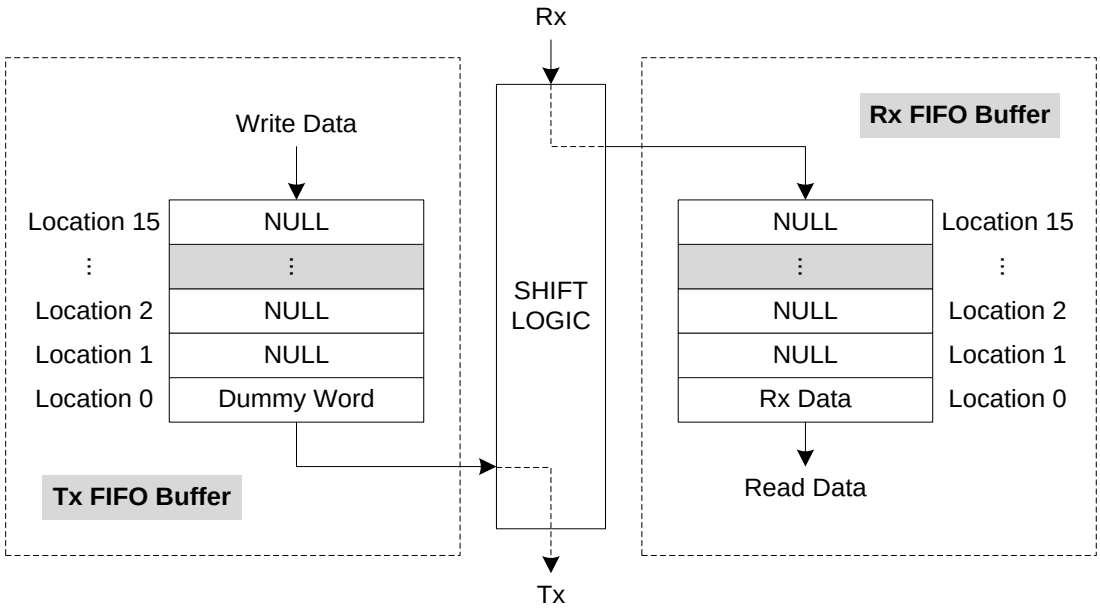
On the below figure, shows the FIFO levels prior to the beginning of a serial transfer and the FIFO levels on completion of the transfer. In this example, one data words are transmitted from the I2S to the external serial device in a continuous transfer.

Figure 4.17-18: One data words to transmit



On the below figure, shows the FIFO levels prior to the beginning of a serial transfer and the FIFO levels on completion of the transfer. In this example, one data words are received by the I2S from the external serial device in a continuous transfer.

Figure 4.17-19: One data words to receive



4.17.2.3 DMA CONTROLLER

In I2S mode, the DMA works in exactly the same way as it does in SPI mode. Refer to SPI DMA Controller for detailed description.

4.17.3 I2S REGISTER MAP

NOTE: Refer to SPI Register Map if the I2S is configured as SPI.

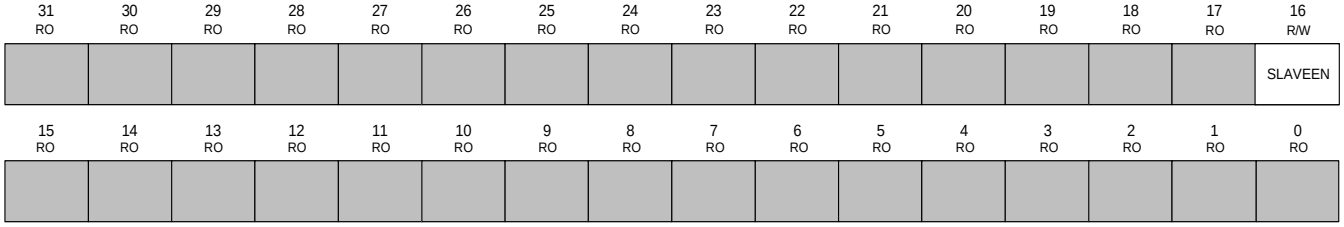
I2S base address:

- SPI0: 0x4800_C000
- SPI1: 0x4800_C100

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	I2S_CTRL0	R/W	0x0000_0007	I2S Control Register 0	437
0x0008	I2S _ENABLE	R/W	0x0000_0000	I2S Enable Register	437
0x0010	I2S_SLV_ENB	R/W	0x0000_0000	I2S Slave Enable Register	438
0x0014	I2S_BR_SL	R/W	0x0000_0000	I2S Baud Rate Select	439
0x0018	I2S_TXFF_TH	R/W	0x0000_0000	I2S Transmit FIFO Threshold Level	440
0x001C	I2S_RXFF_TH	R/W	0x0000_0000	I2S Receive FIFO Threshold Level	441
0x0020	I2S_TXFF_LV	RO	0x0000_0000	I2S Transmit FIFO Level Register	442
0x0024	I2S_RXFF_LV	RO	0x0000_0000	I2S Receive FIFO Level Register	442
0x0028	I2S_STAS	RO	0x0000_0006	I2S Status Register	443
0x002C	I2S_IER	WO	0x0000_0000	I2S Interrupt Enable Register	444
0x0030	I2S_IDR	WO	0x0000_0000	I2S Interrupt Disable Register	445
0x0034	I2S_IMR	RO	0x0000_0000	I2S Interrupt Mask Status Register	446
0x0038	I2S_RIS	RO	0x0000_0000	I2S Raw Interrupt Status Register	447
0x003C	I2S_ISC	R/W1C	0x0000_0000	I2S Interrupt Status and Clear Register	448
0x0044	I2S_CTRL1	R/W	0x0000_0000	I2S Control Register1	449
0x004C	I2S_DMA_CTRL	R/W	0x0000_0000	I2S DMA Control Register	450
0x0050	I2S_DMA_TDLR	R/W	0x0000_0000	I2S DMA Transmit Threshold Register	450
0x0054	I2S_DMA_RDLR	R/W	0x0000_0000	I2S DMA Receive Threshold Register	451
0x0060	I2S_DATA	R/W	0x0000_0000	I2S Data Register	451

4.17.3.1 I2S_CTRL0 - I2S CONTROL REGISTER 0

This register controls the serial data transfer.

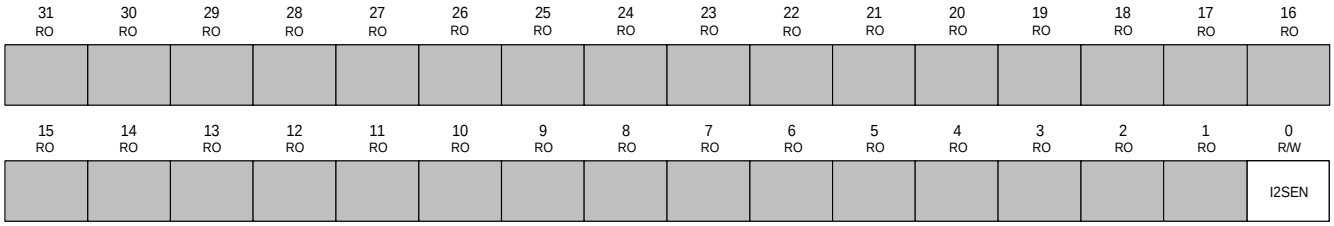


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:17	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
16	*SLAVEEN*	R/W	0	Slave Mode Enable 0: Master Mode 1: Slave Mode
15:0	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.

4.17.3.2 I2S_ENABLE - I2S ENABLE REGISTER

Enables and Disables all I2S operations.

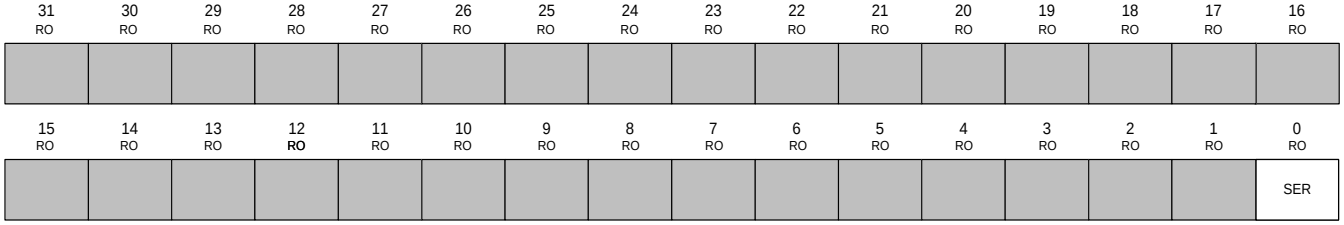


Offset: 0x0008

Bits	Name	Type	Reset	Description
31:1	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*I2SEN*	R/W	0	I2S Enable Enables and disables all I2S operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. It is impossible to program some of the I2S control registers when enabled. 0: Disable I2S 1: Enable I2S

4.17.3.3 I2S_SLV_ENB - I2S SLAVE ENABLE REGISTER

This register is valid only when the I2S is configured as a master device.

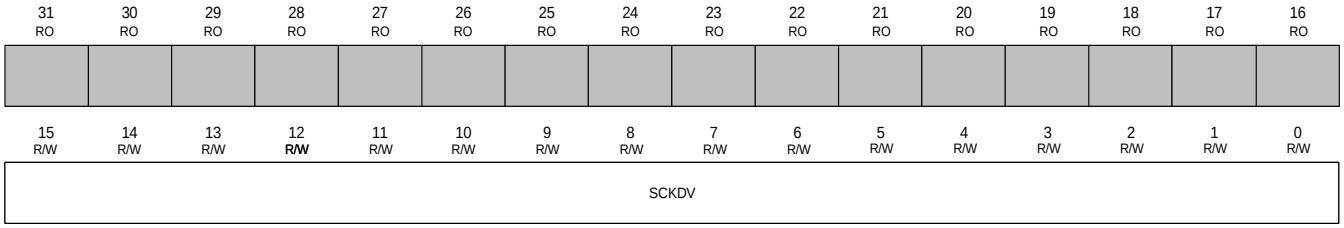


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:1	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
0	*SER*	RO	1	Slave Select Enable Flag When the bit is set, the corresponding slave select line from the master is activated when a serial transfer begins. It should be noted that setting or clearing bits in this register have no effect on the corresponding slave select outputs until a transfer is started. Before beginning a transfer, you should enable the bit in this register that corresponds to the slave device with which the master wants to communicate. 0: Slave is not selected 1: Slave is selected.

4.17.3.4 I2S_BR_SL - I2S BAUD RATE SELECT

This register is valid only when the I2S is configured as a master device. This register derives the frequency of the serial clock that regulates the data transfer.

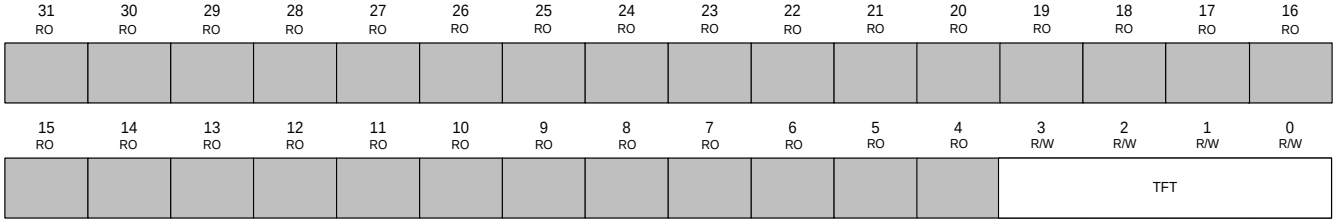


Offset: 0x0000

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15:0	*SCKDV*	R/W	0x0	I2S Clock Divider The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is below 3, the serial output clock (I2S_BCLK) is disabled. The frequency of the I2S_BCLK is derived from the following equation: When Master clock is enabled (*MCLKEN* bit set): $F_{MCLK}=256 * F_{I2S_BCLK}$ $F_{ws} = F_{I2S_Clock} / [(2 * SCKDV) * (16 * 2) * 8]$ when the channel frame is 16-bit wide $F_{ws} = F_{I2S_Clock} / [(2 * SCKDV) * (32 * 2) * 4]$ when the channel frame is 32-bit wide When Master clock is disable (*MCLKEN* bit cleared): $F_{ws} = F_{I2S_Clock} / [(2 * SCKDV) * (16 * 2)]$ when the channel frame is 16-bit wide $F_{ws} = F_{I2S_Clock} / [(2 * SCKDV) * (32 * 2)]$ when the channel frame is 32-bit wide

4.17.3.5 I2S_TXFF_TH - I2S TRANSMIT FIFO THRESHOLD LEVEL

This register controls the threshold value for the transmit FIFO memory.

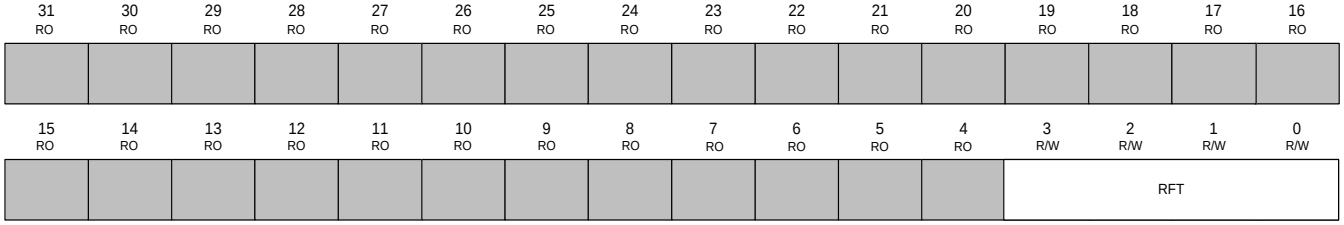


Offset: 0x0018

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:0	*TFT*	R/W	0x0	Transmit FIFO Threshold Controls the level of entries (or below) at which the transmit FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2-8; this register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered. 1000: Interrupt is asserted when 8 data entries are present in transmit FIFO. 000: Interrupt is asserted when 0 data entries are present in transmit FIFO. 001: Interrupt is asserted when 1 data entries are present in transmit FIFO. 010: Interrupt is asserted when 2 data entries are present in transmit FIFO. 011: Interrupt is asserted when 3 data entries are present in transmit FIFO. 100: Interrupt is asserted when 4 data entries are present in transmit FIFO. 101: Interrupt is asserted when 5 data entries are present in transmit FIFO. 110: Interrupt is asserted when 6 data entries are present in transmit FIFO. 111: Interrupt is asserted when 7 data entries are present in transmit FIFO. 1111: Interrupt is asserted when 15 data entries are present in transmit FIFO.

4.17.3.6 I2S_RXFF_TH - I2S RECEIVE FIFO THRESHOLD LEVEL

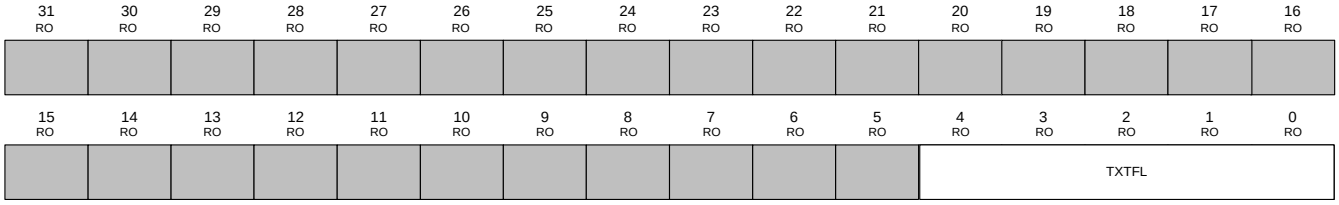
The register controls the threshold value for the receive FIFO memory



Offset: 0x001C

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:0	*RFT*	R/W	0x0	Receive FIFO Threshold Controls the level of entries (or above) at which the receive FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2-8. This registers sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered. 1000: Interrupt is asserted when 9 or more data entry is present in receive FIFO. 000: Interrupt is asserted when 1 or more data entry is present in receive FIFO. 001: Interrupt is asserted when 2 or more data entry is present in receive FIFO. 010: Interrupt is asserted when 3 or more data entry is present in receive FIFO. 011: Interrupt is asserted when 4 or more data entry is present in receive FIFO. 100: Interrupt is asserted when 5 or more data entry is present in receive FIFO. 101: Interrupt is asserted when 6 or more data entry is present in receive FIFO. 110: Interrupt is asserted when 7 or more data entry is present in receive FIFO. 111: Interrupt is asserted when 8 or more data entry is present in receive FIFO. 1111: Interrupt is asserted when 16 or more data entry is present in receive FIFO.

4.17.3.7 I2S_TXFF_LV - I2S TRANSMIT FIFO LEVEL REGISTER

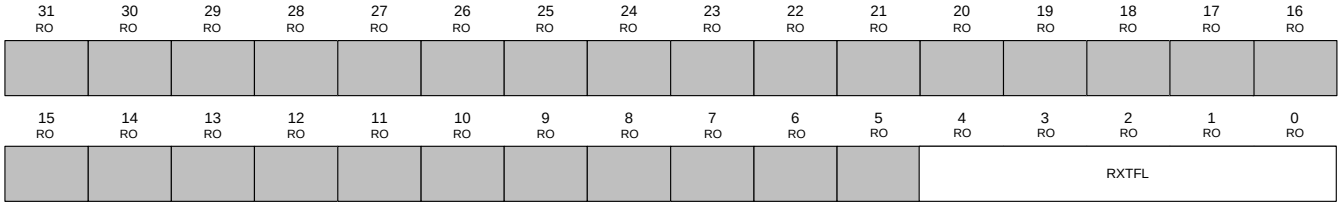


Offset: 0x0020

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4:0	*TXTFL*	RO	0x0	Transmit FIFO Level Contains the number of valid data entries in the transmit FIFO.

4.17.3.8 I2S_RXFF_LV - I2S RECEIVE FIFO LEVEL REGISTER

This register contains the number of valid data entries in the transmit FIFO memory.

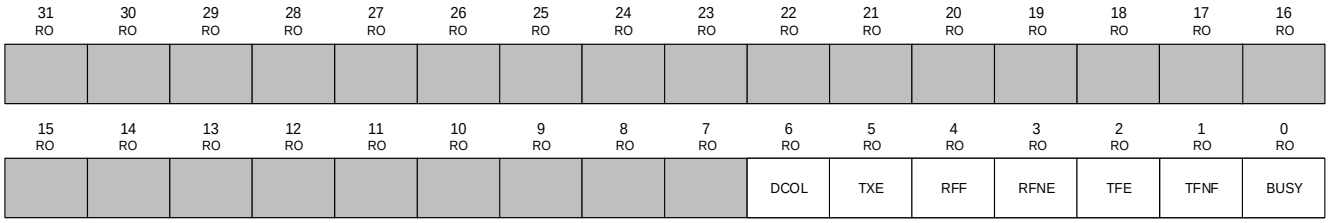


Offset: 0x0024

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4:0	*RXTFL*	RO	0x0	Receive FIFO Level Contains the number of valid data entries in the receive FIFO.

4.17.3.9 I2S_STAS - I2S STATUS REGISTER

This is a read-only register indicating the current transfer status, FIFO status, and any transmission/reception errors that may have occurred

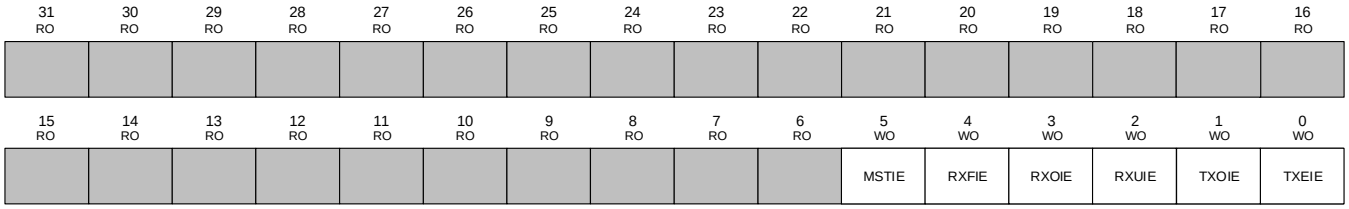


Offset: 0x0028

Bits	Name	Type	Reset	Description
31:7	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6	*DCOL*	RO	0	Data Collision Error Relevant only when the I2S is configured as a master device. This bit is set if the I2S master is actively transmitting when another master selects this device as a slave. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read. 0: No error. 1: Transmit data collision error.
5	*TXE*	RO	0	Transmission Error Set if the transmit FIFO is empty when a transfer is started. This bit can be set only when the I2S is configured as a slave device. Data from the previous transmission is resent on the txd line. This bit is cleared when read. 0: No error. 1: Transmission error.
4	*RFF*	RO	0	Receive FIFO Full When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0: Receive FIFO is not full. 1: Receive FIFO is full.
3	*RFNE*	RO	0	Receive FIFO Not Empty Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. 0: Receive FIFO is empty. 1: Receive FIFO is not empty
2	*TFE*	RO	1	Transmit FIFO Empty When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0: Transmit FIFO is not empty. 1: Transmit FIFO is empty
1	*TFNF*	RO	1	Transmit FIFO Not Full Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0: Transmit FIFO is full. 1: Transmit FIFO is not full

Bits	Name	Type	Reset	Description
0	*BUSY*	RO	0	I2S Busy Flag. When set, indicates that a serial transfer is in progress; when cleared indicates that the I2S is idle or disabled. 0: I2S is idle or disabled. 1: I2S is actively transferring data.

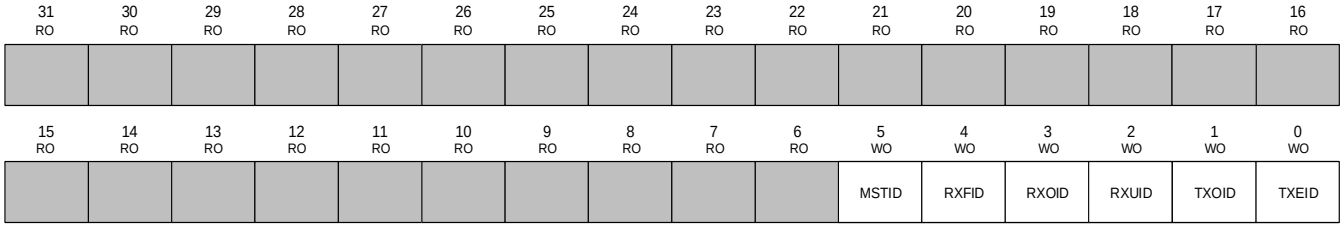
4.17.3.10 I2S_IER - I2S INTERRUPT ENABLE REGISTER



Offset: 0x002C

Bits	Name	Type	Reset	Description
31:6	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTIE*	WO	0	Multi-Master Contention Interrupt Enable 1: Interrupt is enabled.
4	*RXFIE*	WO	0	Receive FIFO Full Interrupt Enable 1: Interrupt is enabled.
3	*RXOIE*	WO	0	Receive FIFO Overflow Interrupt Enable 1: Interrupt is enabled.
2	*RXUIE*	WO	0	Receive FIFO Underflow Interrupt Enable 1: Interrupt is enabled.
1	*TXOIE*	WO	0	Transmit FIFO Overflow Interrupt Enable 1: Interrupt is enabled.
0	*TXEIE*	WO	0	Transmit FIFO Empty Interrupt Enable 1: Interrupt is enabled.

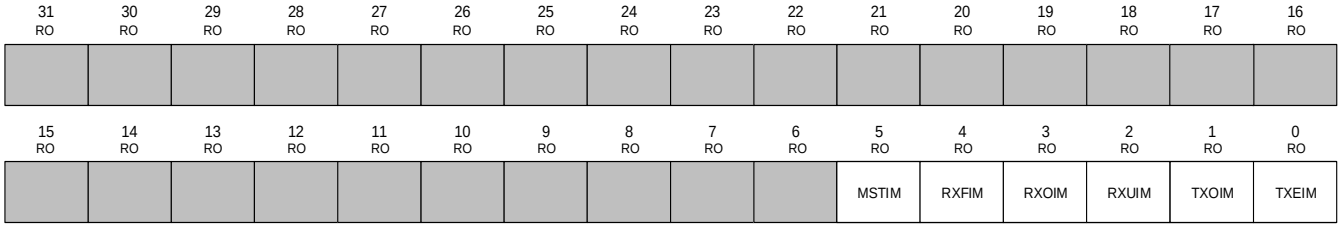
4.17.3.11 I2S_IDR - I2S INTERRUPT DISABLE REGISTER



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:6	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTID*	WO	0	Multi-Master Contention Interrupt Disable 1: Interrupt is disabled.
4	*RXFID*	WO	0	Receive FIFO Full Interrupt Disable 1: Interrupt is disabled.
3	*RXOID*	WO	0	Receive FIFO Overflow Interrupt Disable 1: Interrupt is disabled.
2	*RXUID*	WO	0	Receive FIFO Underflow Interrupt Disable 1: Interrupt is disabled.
1	*TXOID*	WO	0	Transmit FIFO Overflow Interrupt Disable 1: Interrupt is disabled.
0	*TXEID*	WO	0	Transmit FIFO Empty Interrupt Disable 1: Interrupt is disabled.

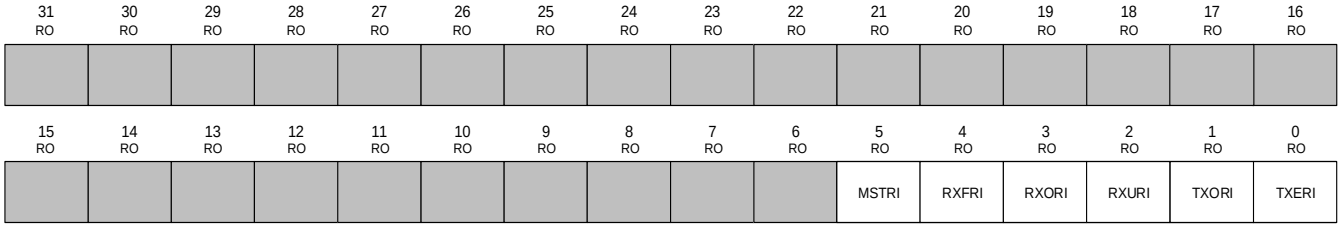
4.17.3.12 I2S_IMR - I2S INTERRUPT MASK STATUS REGISTER



Offset: 0x0034

Bits	Name	Type	Reset	Description
31:6	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTIM*	RO	0	Multi-Master Contention Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
4	*RXFIM*	RO	0	Receive FIFO Full Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
3	*RXOIM*	RO	0	Receive FIFO Overflow Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
2	*RXUIM*	RO	0	Receive FIFO Underflow Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
1	*TXOIM*	RO	0	Transmit FIFO Overflow Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
0	*TXEIM*	RO	0	Transmit FIFO Empty Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked

4.17.3.13 I2S_RIS - I2S RAW INTERRUPT STATUS REGISTER

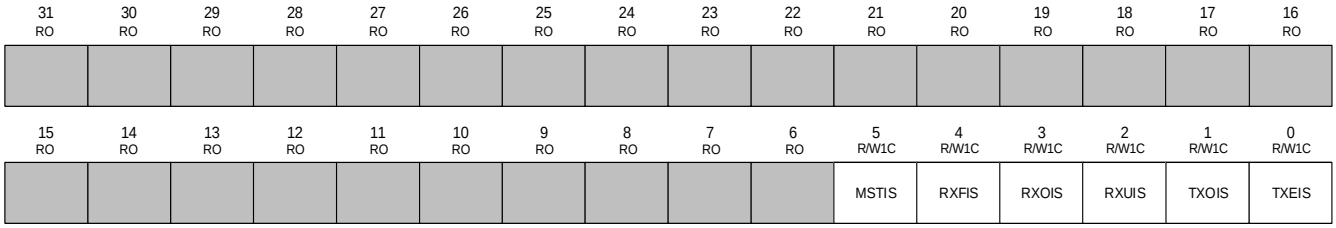


Offset: 0x0038

Bits	Name	Type	Reset	Description
31:6	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTRI*	RO	0	Multi-Master Contention Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
4	*RXFRI*	RO	0	Receive FIFO Full Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
3	*RXORI*	RO	0	Receive FIFO Overflow Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
2	*RXURI*	RO	0	Receive FIFO Underflow Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
1	*TXORI*	RO	0	Transmit FIFO Overflow Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
0	*TXERI*	RO	0	Transmit FIFO Empty Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.17.3.14 I2S_ISC - I2S INTERRUPT STATUS AND CLEAR REGISTER

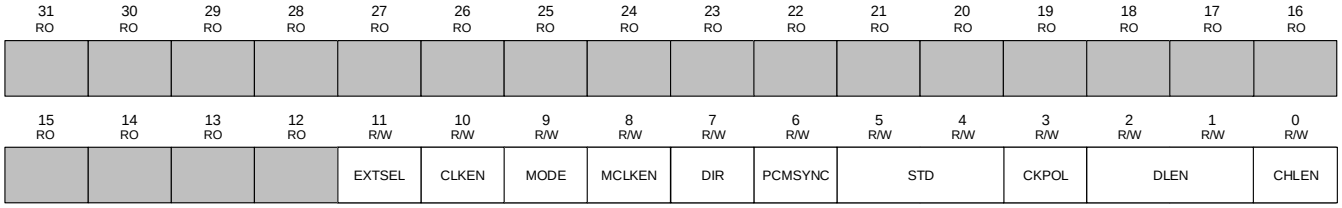
Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.



Offset: 0x003C

Bits	Name	Type	Reset	Description
31:6	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
5	*MSTIS*	R/W1C	0	Multi-Master Contention Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
4	*RXFIS*	R/W1C	0	Receive FIFO Full Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
3	*RXOIS*	R/W1C	0	Receive FIFO Overflow Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
2	*RXUIS*	R/W1C	0	Receive FIFO Underflow Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
1	*TXOIS*	R/W1C	0	Transmit FIFO Overflow Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
0	*TXEIS*	R/W1C	0	Transmit FIFO Empty Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.

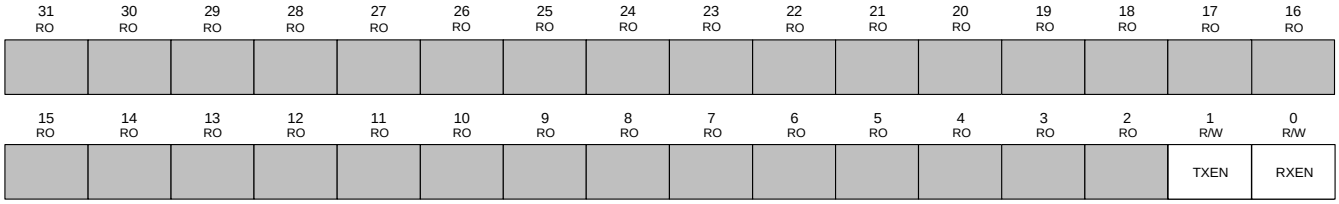
4.17.3.15 I2S_CTRL1 - I2S CONTROL REGISTER 1



Offset: 0x0044

Bits	Name	Type	Reset	Description
31:12	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
11	*EXTSEL*	R/W	0	External Clock Source Select 0: I2S0_CLK is selected. 1: I2S1_CLK is selected.
10	*CLKEN*	R/W	0	I2S Clock Enable 0: APB clock is used as source 1: External clock is used as source.
9	*MODE*	R/W	0	I2S Mode Select 0: SPI mode is selected. 1: I2S mode is selected
8	*MCLKEN*	R/W	0	I2S Master Clock Enable When Master clock is enable (*MCLKEN* bit in the I2S_CTRL2 register is set): $F_{MCLK}=256 * F_{I2S_BCLK}$ 1: Enable I2S master clock.
7	*DIR*	R/W	0	I2S Direction Select 0: Transmit 1: Receive
6	*PCMSYNC*	R/W	0	I2S PCM Frame Synchronization 0: Short Frame Synchronization 1: Long Frame Synchronization
5:4	*STD*	R/W	0x0	I2S Standard Select 00: I2S Philips standard 01: MSB justified standard 10: LSB justified standard 11: PCM standard
3	*CKPOL*	R/W	0	I2S Steady State Clock Polarity 0: I2S clock steady state is low level 1: I2S clock steady state is high level
2:1	*DLEN*	R/W	0x0	I2S Data Length 00: 16-bit data length 01: 24-bit data length 10: 32-bit data length 11: Reserved
0	*CHLEN*	R/W	0	I2S Channel Length 0: 16-bit wide 1: 32-bit wide

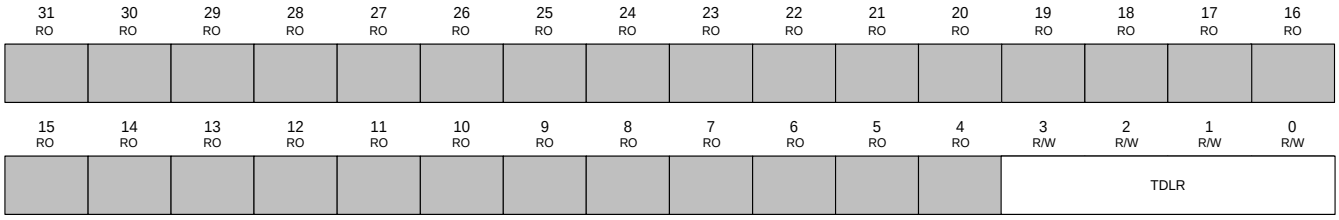
4.17.3.16 I2S_DMA_CTRL - I2S DMA CONTROL REGISTER



Offset: 0x004C

Bits	Name	Type	Reset	Description
31:2	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	*TXEN*	R/W	0	DMA Transmit Enable 1: Enable DMA Tx
0	*RXEN*	R/W	0	DMA Receive Enable 1: Enable DMA Rx

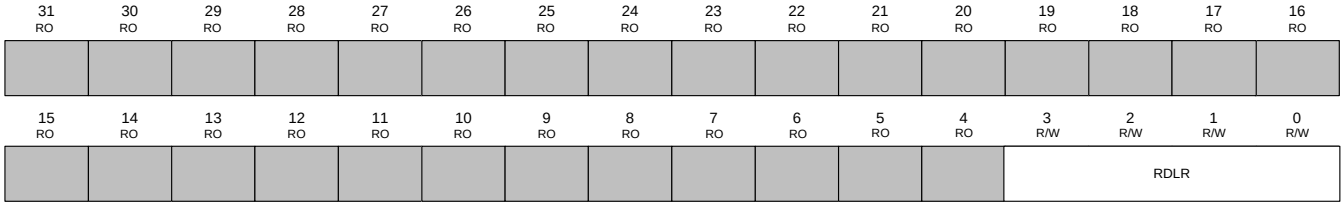
4.17.3.17 I2S_DMA_TDLR - I2S DMA TRANSMIT THRESHOLD REGISTER



Offset: 0x0050

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:0	*TDLR*	R/W	0x0	DMA Transmit Data Level Controls the level of entries (or below) at which the transmit FIFO controller triggers a DMA transmit request. When the number of transmit FIFO entries is less than or equal to this value, the DMA transmit request is triggered.

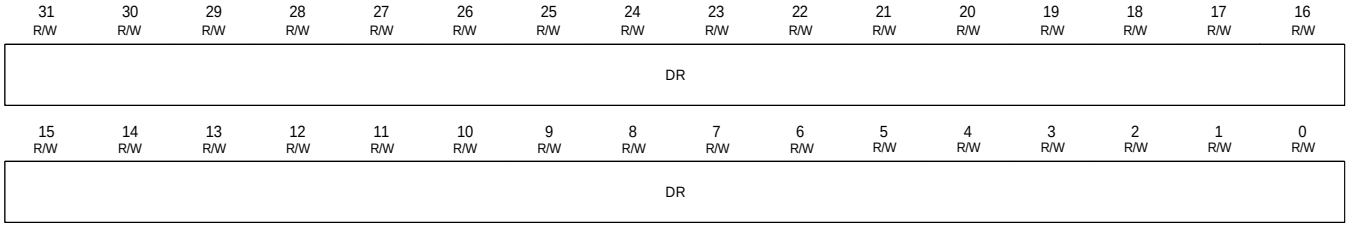
4.17.3.18 I2S_DMA_RDLR - I2S DMA RECEIVE THRESHOLD REGISTER



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:0	RDLR	R/W	0x0	DMA Receive Data Level Controls the level of entries (or above) at which the receive FIFO controller triggers a DMA receive request. When the number of receive FIFO entries is greater than or equal to this value + 1, the DMA receive request is triggered

4.17.3.19 I2S_DATA - I2S DATA REGISTER



Offset: 0x0060

Bits	Name	Type	Reset	Description
31:0	DR	R/W	0x0	Data Register When writing to this register, you must right-justify the data, while reading data are automatically right-justified. Read: Receive FIFO buffer. Write: Transmit FIFO buffer.

4.18 EXTERNAL BUS INTERFACE (EBI)

The Inter-IC Sound (I2S) protocol is a synchronous serial communication interface which can operate in slave or master

The External Bus Interface (EBI) module provides a convenient, high-speed way to interface external parallel memory device to PT32C family device.

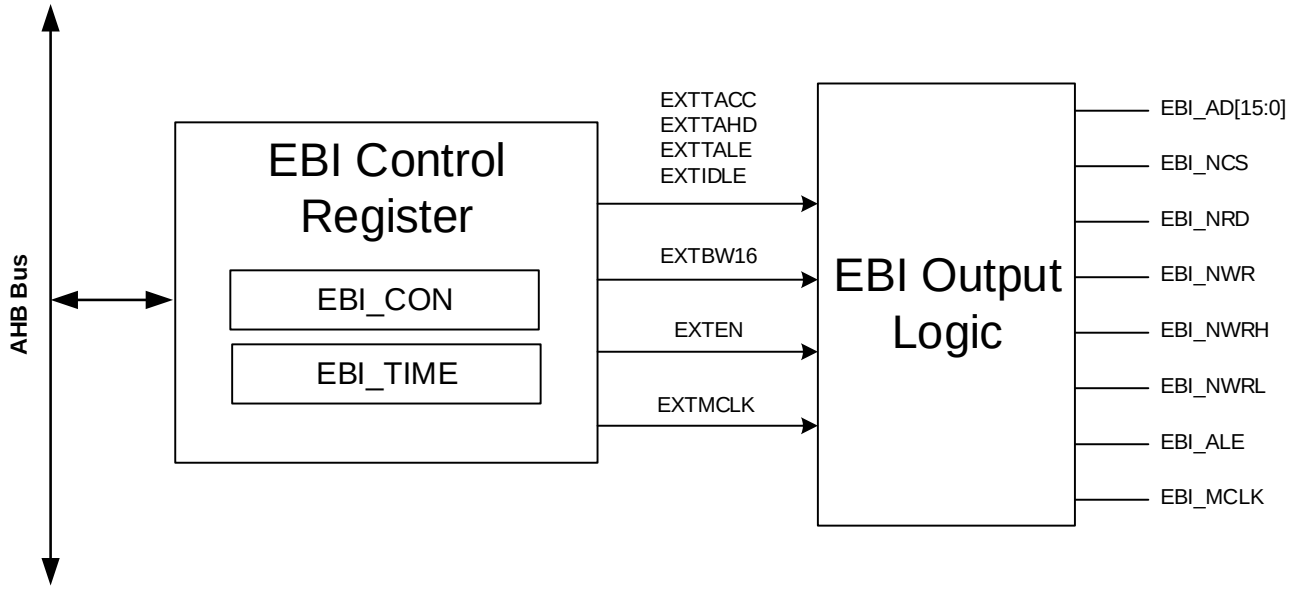
To save the connections between external device and this chip, EBI support address bus and data bus multiplex mode. And, address latch enable (ALE) signal supported differentiate the address and data cycle.

The EBI can interface external SRAM, SDRAM, and peripherals, such as LCD displays and other memory mapped devices. The address space for the external memory is selectable from 64 KB (8-bit) up to 128kB (16-bit).

EBI Main Features:

- Address bus and data bus multiplex mode supported to save the address pins
- Variable data access time (tACC), address latch enable time (tALE) and address hold time (tAHD)
- Variable external bus base clock (MCLK) supported.
- 8 bit or 16 bit data width supported.

Figure 4.18-1: EBI Block Diagram



4.18.1 FUNCTION DESCRIPTION

4.18.1.1 EBI DATA WIDTH CONNECTION

PT32C302 EBI support device whose address bus and data bus are multiplexed. For external device with separated address and data bus, the connection to device needs additional logic to latch the address. In this case, pin EBI_ALE is connected to the latch device. For 16-bit device, EBI_AD [15:0] are shared by address and 16-bit data. For 8-bit device, only EBI_AD [7:0] are share by address and 8-bit data, EBI_AD [15:8] are dedicated for address and could be connected to 8-bit device directly.

Figure 4.18-2: Connection of 16-bit EBI Data Width with 16-bit Device

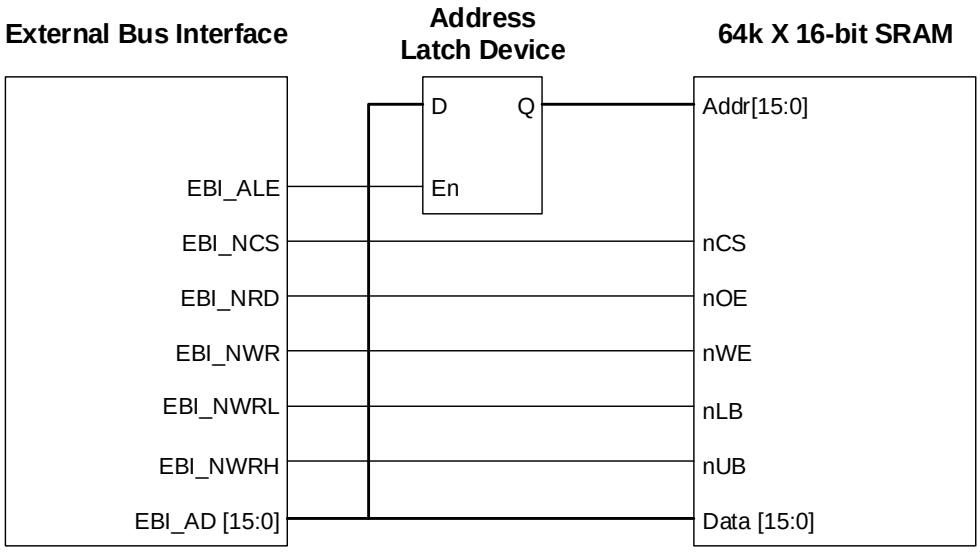
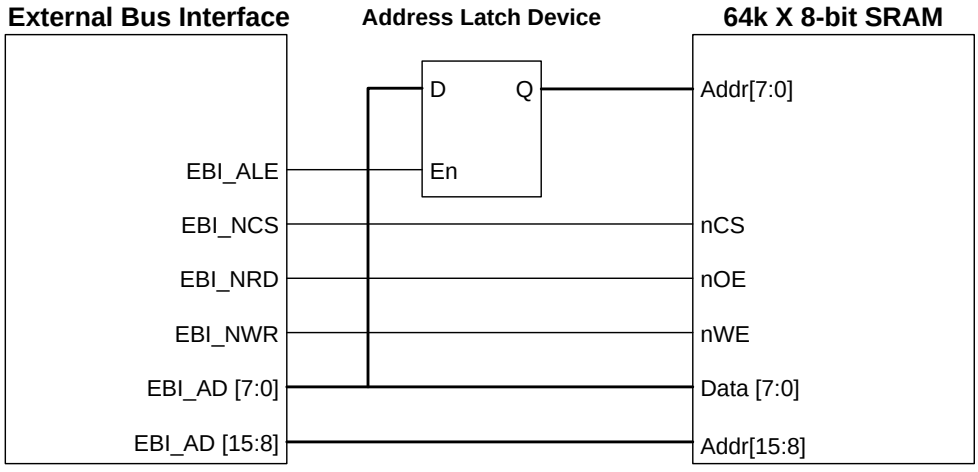


Figure 4.18-3: Connection of 8-bit EBI Data Width with 8-bit Device



When the system access data width is larger than EBI data width (8-bit/16-bit width), the EBI controller will finish a system access command by operating EBI access more than once. For instance, if system request a 32-bit data through EBI device, the EBI controller will operate accessing four times when setting EBI data width with 8-bit data width.

4.18.1.2 EBI DATA WIDTH CONNECTION

MCLK CONTROL

All EBI signals will be synchronized by EBI_MCLK when EBI is operating. When the chip connects to the external device with slower operating frequency the EBI_MCLK can divide most to 32 from HCLK by setting EBI_CON.MCLKDIV. Therefore, the EBI controller is suitable for a wide frequency range of EBI device. If EBI_MCLK frequency is set to the same as HCLK, EBI signals are synchronized by positive edge of EBI_MCLK, else by negative edge of EBI_MCLK.

OPERATION AND ACCESS TIMING CONTROL

At the start of EBI access, chip select signal (EBI_NCS) asserts to low and waits one EBI_MCLK for address setup time (tASU) for address stable. Then address latch enable signal (EBI_ALE) asserts to high after address is stable and keeps for a period of time (tALE) for address latch. After latch address, EBI_ALE signal asserts to low and waits one EBI_MCLK for address latch hold time (tLHD) and another one EBI_MCLK cycle (tA2D) that is inserted behind tLHD to be the bus turn-around time for address change to data. Then EBI_NRD signal asserts to low when read access or EBI_NWR signal asserts to low when write access. Then EBI_NRD or EBI_NWR signal asserts to high after keeps access time (tACC) for reading output stable or writing finish. After that, EBI signals keep for data access hold time (tAHD) and EBI_NCS signal asserts to high then address is released by current access control.

The EBI controller provides a flexible timing control for different external devices, as shown in the following table. In EBI timing control, tASU, tLHD and tA2D are all fixed to 1 MCLK cycle, tAHD can modulate to 1 ~ 8 MCLK cycles by setting *EXTTAHD* or zero tAHD for read/write by setting *RAHDOFF* / *WAHDOFF* bit, tACC can modulate to 1 ~ 32 MCLK cycles by setting *EXTTACC*, and tALE can modulate to 1 - 8 MCLK cycles by setting *EXTTALE*.

Table 4.18-1: EBI Timing Table

Parameter	Value	Unit	Description
tASU	0 ~ 1	MCLK	Address Latch Setup Time, controlled by *TASUOFF* of **EMI_TIME**. Write 1 to *TASUOFF* to select zero tASU cycle, write 0 to select one tASU cycle.
tALE	1 ~ 8	MCLK	ALE High Period. controlled by *EXTTALE* of **EBI_TIME**
tLHD	0 ~ 1	MCLK	Address Latch Hold Time, controlled by *TLHDOFF* of **EMI_TIME.** Write 1 to *TLHDOFF* to select zero tLHD cycle, write 0 to select one tLHD cycle.
tA2D	0 ~ 1	MCLK	Address To Data Delay (Bus Turn-Around Time), controlled by *WA2DOFF* and *RA2DOFF* of **EMI_TIME.** Write 1 to *WA2DOFF* and *RA2DOFF* to select zero tA2D cycle, write 0 to *WA2DOFF* and *RA2DOFF* to select one tA2D cycle.
tACC	1 ~ 32	MCLK	Data Access Time. Controlled by *EXTTACC* of **EBI_TIME**
tAHD	0 ~ 8	MCLK	Data Access Hold Time. Controlled by *EXTTAHD* of **EXI_TIME** to select 1 ~8 tAHD. Setting RAHDOFF / WAHDOFF to select zero tAHD
IDLE	0 ~ 16	MCLK	Controlled by *EXTIDLE* of **EXI_TIME** to select 1~16 Idle Cycle. Write 1 to *RAIDOFF* and *WAIDOFF* to select zero IDLE cycle.

Figure 4.18-4: EBI Timing Control Waveform for 16-bit Data Width

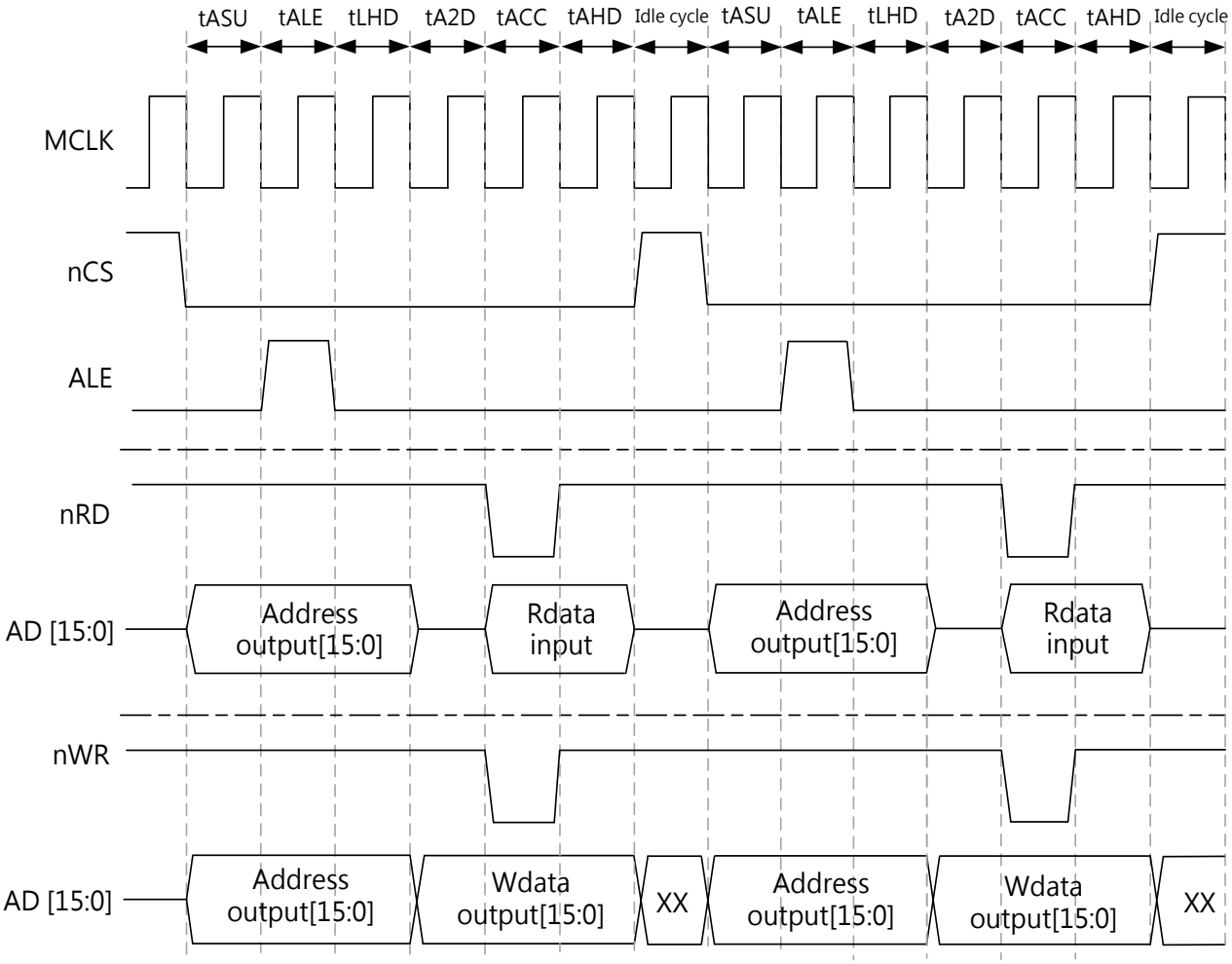
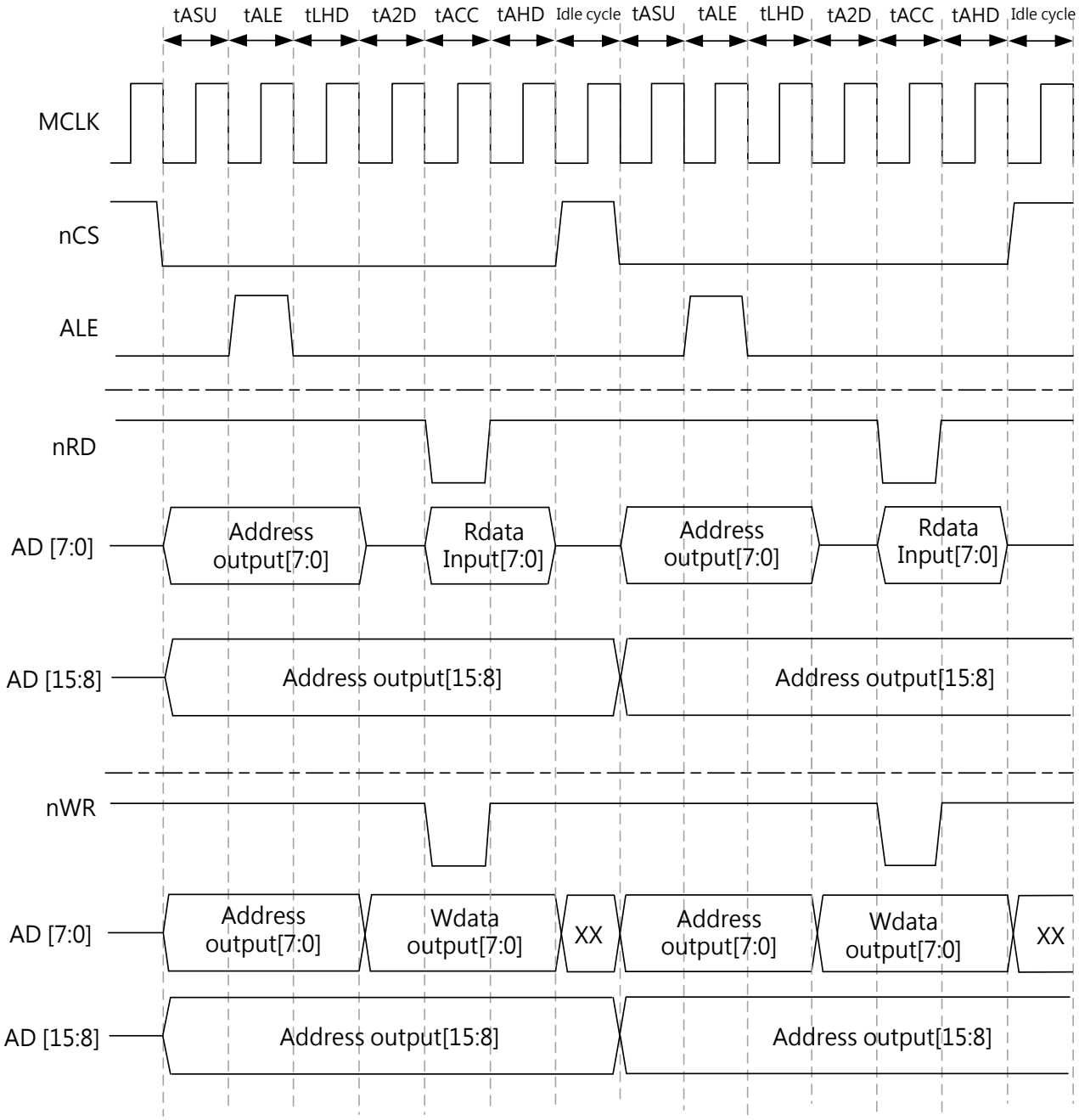


Figure 4.18-5: EBI Timing Control Waveform for 8-bit Data Width



Above figure show the timing diagram when inserts idle cycle. When EBI is accessing continuously, bus conflict may occur if the device access time is much longer compared with system clock frequency. The EBI controller supplies additional idle cycle to solve this problem. During idle cycle period, all control signals of EBI bus are inactive.

Figure 4.18-6: EBI Timing Control Waveform for 16-bit Data Width without ASU/LHD/A2D

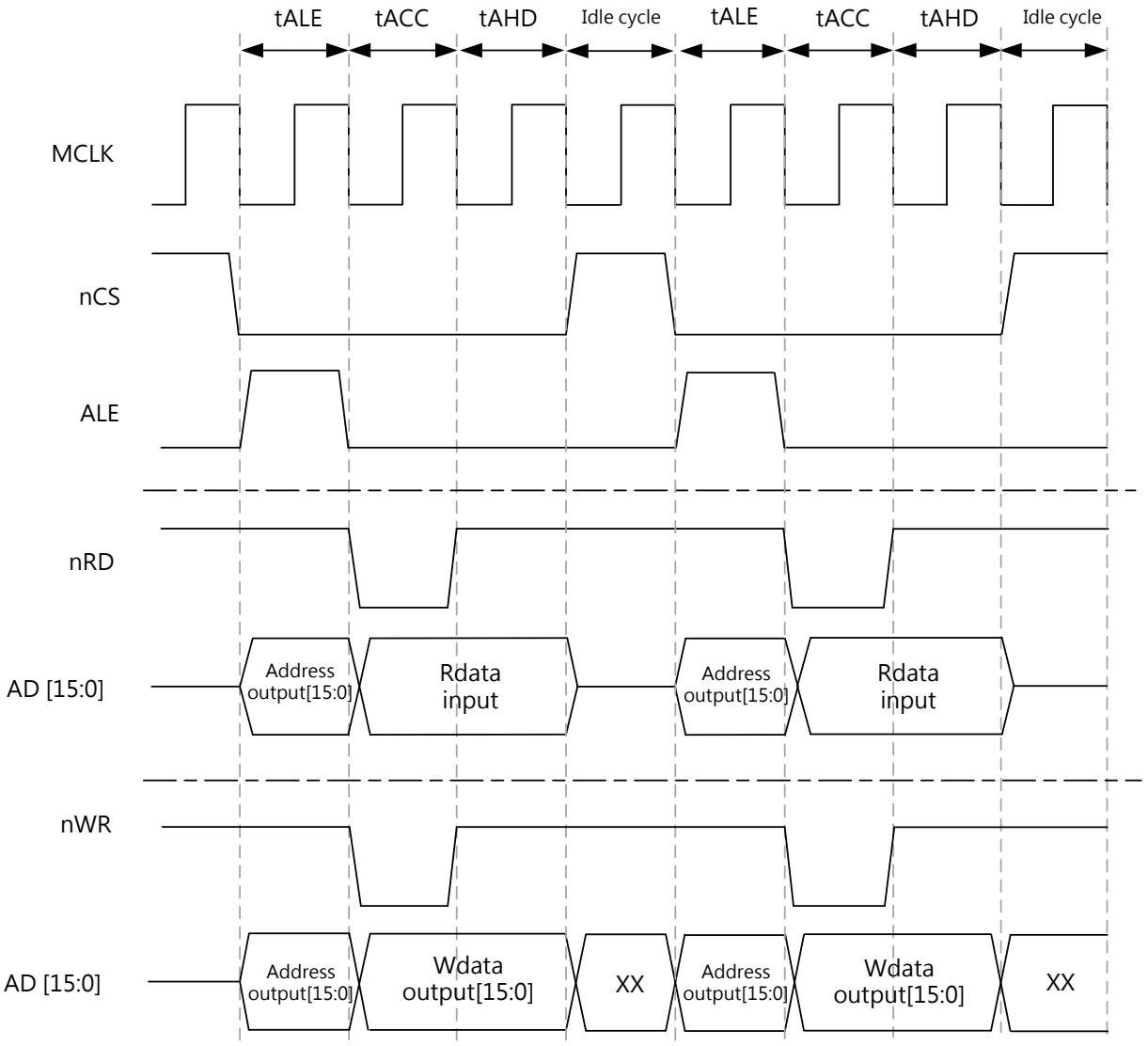


Figure 4.18-7: EBI Timing Control Waveform for 8-bit Data Width without ASU/LHD/A2D

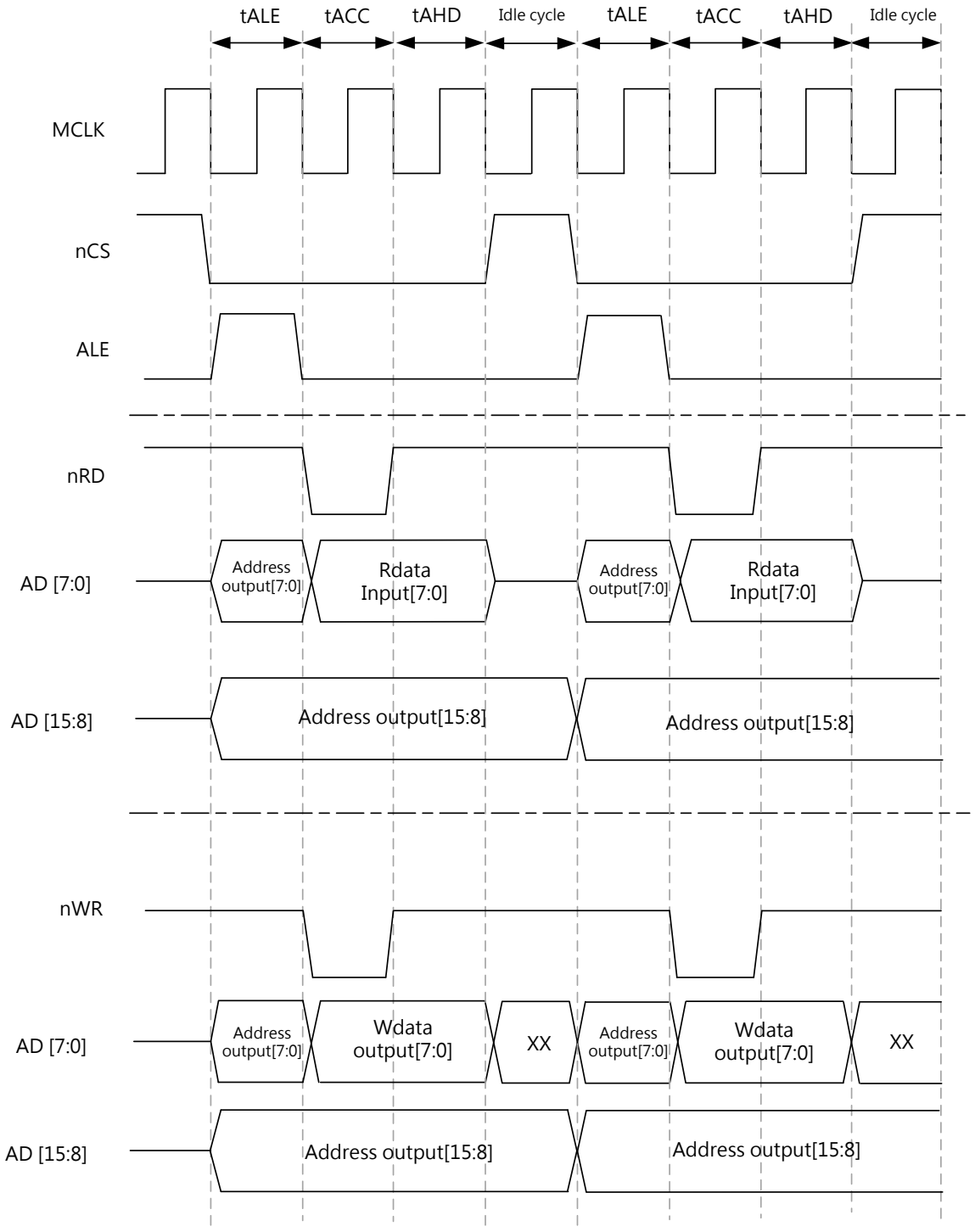


Figure 4.18-6 and Figure 4.18-7 shows an example of setting 16-bit / 8-bit data width for EBI application without inserting tASU, tLHD and tA2D.

Figure 4.18-8: EBI Timing Control Waveform without ASU/LDH/A2D/IDLE (16-bit)

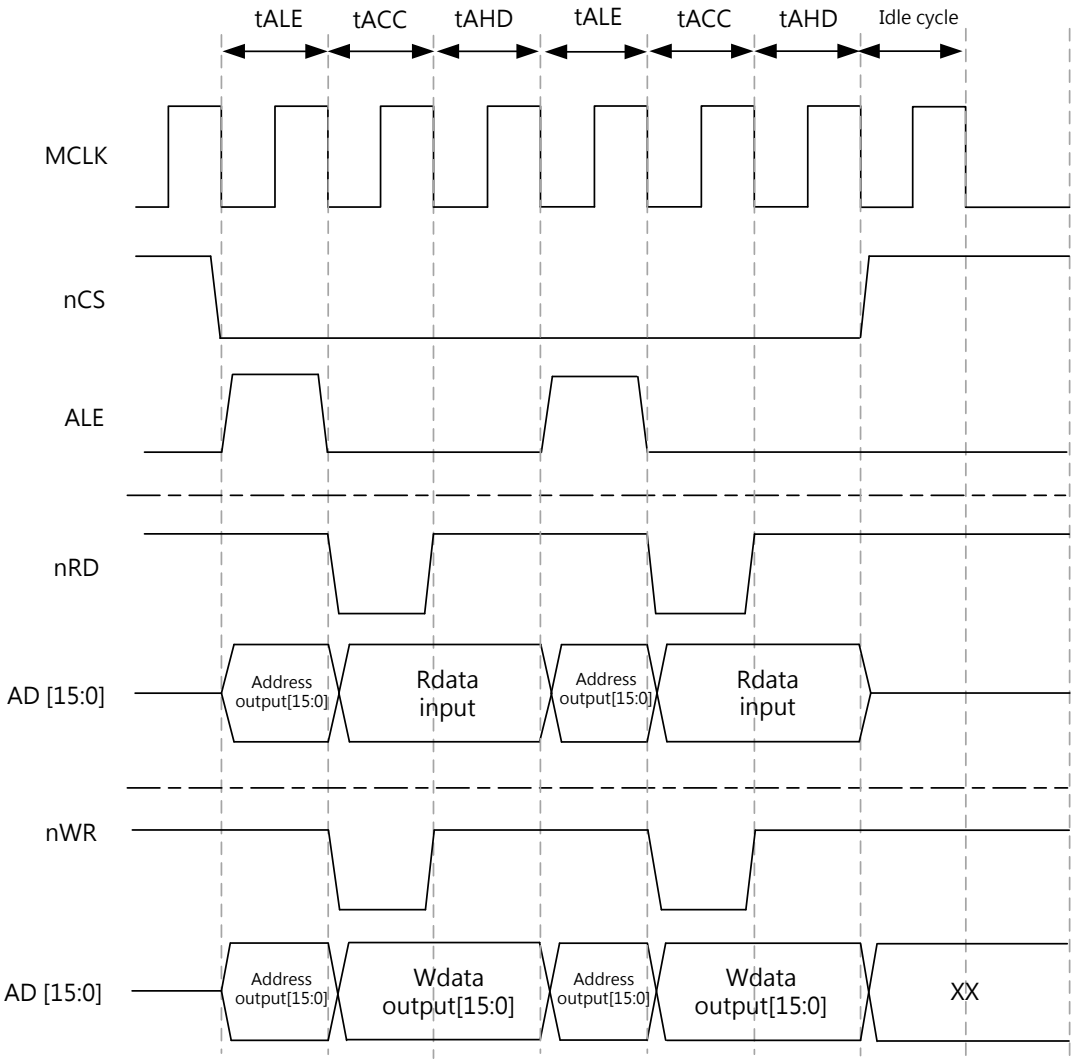


Figure 4.18-9: EBI Timing Control Waveform without ASU/LDH/A2D/IDLE (8-bit)

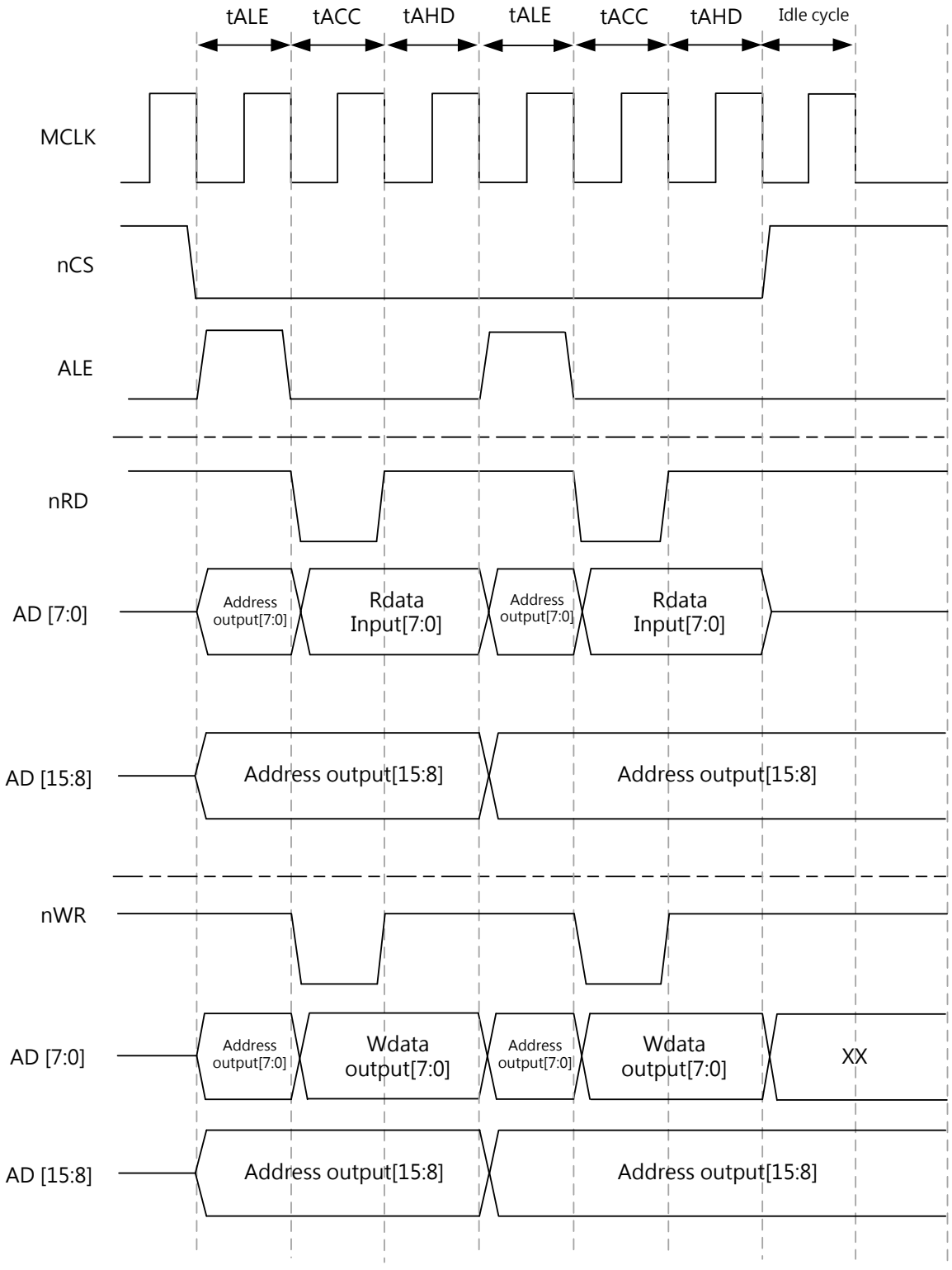


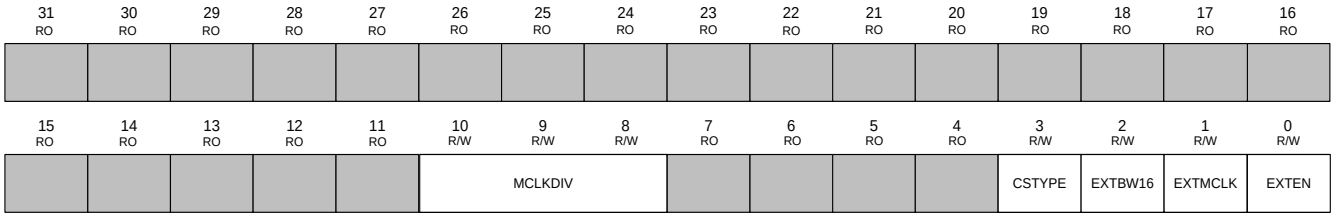
Figure 4.18-8 and **Figure 4.18-9** shows an example of setting 16-bit / 8-bit data width for EBI application without inserting tASU, tLHD, tA2D and IDLE.

4.18.2 EBI REGISTER MAP

Base Address: 0x5006_0000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	EBI_CON	R/W	0x0000_0000	External Bus Interface Control Register	461
0x0004	EBI_TIME	R/W	0x0000_0000	External Bus Interface Timing Control Register	462

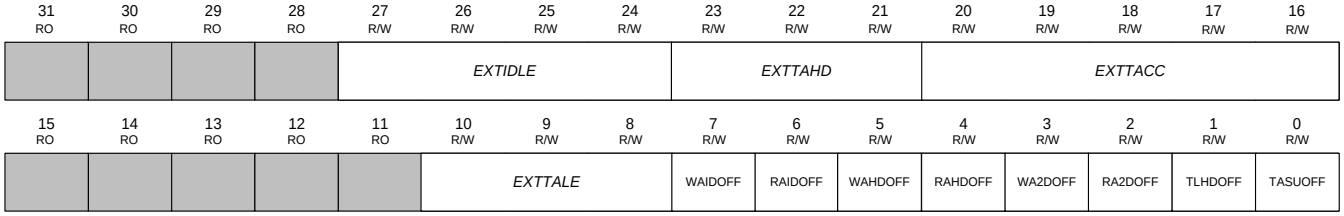
4.18.2.1 EBI_CON - EXTERNAL BUS INTERFACE GENERAL CONTROL REGISTER



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:11	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10:8	*MCLKDIV*	R/W	0x0	External Output Clock Divider The frequency of EBI output clock (MCLK) is controlled by MCLKDIV as follows: 000: MCLK frequency is HCLK/1. 001: MCLK frequency is HCLK/2 010: MCLK frequency is HCLK/4 011: MCLK frequency is HCLK/8 100: MCLK frequency is HCLK/16 101: MCLK frequency is HCLK/32
7:4	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*CSTYPE*	R/W	0x0	EBI Chip Select Type 0: EBI Chip Select Active Low. 1: EBI Chip Select Active High.
2	*EXTBW16*	R/W	0x0	EBI Data Width 16-bit / 8-bit This bit defines if the data bus is 8-bit or 16-bit. 0: EBI data width is 8 bit. 1: EBI data width is 16 bit.
1	*EXTMCLK*	R/W	0x0	EBI External MCLK Enable 0: Disable External MCLK 1: Enable External MCLK
0	*EXTEN*	R/W	0x0	EBI Enable Control This bit is the functional enable bit for EBI. 0: EBI function Disabled. 1: EBI function Enabled.

4.18.2.2 EBI_TIME - EXTERNAL BUS INTERFACE TIMING CONTROL REGISTER



Offset: 0x0004

Bits	Name	Type	Reset	Description
31:28	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
27:24	*EXTIDLE*	R/W	0x0	Idle State Cycle Idle state cycle = (ExtIdle+1) * MCLK.
23:21	*EXTTAHD*	R/W	0x0	EBI Data Access Hold Time ExttAHD defines data access hold time (tAHD). tAHD = (ExttAHD +1) * MCLK.
20:16	*EXTTACC*	R/W	0x0	EBI Data Access Time ExttACC defines data access time (tACC). tACC = (ExttACC +1) * MCLK.
15:11	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
10:8	*EXTTALE*	R/W	0x0	Expand Time of ALE This field is used to control the ALE pulse width (tALE) for address latch. tALE = (ExttALE + 1) * MCLK.
7	*WAID_OFF*	R/W	0x0	Access IDLE Time Disable Control When Write
6	*RAID_OFF*	R/W	0x0	Access IDLE Time Disable Control When Read
5	*WAHD_OFF*	R/W	0x0	Access Hold Time Disable Control When Write 0 = tAHD is controlled by ExttAHD[2:0] when write through EBI. 1 = Zero tAHD when write through EBI.
4	*RAHD_OFF*	R/W	0x0	Access Hold Time Disable Control When Read 0 = tAHD is controlled by ExttAHD[2:0] when read through EBI. 1 = Zero tAHD when read through EBI.
3	*WA2D_OFF*	R/W	0x0	Access TA2D Time Disable Control When Write
2	*RA2D_OFF*	R/W	0x0	Access TA2D Time Disable Control When Read
1	*TLHD_OFF*	R/W	0x0	Access TLHD Time Disable Control
0	*TASU_OFF*	R/W	0x0	Access TASU Time Disable Control

4.19 DIRECT MEMORY ACCESS CONTROLLER (DMA)

The Direct Memory Access Controller contains both a Direct Memory Access engine and a Cyclic Redundancy Check (CRC). The DMA can transfer data between memories and peripheral via six channels of DMA (Peripheral-to-Memory or Memory-to-Peripheral or Memory-to-Memory).

For each DMA channel - CH [5:0], there is one buffer as transfer buffer between the Peripherals APB devices and Memory. The CPU can recognize the completion of a DMA operation by software polling or when it receives an internal DMA interrupt.

DMA MAIN FEATURES:

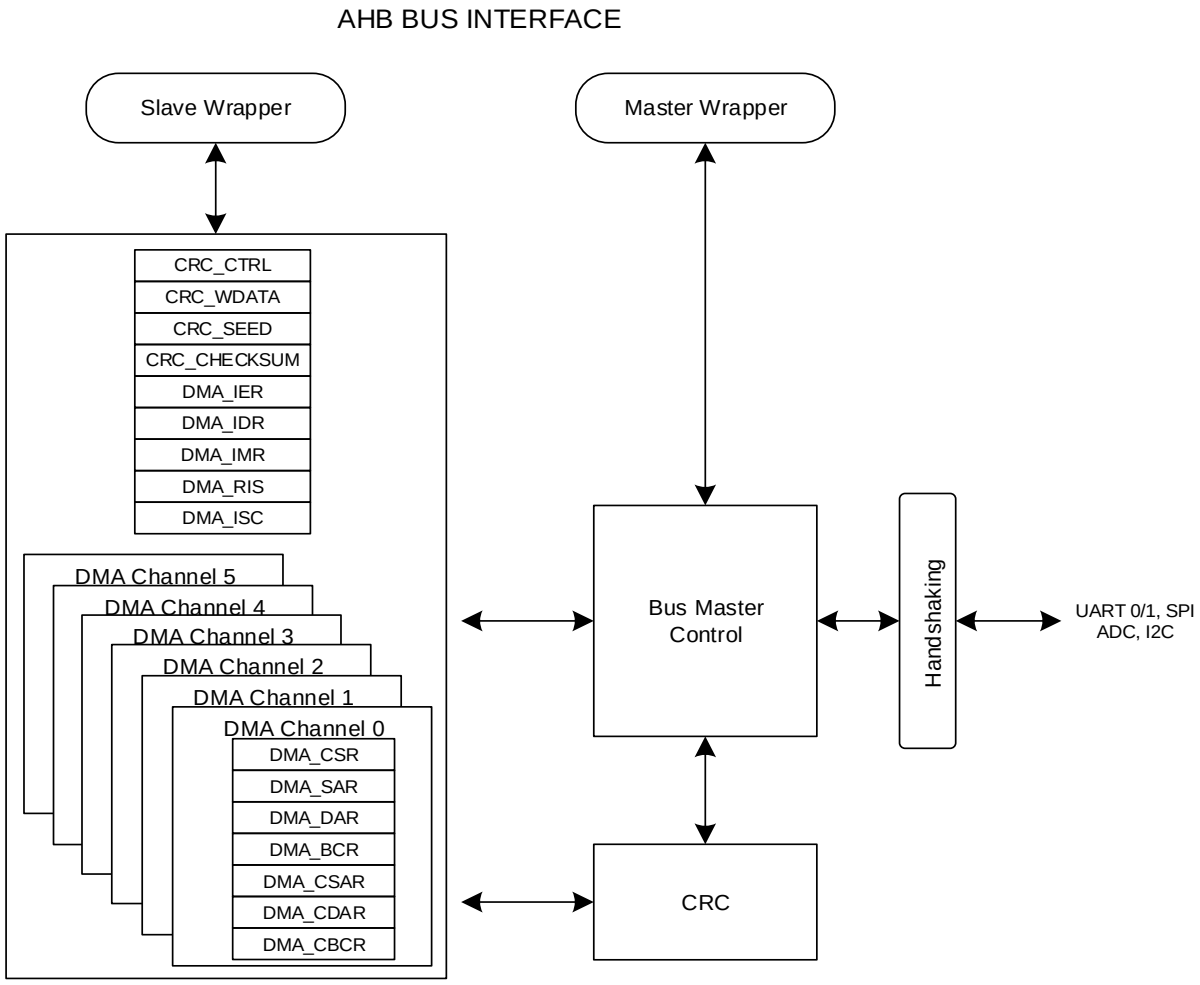
- Support six DMA channels. Each channel can support a unidirectional transfer.
- AMBA AHB master/slave interface compatible, for data transfer and register read/write.
- Support source and destination address increased mode or decreased mode or fixed mode.
- Hardware channel priority. DMA channel 0 has the highest priority and channel 5 has the lowest priority.
- Software channel priority. Set the software channel priority, when the two DMA channel was set the same priority, it will be performed according to the hardware channel priority order.

CRC MAIN FEATURES:

- Supports four common polynomials CRC-CCITT, CRC-8, CRC-16 and CRC-32.
  - CRC-CCITT: $X_{16} + X_{12} + X_5 + 1$.
  - CRC-8: $X_8 + X_2 + X + 1$.
  - CRC-16: $X_{16} + X_{15} + X_2 + 1$.
  - CRC-32: $X_{32} + X_{26} + X_{23} + X_{22} + X_{16} + X_{12} + X_{11} + X_{10} + X_8 + X_7 + X_5 + X_4 + X_2 + X + 1$.
- Programmable seed value.
- Supports programmable order reverse setting for input data and CRC checksum
- Supports programmable 1's complement setting for input data and CRC checksum
- Supports CPU PIO mode or DMA transfer mode.
- Supports 8/16/32-bit of data width in CPU PIO mode.
  - 8-bit write mode: 1-AHB clock cycle operation.
  - 16-bit write mode: 2-AHB clock cycle operation.
  - 32-bit write mode: 4-AHB clock cycle operation.

4.19.1 BLOCK DIAGRAM

Figure 4.19-1: DMA Block Diagram



4.19.2 FUNCTIONAL DESCRIPTION

4.19.2.1 DMA

The DMA controller contains six channels DMA (Peripheral-to-Memory or Memory-to-Peripheral or Memory-to-Memory). The CPU can recognize the completion of a DMA operation by software polling or when it receives an internal DMA interrupt. As to the source and destination address, the DMA controller has three different modes: increased, decreased and fixed operation mode. Software must write a valid source address to the DMA_SARx register, a destination address to the DMA_DARx register, and a transfer count to the DMA_BCRx register, and then enable DMA channel DMA_CSRx [DMACEN]. The DMA will continue the transfer until DMA_CBCRx comes down to zero, if an error occurs during the DMA operation, the channel stops and generate an internal DMA interrupt.

In DMA (Peripheral-to-Memory or Memory-to-Peripheral) mode, DMA can transfer data between the Peripherals APB IP (ex: UART, SPI, ADC and I2C) and Memory.

4.19.2.2 CRC

The DMA controller contains a cyclic redundancy check (CRC) generator that can perform CRC calculation with programmable polynomial settings. The operation polynomial includes CRCCCITT, CRC-8, CRC-16 and CRC-32; Software can choose the operation polynomial mode by setting MODE fields in CRC_CTRL register. The CRC engine support CPU PIO mode and DMA transfer mode (CRC_CTRL.CRCEN = 1, CRC_CTRL.WKMODE = '01'or'10'). When operating in CPU Programmed input/output (PIO) mode:

Enable CRC engine by setting CRCEN bit in CRC_CTRL register. When user want to initialize the DMA , they should setting the data format (WDRVS, CSRVS, WDCOM and CSCOM by setting CRC_CTRL register), seed value (CRC_SEED register) and select the data length by setting CRC_CTRL. CPUWDL. Setting CRC reset to load the initial seed value to CRC circuit by setting RST bit in CRC_CTRL register. Write data to CRC_WDATA register to perform CRC calculation. Get the CRC checksum result by reading CRC_CHECKSUM register.

When operating in CRC DMA read only mode:

Enable CRC engine by setting CRCEN bit in CRC_CTRL register. When user want to initialize the DMA, they should setting the data format (WDRVS, CSRVS, WDCOM and CSCOM by setting CRC_CTRL register), seed value (CRC_SEED register) and give a valid of source address and transfer count by setting DMA_SARx and DMA_BCRx. Enable DMA_CSRx.DMACEN and then hardware will reset the seed value and read memory data to perform CRC calculation. Wait CRC DMA transfer and CRC calculation done, then get the CRC checksum result by reading CRC_CHECKSUM register.

When operating in CRC DMA normal mode:

Enable CRC engine by setting CRCEN bit in CRC_CTL register. When user want to initialize the DMA, they should setting the data format (WDRVS, CSRVS, WDCOM and CSCOM by setting CRC_CTL register), seed value (CRC_SEED register) and give a valid of source address, destination address and transfer count by setting DMA_SARx, DMA_DARx and DMA_BCRx. Enable DMA_CSRx.DMACEN and then hardware will reset the seed value and read memory data to perform CRC calculation. Wait CRC DMA transfer and CRC calculation done, then get the CRC checksum result by reading CRC_CHECKSUM register.

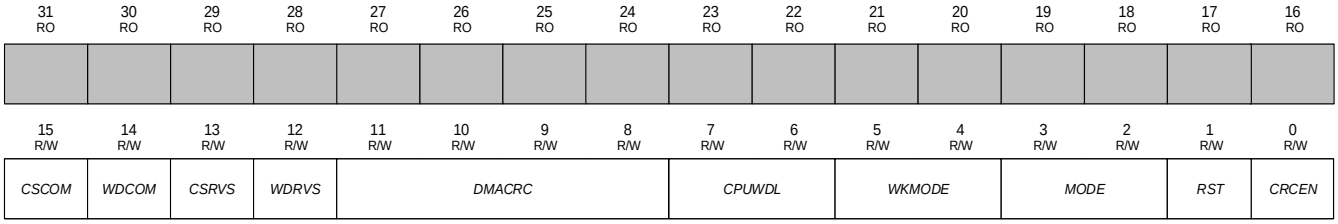
4.19.3 DMA REGISTER MAP

Base Address: 0x5003_0000

Offset	Symbol	Type	Reset Value	Description	See page
Global Control					
0x1000	CRC_CTRL	R/W	0x0000_0000	CRC Control Register	468
0x1004	CRC_WDATA	R/W	0x0000_0000	CRC Write Data Register	469
0x1008	CRC_SEED	R/W	0x0000_0000	CRC Seed Register	470
0x100C	CRC_CHECKSUM	RO	0x0000_0000	CRC Checksum Register	470
0x1010	DMA_IER	WO	0x0000_0000	DMA Interrupt Enable Register	471
0x1014	DMA_IDR	WO	0x0000_0000	DMA Interrupt Disable Register	472
0x1018	DMA_IMR	RO	0x0000_0000	DMA Interrupt Mask Register	473
0x101C	DMA_RIS	RO	0x0000_0000	DMA Raw Interrupt Status Register	474
0x1020	DMA_ISC	R/W	0x0000_0000	DMA Interrupt Status And Clear Register	476
DMA Channel [5:0] Control					
0x0000	DMA_CSR0	R/W	0x0000_0000	DMA Channel 0 Control and Status Register	477
0x0004	DMA_SAR0	R/W	0x0000_0000	DMA Channel 0 Source Address Register	479
0x0008	DMA_DAR0	R/W	0x0000_0000	DMA Channel 0 Destination Address Register	480
0x000C	DMA_BCR0	R/W	0x0000_0000	DMA Channel 0 Transfer Byte Count Register	480
0x0010	DMA_CSAR0	RO	0x0000_0000	DMA Channel 0 Current Source Address Register	481
0x0014	DMA_CDAR0	RO	0x0000_0000	DMA Channel 0 Current Destination Address Register	481
0x0018	DMA_CBCR0	RO	0x0000_0000	DMA Channel 0 Current Transfer Byte Count Register	482
0x0100	DMA_CSR1	R/W	0x0000_0000	DMA Channel 1 Control and Status Register	477
0x0104	DMA_SAR1	R/W	0x0000_0000	DMA Channel 1 Source Address Register	479
0x0108	DMA_DAR1	R/W	0x0000_0000	DMA Channel 1 Destination Address Register	480
0x010C	DMA_BCR1	R/W	0x0000_0000	DMA Channel 1 Transfer Byte Count Register	480
0x0110	DMA_CSAR1	RO	0x0000_0000	DMA Channel 1 Current Source Address Register	481
0x0114	DMA_CDAR1	RO	0x0000_0000	DMA Channel 1 Current Destination Address Register	481
0x0118	DMA_CBCR1	RO	0x0000_0000	DMA Channel 1 Current Transfer Byte Count Register	482
0x0200	DMA_CSR2	R/W	0x0000_0000	DMA Channel 2 Control and Status Register	477
0x0204	DMA_SAR2	R/W	0x0000_0000	DMA Channel 2 Source Address Register	479
0x0208	DMA_DAR2	R/W	0x0000_0000	DMA Channel 2 Destination Address Register	480
0x020C	DMA_BCR2	R/W	0x0000_0000	DMA Channel 2 Transfer Byte Count Register	480
0x0210	DMA_CSAR2	RO	0x0000_0000	DMA Channel 2 Current Source Address Register	481
0x0214	DMA_CDAR2	RO	0x0000_0000	DMA Channel 2 Current Destination Address Register	481
0x0218	DMA_CBCR2	RO	0x0000_0000	DMA Channel 2 Current Transfer Byte Count Register	482
0x0300	DMA_CSR3	R/W	0x0000_0000	DMA Channel 3 Control and Status Register	477
0x0304	DMA_SAR3	R/W	0x0000_0000	DMA Channel 3 Source Address Register	479
0x0308	DMA_DAR3	R/W	0x0000_0000	DMA Channel 3 Destination Address Register	480

Offset	Symbol	Type	Reset Value	Description	See page
Global Control					
0x030C	DMA_BCR3	R/W	0x0000_0000	DMA Channel 3 Transfer Byte Count Register	480
0x0310	DMA_CSAR3	RO	0x0000_0000	DMA Channel 3 Current Source Address Register	481
0x0314	DMA_CDAR3	RO	0x0000_0000	DMA Channel 3 Current Destination Address Register	481
0x0318	DMA_CBCR3	RO	0x0000_0000	DMA Channel 3 Current Transfer Byte Count Register	482
0x0400	DMA_CSR4	R/W	0x0000_0000	DMA Channel 4 Control and Status Register	477
0x0404	DMA_SAR4	R/W	0x0000_0000	DMA Channel 4 Source Address Register	479
0x0408	DMA_DAR4	R/W	0x0000_0000	DMA Channel 4 Destination Address Register	480
0x040C	DMA_BCR4	R/W	0x0000_0000	DMA Channel 4 Transfer Byte Count Register	480
0x0410	DMA_CSAR4	RO	0x0000_0000	DMA Channel 4 Current Source Address Register	481
0x0414	DMA_CDAR4	RO	0x0000_0000	DMA Channel 4 Current Destination Address Register	481
0x0418	DMA_CBCR4	RO	0x0000_0000	DMA Channel 4 Current Transfer Byte Count Register	482
0x0500	DMA_CSR5	R/W	0x0000_0000	DMA Channel 5 Control and Status Register	477
0x0504	DMA_SAR5	R/W	0x0000_0000	DMA Channel 5 Source Address Register	479
0x0508	DMA_DAR5	R/W	0x0000_0000	DMA Channel 5 Destination Address Register	480
0x050C	DMA_BCR5	R/W	0x0000_0000	DMA Channel 5 Transfer Byte Count Register	480
0x0510	DMA_CSAR5	RO	0x0000_0000	DMA Channel 5 Current Source Address Register	481
0x0514	DMA_CDAR5	RO	0x0000_0000	DMA Channel 5 Current Destination Address Register	481
0x0518	DMA_CBCR5	RO	0x0000_0000	DMA Channel 5 Current Transfer Byte Count Register	482

4.19.3.1 CRC_CTRL - CRC CONTROL REGISTER

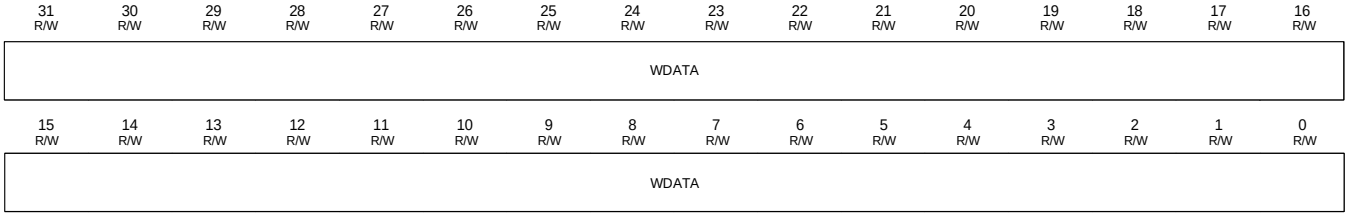


Offset: 0x1000

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
15	*CSCOM*	R/W	0	Checksum Complement. 0: No complement for CRC checksum. 1: 1's complement for CRC checksum.
14	*WDCOM*	R/W	0	Write Data Complement 0: No complement for CRC write data in. 1: 1's complement for CRC write data in.
13	*CSRVS*	R/W	0	Checksum Reverse 0: No bit order reverse for CRC checksum. 1: Bit order reverse for CRC checksum. If the checksum data is 0xDD7B0F2E, the bit order reverse for CRC checksum is 0x74F0DEBB.
12	*WDRVS*	R/W	0	Write Data Order Reverse 0: No bit order reverse for CRC write data in. 1: Bit order reverse for CRC write data in (per byte). If the write data is 0xAABBCCDD, the bit order reverse for CRC write data in is 0x55DD33BB.
11:8	*DMACRC*	R/W	0x0	DMA CRC Channel Select. 0x0: Channel 0 is selected. 0x1: Channel 1 is selected. 0x2: Channel 2 is selected. 0x3: Channel 3 is selected. 0x4: Channel 4 is selected. 0x5: Channel 5 is selected.
7:6	*CPUWDL*	R/W	0x0	CPU Write Data Length When operating in CPU PIO mode, this field indicates the write data length. 00: The data length is 8-bit mode. 01: The data length is 16-bit mode. 10: The data length is 32-bit mode. 11 : Reserved
5:4	*WKMODE*	R/W	0x0	CRC Work Mode. 00: CPU PIO Mode 01: DMA Read Only Mode. In this mode, DMA only read source address data, but not write data to the destination address. 10: DMA Normal Mode. In this mode, DMA will read source address data, and write the data to the destination address. 11: Reserved.

Bits	Name	Type	Reset	Description
3:2	*MODE*	R/W	0x0	CRC Polynomial Mode. 00: CRC-CCITT Polynomial Mode. 01: CRC-8 Polynomial Mode. 10: CRC-16 Polynomial Mode. 11: CRC-32 Polynomial Mode.
1	*RST*	R/W	0	CRC Engine Reset. 0: Writing 0 to this bit has no effect. 1: Writing 1 to this bit will reset the internal CRC state machine and internal buffer. The contents of control register will not be cleared. This bit will be auto cleared after few clock cycles. When operating in CPU PIO mode, setting this bit will reload the initial seed value.
0	*CRCEN*	R/W	0	CRC Channel Enable. 1: Enable CRC's operation.

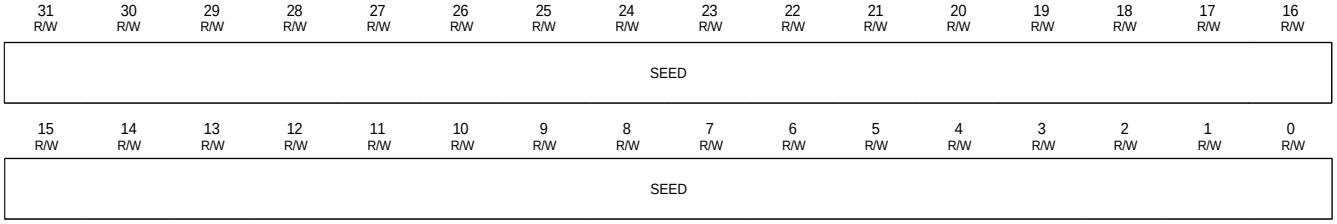
4.19.3.2 CRC_WDATA - CRC WRITE DATA REGISTER



Offset: 0x1004

Bits	Name	Type	Reset	Description
31:0	*WDATA*	R/W	0x0	CRC Write Data Register When operating in CPU PIO mode, software can write data to this field to perform CRC operation. When operating in CRC DMA mode, this field has no effect. When operating in CRC DMA mode, so don't filled any data in this field. The **CRC_CTRL**.*WDCOM* and **CRC_CTRL**.*WDRVS* bit setting will affected this field. For example, if *WDRVS* = 1, if the write data in **CRC_WDATA** register is 0xAABBCCDD, the read data from CRC_WDATA register will be 0x55DD33BB.

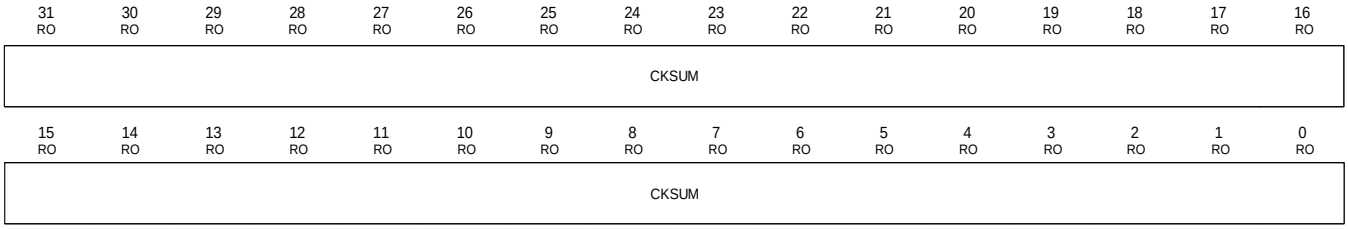
4.19.3.3 CRC_SEED - CRC SEED REGISTER



Offset: 0x1008

Bits	Name	Type	Reset	Description
31:0	*SEED*	R/W	0x0	CRC Send Register This field indicates the CRC seed value.

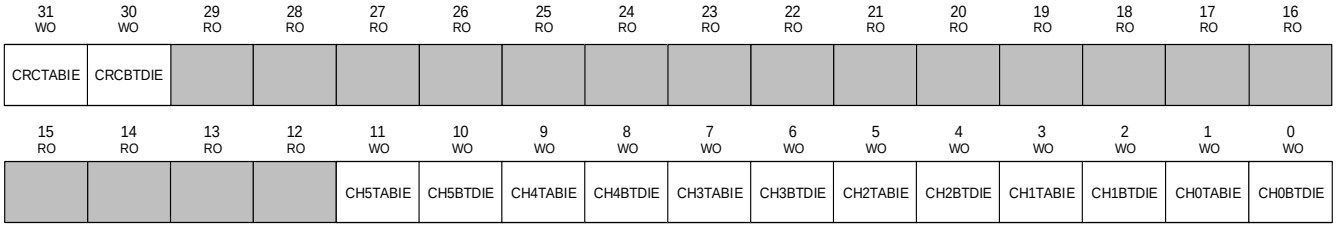
4.19.3.4 CRC_CHECKSUM - CRC CHECKSUM REGISTER



Offset: 0x100C

Bits	Name	Type	Reset	Description
31:0	*CKSUM*	RO	0x0	CRC Checksum Register. This field indicates the CRC checksum.

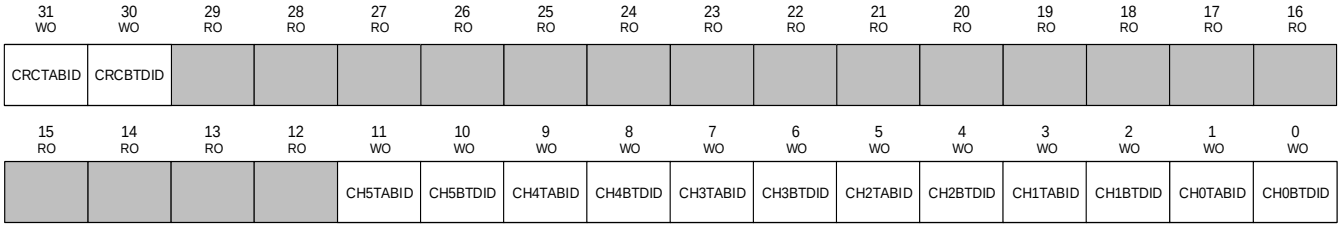
4.19.3.5 DMA_IER - INTERRUPT ENABLE/DISABLE



Offset: 0x1010

Bits	Name	Type	Reset	Description
31	*CRCTABIE*	WO	0	CRC DMA TABORT Interrupt Enable 1: Interrupt is enable
30	*CRCBTDIE*	WO	0	CRC DMA Block Transmit Done Interrupt Enable 1: Interrupt is enable
29:12	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
11	*CH5TABIE*	WO	0	DMA Channel 5 TABORT Interrupt Enable 1: Interrupt is enable
10	*CH5BTDIE*	WO	0	DMA Channel 5 Block Transmit Done Interrupt Enable 1: Interrupt is enable
9	*CH4TABIE*	WO	0	DMA Channel 4 TABORT Interrupt Enable 1: Interrupt is enable
8	*CH4BTDIE*	WO	0	DMA Channel 4 Block Transmit Done Interrupt Enable 1: Interrupt is enable
7	*CH3TABIE*	WO	0	DMA Channel 3 TABORT Interrupt Enable 1: Interrupt is enable
6	*CH3BTDIE*	WO	0	DMA Channel 3 Block Transmit Done Interrupt Enable 1: Interrupt is enable
5	*CH2TABIE*	WO	0	DMA Channel 2 TABORT Interrupt Enable 1: Interrupt is enable
4	*CH2BTDIE*	WO	0	DMA Channel 2 Block Transmit Done Interrupt Enable 1: Interrupt is enable
3	*CH1TABIE*	WO	0	DMA Channel 1 TABORT Interrupt Enable 1: Interrupt is enable
2	*CH1BTDIE*	WO	0	DMA Channel 1 Block Transmit Done Interrupt Enable 1: Interrupt is enable
1	*CH0TABIE*	WO	0	DMA Channel 0 TABORT Interrupt Enable 1: Interrupt is enable
0	*CH0BTDIE*	WO	0	DMA Channel 0 Block Transmit Done Interrupt Enable 1: Interrupt is enable

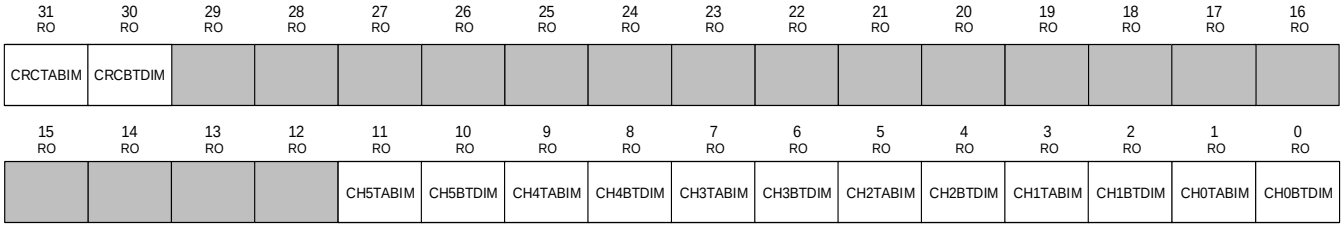
4.19.3.6 DMA_IDR - INTERRUPT DISABLE



Offset: 0x1014

Bits	Name	Type	Reset	Description
31	*CRCTABID*	WO	0	CRC DMA TABORT Interrupt Disable 1: Interrupt is disabled
30	*CRCBTDID*	WO	0	CRC DMA Block Transmit Done Interrupt Disable 1: Interrupt is disabled
29:12	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
11	*CH5TABID*	WO	0	DMA Channel 5 TABORT Interrupt Disable 1: Interrupt is disabled
10	*CH5BTDID*	WO	0	DMA Channel 5 Block Transmit Done Interrupt Disable 1: Interrupt is disabled
9	*CH4TABID*	WO	0	DMA Channel 4 TABORT Interrupt Disable 1: Interrupt is disabled
8	*CH4BTDID*	WO	0	DMA Channel 4 Block Transmit Done Interrupt Disable 1: Interrupt is disabled
7	*CH3TABID*	WO	0	DMA Channel 3 TABORT Interrupt Disable 1: Interrupt is disabled
6	*CH3BTDID*	WO	0	DMA Channel 3 Block Transmit Done Interrupt Disable 1: Interrupt is disabled
5	*CH2TABID*	WO	0	DMA Channel 2 TABORT Interrupt Disable 1: Interrupt is disabled
4	*CH2BTDID*	WO	0	DMA Channel 2 Block Transmit Done Interrupt Disable 1: Interrupt is disabled
3	*CH1TABID*	WO	0	DMA Channel 1 TABORT Interrupt Disable 1: Interrupt is disabled
2	*CH1BTDID*	WO	0	DMA Channel 1 Block Transmit Done Interrupt Disable 1: Interrupt is disabled
1	*CH0TABID*	WO	0	DMA Channel 0 TABORT Interrupt Disable 1: Interrupt is disabled
0	*CH0BTDID*	WO	0	DMA Channel 0 Block Transmit Done Interrupt Disable 1: Interrupt is disabled

4.19.3.7 DMA_IMR - INTERRUPT MASK REGISTER



Offset: 0x1018

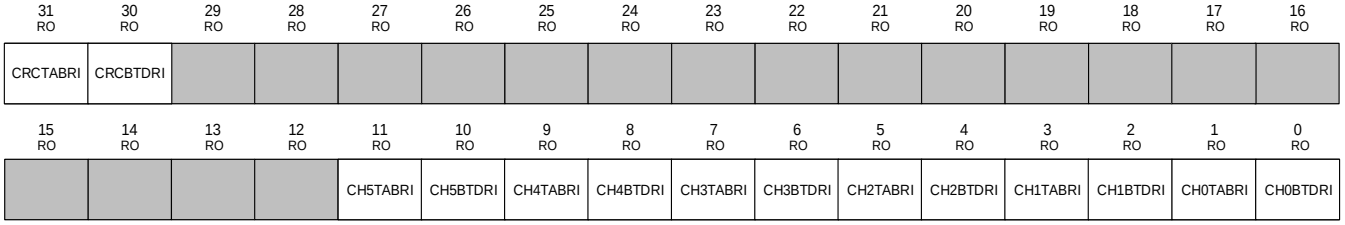
Bits	Name	Type	Reset	Description
31	*CRCTABIM*	RO	0	CRC DMA TABORT Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
30	*CRCBTDIM*	RO	0	CRC DMA Block Transmit Done Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
29:12	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
11	*CH5TABIM*	RO	0	DMA Channel 5 TABORT Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
10	*CH5BTDIM*	RO	0	DMA Channel 5 Block Transmit Done Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
9	*CH4TABIM*	RO	0	DMA Channel 4 TABORT Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
8	*CH4BTDIM*	RO	0	DMA Channel 4 Block Transmit Done Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
7	*CH3TABIM*	RO	0	DMA Channel 3 TABORT Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
6	*CH3BTDIM*	RO	0	DMA Channel 3 Block Transmit Done Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
5	*CH2TABIM*	RO	0	DMA Channel 2 TABORT Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
4	*CH2BTDIM*	RO	0	DMA Channel 2 Block Transmit Done Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
3	*CH1TABIM*	RO	0	DMA Channel 1 TABORT Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
2	*CH1BTDIM*	RO	0	DMA Channel 1 Block Transmit Done Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked

Bits	Name	Type	Reset	Description
1	*CH0TABIM*	RO	0	DMA Channel 0 TABORT Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
0	*CH0BTDIM*	RO	0	DMA Channel 0 Block Transmit Done Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked

4.19.3.8 DMA_RIS - RAW INTERRUPT REGISTER

Note: Target Abort (TABORT) happens under following conditions:

1. DMA transfer or receive an address which has exceeded the valid mapping address.
2. DMA parameters error which is set by the user.
   a. DMA_CSRx.SWSEL =8, DMA_CSRx.DWSEL=16, DMA_BCRx.BCR=1.3.5.7…
   b. DMA_CSRx.SWSEL =8, DMA_CSRx.DWSEL =32, DMA_BCRx.BCR = except any multiple of 4.
   c. DMA_CSRx.SWSEL =16, DMA_CSRx.DWSEL =32, DMA_BCRx.BCR =1.3.5.7…



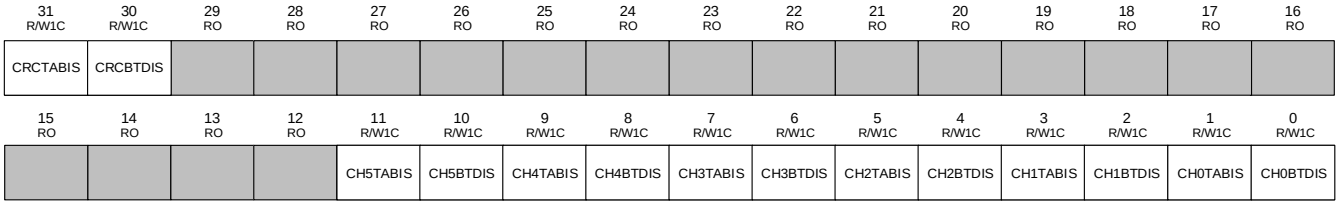
Offset: 0x101C

Bits	Name	Type	Reset	Description
31	*CRCTABRI*	RO	0	CRC DMA TABORT Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
30	*CRCBTDRI*	RO	0	CRC DMA Block Transmit Done Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
29: 12	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
11	*CH5TABRI*	RO	0	DMA Channel 5 TABORT Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
10	*CH5BTDRI*	RO	0	DMA Channel 5 Block Transmit Done Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
9	*CH4TABRI*	RO	0	DMA Channel 4 TABORT Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
8	*CH4BTDRI*	RO	0	DMA Channel 4 Block Transmit Done Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
7	*CH3TABRI*	RO	0	DMA Channel 3 TABORT Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

Bits	Name	Type	Reset	Description
6	*CH3BTDRI*	RO	0	DMA Channel 3 Block Transmit Done Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
5	*CH2TABRI*	RO	0	DMA Channel 2 TABORT Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
4	*CH2BTDRI*	RO	0	DMA Channel 2 Block Transmit Done Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
3	*CH1TABRI*	RO	0	DMA Channel 1 TABORT Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
2	*CH1BTDRI*	RO	0	DMA Channel 1 Block Transmit Done Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
1	*CH0TABRI*	RO	0	DMA Channel 0 TABORT Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
0	*CH0BTDRI*	RO	0	DMA Channel 0 Block Transmit Done Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.19.3.9 DMA_ISC - INTERRUPT STATUS AND CLEAR REGISTER

Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.

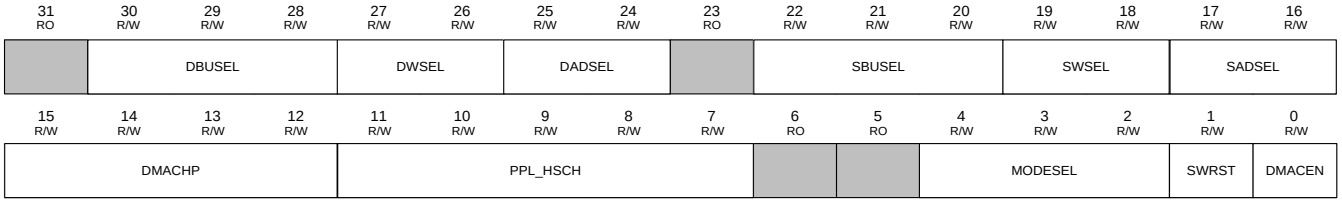


Offset: 0x1020

Bits	Name	Type	Reset	Description
31	*CRCTABIS*	R/W1C	0	CRC DMA TABORT Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
30	*CRCBTDIS*	R/W1C	0	CRC DMA Block Transmit Done Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
29:12	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
11	*CH5TABIS*	R/W1C	0	DMA Channel 5 TABORT Interrupt Status and Clear. 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
10	*CH5BTDIS*	R/W1C	0	DMA Channel 5 Block Transmit Done Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
9	*CH4TABIS*	R/W1C	0	DMA Channel 4 TABORT Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
8	*CH4BTDIS*	R/W1C	0	DMA Channel 4 Block Transmit Done Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
7	*CH3TABIS*	R/W1C	0	DMA Channel 3 TABORT Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
6	*CH3BTDIS*	R/W1C	0	DMA Channel 3 Block Transmit Done Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
5	*CH2TABIS*	R/W1C	0	DMA Channel 2 TABORT Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
4	*CH2BTDIS*	R/W1C	0	DMA Channel 2 Block Transmit Done Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
3	*CH1TABIS*	R/W1C	0	DMA Channel 1 TABORT Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
2	*CH1BTDIS*	R/W1C	0	DMA Channel 1 Block Transmit Done Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.

Bits	Name	Type	Reset	Description
1	*CH0TABIS*	R/W1C	0	DMA Channel 0 TABORT Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
0	*CH0BTDIS*	R/W1C	0	DMA Channel 0 Block Transmit Done Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.

4.19.3.10 DMA_CSRX - DMA CHANNEL X CONTROL AND STATUS REGISTER



Offset:

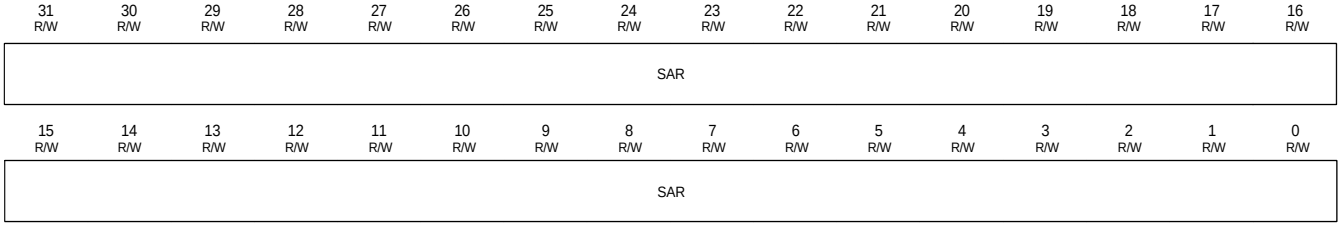
- DMA_CSR0: 0x0000
- DMA_CSR1: 0x0100
- DMA_CSR2: 0x0200
- DMA_CSR3: 0x0300
- DMA_CSR4: 0x0400
- DMA_CSR5: 0x0500

Bits	Name	Type	Reset	Description
31	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
30:28	*DBUSEL*	R/W	0x0	Transfer Destination Burst Selection. 000 : BURST SINGLE 001 : Reserved 010 : Reserved 011 : BURST INCR4 100 : Reserved 101 : BURST INCR8 110 : Reserved 111 : BURST INCR16
27:26	*DWSEL*	R/W	0x0	Transfer Destination Width Selection. 00: One byte (8-bit) is transferred for every DMA operation. 01: One half-word (16-bit) is transferred for every DMA operation. 10: One word (32-bit) is transferred for every DMA operation. 11 : Reserved
25:24	*DADSEL*	R/W	0x0	Transfer Destination Address Direction Selection. 00: Transfer destination address was increased. 01: Transfer destination address was decreased. 10: Transfer destination address was fixed. (This feature can be used when data where transferred from multiple sources to a single destination) 11 : Reserved
23	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

Bits	Name	Type	Reset	Description
22:20	*SBUSEL*	R/W	0x0	Transfer Source Burst Select 000: BURST SINGLE 001: Reserved 010: Reserved 011: BURST INCR4 100: Reserved 101: BURST INCR8 110: Reserved 111: BURST INCR16
19:18	*SWSEL*	R/W	0x0	Transfer Source Width Select 00: One byte (8-bit) is transferred for every DMA operation. 01: One half-word (16-bit) is transferred for every DMA operation. 10: One word (32-bit) is transferred for every DMA operation. 11: Reserved
17:16	*SADSEL*	R/W	0x0	Transfer Source Address Direction Select 00: Transfer Source address was increased. 01: Transfer Source address was decreased. 10: Transfer Source address was fixed. (This feature can be used when data where transferred from multiple sources to a single destination) 11 : Reserved
15:12	*DMACHP*	R/W	0x0	DMA Channel Priority. The priority's value is from 5 to 0. When two or above c 0 is the highest priority… 5 is the lowest priority.
11:7	*PPL_HSCH*	R/W	0x0	Peripheral Software Handshake Select 0: UART0 Rx 1: UART0 Tx 2: UART1 Rx 3: UART1 Tx 4: UART2 Rx 5: UART2 Tx 6: SPI Rx 7: SPI Tx 8: I2C0 Rx 9: I2C0 Tx 10: I2C1 Rx 11: I2C1 Tx 12: ADC Rx Group Low 13: ADC Rx Group High 14: SPIS0 Rx 15: SPIS0 Tx 16: SPIS1 Rx 17: SPIS1 Tx 18: DAC Tx …31: reserved
6:5	*reserved*	R/W	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

Bits	Name	Type	Reset	Description
4:2	*MODESEL*	R/W	0x0	DMA Mode Select. 000: Memory to Memory mode (Memory-to-Memory). 001: Peripheral to Memory mode (Peripheral-to-Memory). 010: Memory to Peripheral mode (Memory-to-Peripheral). 011-111: Reserved
1	*SWRST*	R/W	0	Software Engine Reset. 0: No effect. 1: Reset the internal state machine and pointers. The contents of control register will not be cleared. This bit will be auto cleared after few clock cycles.
0	*DMACEN*	R/W	0	DMA Channel Enable. Setting this bit to “1” enables DMA’s operation. If this bit is cleared, DMA will ignore all DMA request and force Bus Master into IDLE state. Note: SW_RST will clear this bit.

4.19.3.11 DMA_SARX - DMA CHANNEL X SOURCE ADDRESS REGISTER

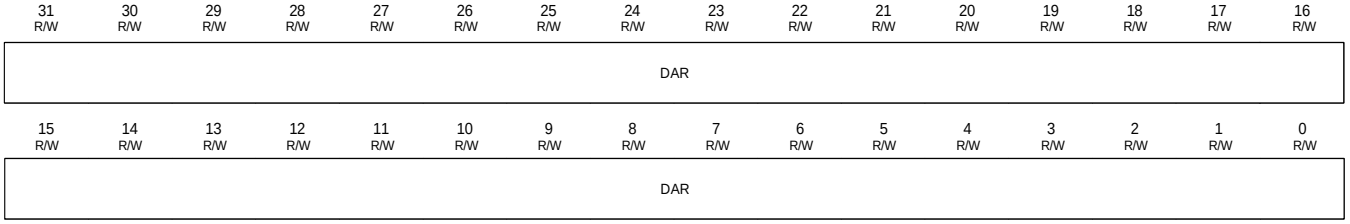


Offset:

- DMA_SAR0: 0x0004
- DMA_SAR1: 0x0104
- DMA_SAR2: 0x0204
- DMA_SAR3: 0x0304
- DMA_SAR4: 0x0404
- DMA_SAR5: 0x0504

Bits	Name	Type	Reset	Description
31:0	*SAR*	R/W	0x0	DMA Transfer Source Address Register This field indicates a 32-bit source address of DMA.

4.19.3.12 DMA_DARX - DMA CHANNEL X DESTINATION ADDRESS REGISTER

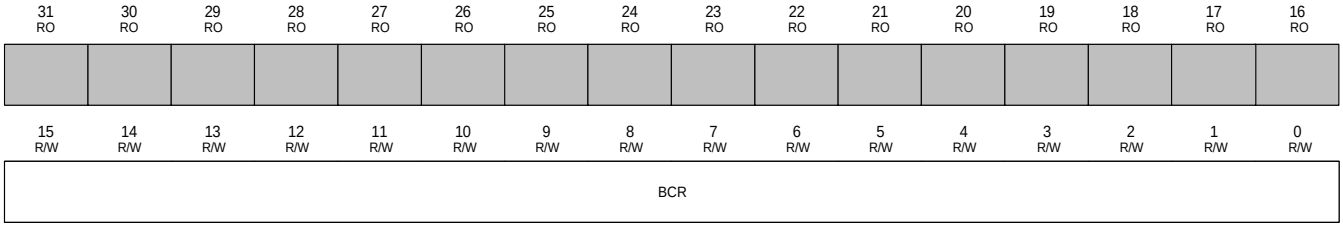


Offset:

- DMA_DAR0: 0x0008
- DMA_DAR1: 0x0108
- DMA_DAR2: 0x0208
- DMA_DAR3: 0x0308
- DMA_DAR4: 0x0408
- DMA_DAR5: 0x0508

Bits	Name	Type	Reset	Description
31:0	*DAR*	R/W	0x0	DMA Transfer Destination Address Register This field indicates a 32-bit destination address of DMA.

4.19.3.13 DMA_BCRX - DMA CHANNEL X TRANSFER BYTE COUNT REGISTER

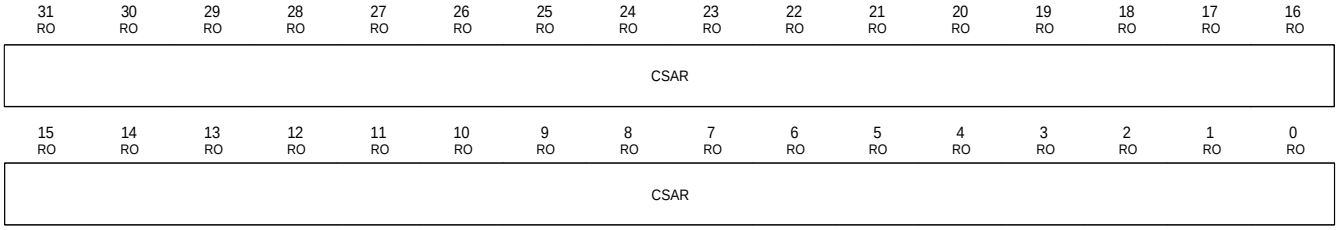


Offset:

- DMA_BCR0: 0x000C
- DMA_BCR1: 0x010C
- DMA_BCR2: 0x020C
- DMA_BCR3: 0x030C
- DMA_BCR4: 0x040C
- DMA_BCR5: 0x050C

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
15:0	*BCR*	R/W	0x0	DMA Transfer Byte Count Register. This field indicates a 16-bit transfer byte count of DMA.

4.19.3.14 DMA_CSARX - DMA CHANNEL X CURRENT SOURCE ADDRESS REGISTER

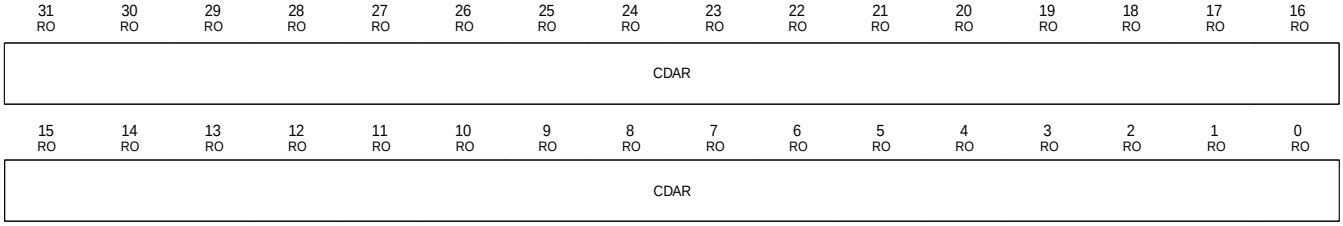


Offset:

- DMA_CSAR0: 0x0010
- DMA_CSAR1: 0x0110
- DMA_CSAR2: 0x0210
- DMA_CSAR3: 0x0310
- DMA_CSAR4: 0x0410
- DMA_CSAR5: 0x0510

Bits	Name	Type	Reset	Description
31:0	*CSAR*	RO	0x0	DMA Current Source Address Register. This field indicates the source address where the DMA transfer is just occurring

4.19.3.15 DMA_CDARX - DMA CHANNEL X CURRENT DESTINATION ADDRESS REGISTER

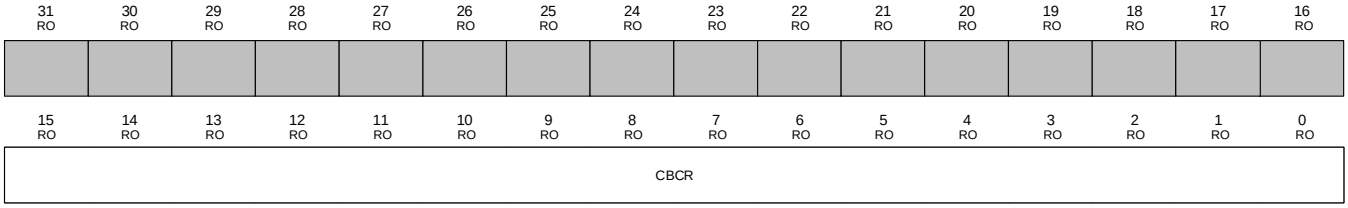


Offset:

- DMA_CDAR0: 0x0014
- DMA_CDAR1: 0x0114
- DMA_CDAR2: 0x0214
- DMA_CDAR3: 0x0314
- DMA_CDAR4: 0x0414
- DMA_CDAR5: 0x0514

Bits	Name	Type	Reset	Description
31:0	*CDAR*	RO	0x0	DMA Current Destination Address Register. This field indicates the destination address where the DMA transfer is just occurring

4.19.3.16 DMA_CBCRX - DMA CHANNEL X CURRENT TRANSFER BYTE COUNT REGISTER



Offset:

- DMA_CBCR0: 0x0018
- DMA_CBCR1: 0x0118
- DMA_CBCR2: 0x0218
- DMA_CBCR3: 0x0318
- DMA_CBCR4: 0x0418
- DMA_CBCR5: 0x0518

Bits	Name	Type	Reset	Description
31:16	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
15:0	*CBCR*	RO	0x0	DMA Current Byte Count Register. This field indicates the current remained byte count of DMA. Note : These fields will be changed when DMA finish data transfer (data transfer to destination address)

4.20 UNIVERSAL SERIAL BUS (USB)

The USB controller is a USB 2.0 full-speed controller. It is compliant with USB 2.0 full speed device specification and supports control/bulk/interrupt/isochronous transfer types.

The USB controller can be used in a range of different environment. It can be used as either a full-speed or a low-speed USB peripheral device attached to a conventional USB host. It can be used as either a host or a peripheral device in a point-to-point type of setup/arrangement or it can be used as either a host connecting to a range of peripheral devices in a multi-point setup. DMA access to the FIFO allows minimal interference from system software.

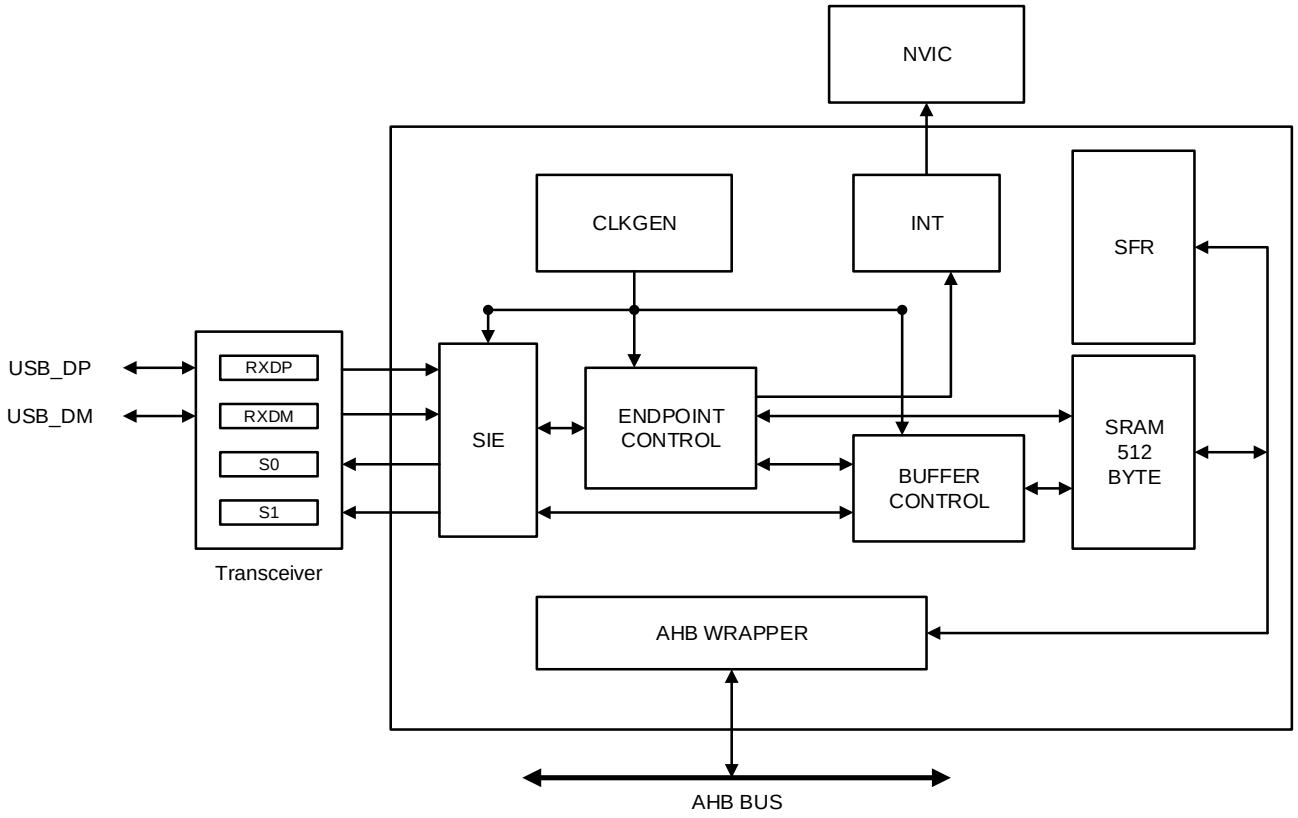
14 endpoints including hard-wired for control transfers (one endpoint for IN and one endpoint for OUT) plus 12 endpoints defined by firmware along with a dynamic sizable FIFO.

The USB controller, there are two main interfaces: the AHB bus and USB bus which comes from the USB PHY transceiver. For the AHB bus, the CPU can program control registers through it. There is an internal 1Kbyte SRAM FIFO buffers as data buffer in this controller. For IN token or OUT token transfer, it is necessary to write data to SRAM or read data from SRAM through the AHB interface.

- Operates either as a function controller for a USB peripheral or as the host/peripheral in point-to-point communications with another USB function.
- Compatible with the USB 2.0 specification
- Supports full (12Mbit/s) and low (1.5Mbit/s) speed communication
- *Note: In host mode, all speeds are supported. However, in device mode, only full speed are supported.*
- 4 transfer types: Control, Interrupt, Bulk, and Isochronous
- 14 endpoints
  - 1 dedicated control IN endpoint and 1 dedicated control OUT endpoint
  - 6 configurable IN endpoints and 6 configurable OUT endpoints
- Configurable USB power
- Configurable pull up/down on DP & DM pinout.
- Supports Session Request Protocol (SRP) and Host Negotiation Protocol (HNP).
- Supports Suspend and Resume signaling.
- Dynamic FIFO sizing up to 1K bytes
- Efficient transfer using DMA

4.20.1 BLOCK DIAGRAM

Figure 4.20-1: USB Block Diagram



4.20.2 FUNCTIONAL DESCRIPTION

4.20.2.1 OPERATION MODE

The USB has two main modes of operation - Device mode and Host mode.

OPERATION AS A DEVICE

The USB controller can work as a device. The USB encodes, decodes, checks and directs all USB packets sent and received. IN transactions are handled through the device's Tx FIFOs, OUT transactions are handled through its Rx FIFOs. This is the mode in which the USB needs to be operating when the device in which it is used is acting as the peripheral to a standard USB Host or as the peripheral in point-to-point communications. Control, Bulk, Isochronous and Interrupt transactions are supported.

OPERATION AS A HOST

The USB offers the (limited) capabilities needed in order to act as the host in point-to-point communications with another USB function. In this mode, it can communicate with one full-speed or low-speed device only (not a hub), supporting Control, Bulk, Isochronous or Interrupt transactions. IN transactions are handled through the Rx FIFOs, OUT transactions are handled through the Tx FIFOs. As well as encoding, decoding and checking the USB packets sent and received, the USB will also automatically schedule Isochronous endpoints and Interrupt endpoints to perform one transaction every n frames, where n is the polling interval that has been programmed for the endpoint (between 1 and 255ms). The remaining bus bandwidth is shared equally amongst the Control and Bulk endpoints.

4.20.2.2 SERIAL INTERFACE ENGINE (SIE)

The SIE handles NRZI encoding/decoding, bit stuffing/unstuffing, and CRC generation/checking of USB packets sent and received. It generates a 12 MHz USB clock from the 48 MHz input clock to the core, and synchronizes this clock to the incoming data stream when receiving data from the USB. It generates headers for packets to be transmitted and decodes the headers of received packets.

4.20.2.3 ENDPOINT CONTROLLERS

Two controller state machines are used: one for control transfers over Endpoint 0, and one for Bulk, Interrupt or Isochronous transactions over Endpoints 1 to 6.

The detailed behavior of the Endpoint 0 state machine depends on whether the device is being used in Host mode or in Peripheral mode. In particular, when operating in Host mode, the controller additionally calls on a transaction scheduler to ensure that Isochronous and Interrupt transactions occur in the required frame and to distribute the remaining bus bandwidth amongst the Bulk endpoints.

When configuring the size of the FIFOs for endpoints, take into account the maximum packet size for an endpoint.

4.20.2.4 RAM CONTROLLER

The RAM Controller provides an interface to a single block of synchronous single-port RAM, which is used to buffer packets between the CPU and the USB bus. It takes the FIFO pointers from the endpoint controllers, converts them to address pointers within the RAM block and generates the necessary RAM access control signals.

4.20.3 USB REGISTER MAP

The USB register map is split into four sections:

- Common USB registers (00h-0Fh) - These registers provide control and status for the complete core.
- Endpoint Control/Status registers (10h-1Fh, indexed) - These registers provide control and status for the endpoints. The registers mapped into this section depend on whether the core is in Peripheral mode (DevCtl.D2=0) or in Host mode (DevCtl.D2=1) and on the value of the Index register.
- FIFOs (20h-38h) - This address range provides access to the endpoint FIFOs.
- Interrupts (80h-B0h) - This address range provides access to the USB's interrupts control.

Base Address: 0x5004_0000

COMMON USB REGISTER:

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	USB_FADDR	R/W	0x00	Function Address Register.	491
0x0001	USB_POWER	R/W	0x00	Power Management Register.	492
0x0002	USB_DPDMCTL	R/W	0x00	DP/DM control register	494
0x0003	USB_SWCID	R/W	0x03	Software control CID register	494
0x0004	USB_SWVBUS	R/W	0x0F	Software control VBUS register	495
0x0005	USB_TMODE	R/W	0x00	Test mode register	495
0x0006~ 0x000B	reserved	N/A	0x00	Reserved	-
0x000C	USB_FRAME1	R/W	0x00	Frame Number Bits 0 To 7.	496
0x000D	USB_FRAME2	R/W	0x00	Frame Number Bits 8 To 10.	496
0x000E	USB_INDEX	R/W	0x00	Index Register For Selecting Endpoint Status and Control Registers.	497
0x000F	USB_DEVCTL	R/W	0x88	USB Device Control Register.	498

INDEXED REGISTERS - PERIPHERAL MODE:

(Control Status registers for endpoint selected by the Index register when USB_DEVCTL.*HOST* = 0)

Offset	Symbol	Type	Reset Value	Description	See page
0x0010	USB_TXMAXP	R/W	0x00	Maximum Packet Size For Peripheral IN Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	499
0x0x11	USB_CSR0	R/W	0x00	Main Control Status Register For Endpoint 0. (Index Register Set To Select Endpoint 0)	500
	USB_TXCSR1	R/W	0x00	Control Status Register 1 For Peripheral IN Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	502
0x0x12	USB_CSR02	R/W	0x00	Subsidiary Control Status Register For Endpoint 0, Containing Flush fifo Bit. (Index Register Set To Select Endpoint 0)	504
	USB_TXCSR2	R/W	0x00	Control Status Register 2 For Peripheral IN Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	505
0x0013	USB_RXMAXP	R/W	0x00	Maximum Packet Size For Peripheral OUT Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	506
0x0014	USB_RXCSR1	R/W	0x00	Control Status Register 1 For Peripheral OUT Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	507
0x0015	USB_RXCSR2	R/W	0x00	Control Status Register 2 For Peripheral OUT Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	509
0x0x16	USB_COUNT0	R/W	0x00	Number Of Received Bytes In Endpoint 0 FIFO. (Index Register Set To Select Endpoints 1 - 6 Only)	510
	USB_RXCOUNT1	R/W	0x00	Number Of Bytes In Peripheral OUT Endpoint FIFO (Lower Byte). (Index Register Set To Select Endpoints 1 - 6 Only)	510
0x0017	USB_RXCOUNT2	R/W	0x00	Number Of Bytes In Peripheral OUT Endpoint FIFO (Upper Byte). (Index Register Set To Select Endpoints 1 - 6 Only)	511
0x0018~ 0x001B	reserved	N/A	0x00	Reserved	-
0x001C	USB_TXFIFO1	R/W	0x00	Tx Endpoint FIFO Details	514
0x001D	USB_TXFIFO2	R/W	0x00	Tx Endpoint FIFO Details	515
0x001E	USB_RXFIFO1	R/W	0x00	Rx Endpoint FIFO Details	516
0x001F	USB_RXFIFO2	R/W	0x00	Rx Endpoint FIFO Details	517

INDEXED REGISTERS - HOST MODE:

(Control Status registers for endpoint selected by the Index register when USB_DEVCTL.*HOST* = 1)

Offset	Symbol	Type	Reset Value	Description	See page
0x0010	USB_TXMAXP	R/W	0x00	Maximum Packet Size For Peripheral IN Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	499
0x0x11	USB_CSR0	R/W	0x00	Control Status Register For Endpoint 0. (Index Register Set To Select Endpoint 0)	500
	USB_TXCSR1	R/W	0x00	Status Register 1 For Peripheral IN Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	502
0x0x12	USB_CSR02	R/W	0x00	Subsidiary Control Status Register For Endpoint 0, Containing Flush fifo Bit. (Index Register Set To Select Endpoint 0)	504
	USB_TXCSR2	R/W	0x00	Control Status Register 2 For Peripheral IN Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	505
0x0013	USB_RXMAXP	R/W	0x00	Maximum packet size for host IN endpoint. (Index register set to select Endpoints 1 - 6 only)	506
0x0014	USB_RXCSR1	R/W	0x00	Control Status Register 1 For Peripheral OUT Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	507
0x0015	USB_RXCSR2	R/W	0x00	Control Status Register 2 For Peripheral OUT Endpoint. (Index Register Set To Select Endpoints 1 - 6 Only)	509
0x0x16	USB_COUNT0	R/W	0x00	Control Status register 2 for host IN endpoint. (Index register set to select Endpoints 1 - 6 only)	510
	USB_RXCOUNT1	R/W	0x00	Number of bytes in host IN endpoint FIFO (lower byte). (Index register set to select Endpoints 1 - 6 only)	510
0x0017	USB_RXCOUNT2	R/W	0x00	Number of bytes in host IN endpoint FIFO (upper byte). (Index register set to select Endpoints 1 - 6 only)	511
0x0018	USB_TXTYPE	R/W	0x00	Sets the transaction protocol and peripheral endpoint number for the host OUT endpoint. (Index register set to select Endpoints 1 - 6 only)	511
0x0x19	USB_NAKLIMIT0	R/W	0x00	Sets the NAK response timeout on Endpoint 0. (Index register set to select Endpoint 0)	512
	USB_TXINTERVAL	R/W	0x00	Sets the polling interval for an OUT Interrupt endpoint, in ms. (Index register set to select Endpoints 1 - 6 only)	512
0x001A	USB_RXTYPE	R/W	0x00	Sets the transaction protocol and peripheral endpoint number for the host IN endpoint. (Index register set to select Endpoints 1 - 6 only)	513
0x001B	USB_RXINTERVAL	R/W	0x00	Sets the polling interval for an IN Interrupt endpoint, in ms. (Index register set to select Endpoints 1 - 6 only)	513
0x001C	USB_TXFIFO1	R/W	0x00	Tx Endpoint FIFO Details	514
0x001D	USB_TXFIFO2	R/W	0x00	Tx Endpoint FIFO Details	515
0x001E	USB_RXFIFO1	R/W	0x00	Rx Endpoint FIFO Details	516
0x001F	USB_RXFIFO2	R/W	0x00	Rx Endpoint FIFO Details	517

FIFOS REGISTERS：

Offset	Symbol	Type	Reset Value	Description	See page
0x0020	USB_EP0FIFO	R/W	0x00	FIFOs for Endpoints 0	518
0x0021~ 0x0023	reserved	N/A	0x00	Reserved	-
0x0024	USB_EP1FIFO	R/W	0x00	FIFOs for Endpoints 1	518
0x0025~ 0x0027	reserved	N/A	0x00	Reserved	-
0x0028	USB_EP2FIFO	R/W	0x00	FIFOs for Endpoints 2	518
0x0029~ 0x002B	reserved	N/A	0x00	Reserved	-
0x002C	USB_EP3FIFO	R/W	0x00	FIFOs for Endpoints 3	519
0x002D~ 0x002F	reserved	N/A	0x00	Reserved	-
0x0030	USB_EP4FIFO	R/W	0x00	FIFOs for Endpoints 4	519
0x0031~ 0x0033	reserved	N/A	0x00	Reserved	-
0x0034	USB_EP5FIFO	R/W	0x00	FIFOs for Endpoints 5	519
0x0035~ 0x0037	reserved	N/A	0x00	Reserved	-
0x0038	USB_EP6FIFO	R/W	0x00	FIFOs for Endpoints 6	520
0x0039~ 0x003B	reserved	N/A	0x00	Reserved	-

INTERRUPTS REGISTERS:

Offset	Symbol	Type	Reset Value	Description	See page
0x0080	USB_TXIER1	WO	0x00	TX Endpoint 0~6 Interrupt Enable	521
0x0081	reserved	N/A	0x00	Reserved	-
0x0082	USB_RXIER1	WO	0x00	RX Endpoint 0~6 Interrupt Enable	522
0x0083	reserved	N/A	0x00	Reserved	-
0x0084	USB_TXIDR1	WO	0x00	TX Endpoint 0~6 Interrupt Disable	523
0x0085	reserved	N/A	0x00	Reserved	-
0x0086	USB_RXIDR1	WO	0x00	RX Endpoint 0~6 Interrupt Disable	524
0x0087	reserved	N/A	0x00	Reserved	-
0x0088	USB_TXIMR1	RO	0x00	TX Endpoint 0~6 Interrupt Mask Status	525
0x0089	reserved	N/A	0x00	Reserved	-
0x008A	USB_RXIMR1	RO	0x00	RX Endpoint 0~6 Interrupt Mask Status	526
0x008B	reserved	N/A	0x00	Reserved	-
0x008C	USB_TXRIS1	RO	0x00	TX Endpoint 0~6 Raw Interrupt Status	527
0x008D	reserved	N/A	0x00	Reserved	-
0x008E	USB_RXRIS1	RO	0x00	RX Endpoint 0~6 Raw Interrupt Status	528
0x008F	reserved	N/A	0x00	Reserved	-
0x0090	USB_TXISC1	R/W	0x00	TX Endpoint 0~6 Interrupt Status and Clear	529
0x0091	reserved	N/A	0x00	Reserved	-
0x0092	USB_RXISC1	R/W	0x00	RX Endpoint 0~6 Interrupt Status and Clear	530
0x0093~ 0x009C	reserved	N/A	0x00	Reserved	-
0x00A0	USB_IER	W	0x00	USB Interrupt Enable	531
0x00A4	USB_IDR	W	0x00	USB Interrupt Disable	533
0x00A8	USB_IMR	R	0x00	USB Interrupt Mask Status	535
0x00AC	USB_RIS	R	0x00	USB Raw Interrupt Status	537
0x00B0	USB_ISC	R/W	0x00	USB Interrupt Status and Clear	539

4.20.3.1 COMMON REGISTERSCOM

4.20.3.1.1 USB_FADDR - USB DEVICE FUNCTION ADDRESS REGISTER

7	6	5	4	3	2	1	0
RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	FUNC_ADDR						

Offset: 0x0000

Bits	Name	Type	Reset	Description
7	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6:0	*FUNC_ADDR*	R/W/R	0x0	Function Address

4.20.3.1.2 USB_POWER - USB POWER MANAGEMENT REGISTER

This 8-bit register is used for controlling Suspend and Resume signaling, IN packet timing for isochronous endpoints and, for point-to-point communications, indicating the type of device connected.

Host mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ISO_UPDATE	VBUS_VAL	VBUS_SESS	VBUS_LO	RESET	RESUME	SUSPEND_MODE	ENABLE_SUSPEND

Offset: 0x0001

Bits	Name	Type	Reset	Description
7	*ISO_UPDATE*	R/RW R/RW	0	Set by the CPU to cause the USB to wait for a SOF token after TxPktRdy has been set before sending the packet. If an IN token is received before a SOF token, a zero length data packet will be sent. This bit is only valid when the USB is in Peripheral mode and then only affects endpoints performing Isochronous transfers.
6	*VBUS_VAL*	R/RW R/RW	0	This bit indicates the state of the VBUSVAL input signal to the USB. 0: VBus below this threshold. 1: VBus above the VBus Valid threshold.
5	*VBUS_SESS*	R/RW R/RW	0	This bit indicates the state of the VBUSSES input signal to the USB. 0: VBus below this threshold. 1: VBus above the Session Valid threshold for a 'B' device.
4	*VBUS_LO*	R/RW R/RW	0	This bit indicates the state of the VBUSLO input signal to the USB. 0: VBus below this threshold. 1: VBus above the Session End threshold.
3	*RESET*	RW/RW R/RW	0	In Peripheral mode, this read-only bit is set while Reset signaling is present on the bus. In Host mode, this bit is set by the CPU for 20ms, to generate Reset signaling on the bus. It is automatically set when Host mode is entered following Host Negotiation: the CPU should then clear it after 20ms.
2	*RESUME*	RW/R RW/R	0	Set by the CPU to generate Resume signaling when the device is in Suspend mode. In Peripheral mode, the CPU should clear this bit after 10ms (a maximum of 15ms), to end Resume signaling. In Host mode, the CPU should clear this bit after 20ms.
1	*SUSPEND_MODE*	SET/CLR R/SET	0	In Peripheral mode, this bit is set by the USB when Suspend mode is entered. In Host mode, this bit is set by the CPU when Suspend mode is to be entered. The bit is cleared when the CPU reads the Interrupt register or sets the Resume bit (above) or leaves Host mode.
0	*ENABLE_SUSPEND*	R/R RW/R	0	Set by the CPU to enable entry into Suspend mode when Suspend signaling is received on the bus. Note: This bit is only valid when the USB is in Peripheral mode.

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	RO	RO	RO	R/W	R/W	R/W	R/W
ISOUDT				RESET	RESUME	*SUSPEND*	SUSPENDEN

Offset: 0x0001

Bits	Name	Type	Reset	Description
7	*ISO_UPDATE*	R/RW R/RW	0	Set by the CPU to cause the USB to wait for a SOF token after TxPktRdy has been set before sending the packet. If an IN token is received before a SOF token, a zero length data packet will be sent. This bit is only valid when the USB is in Peripheral mode and then only affects endpoints performing Isochronous transfers.
6	*VBUS_VAL*	R/RW R/RW	0	This bit indicates the state of the VBUSVAL input signal to the USB. 0: VBus below this threshold. 1: VBus above the VBus Valid threshold.
5	*VBUS_SESS*	R/RW R/RW	0	This bit indicates the state of the VBUSSES input signal to the USB. 0: VBus below this threshold. 1: VBus above the Session Valid threshold for a 'B' device.
4	*VBUS_LO*	R/RW R/RW	0	This bit indicates the state of the VBUSLO input signal to the USB. 0: VBus below this threshold. 1: VBus above the Session End threshold.
3	*RESET*	RW/RW R/RW	0	In Peripheral mode, this read-only bit is set while Reset signaling is present on the bus. In Host mode, this bit is set by the CPU for 20ms, to generate Reset signaling on the bus. It is automatically set when Host mode is entered following Host Negotiation: the CPU should then clear it after 20ms.
2	*RESUME*	RW/R RW/R	0	Set by the CPU to generate Resume signaling when the device is in Suspend mode. In Peripheral mode, the CPU should clear this bit after 10ms (a maximum of 15ms), to end Resume signaling. In Host mode, the CPU should clear this bit after 20ms.
1	*SUSPEND_MODE*	SET/CLR R/SET	0	In Peripheral mode, this bit is set by the USB when Suspend mode is entered. In Host mode, this bit is set by the CPU when Suspend mode is to be entered. The bit is cleared when the CPU reads the Interrupt register or sets the Resume bit (above) or leaves Host mode.
0	*ENABLE_SUSPEND*	R/R RW/R	0	Set by the CPU to enable entry into Suspend mode when Suspend signaling is received on the bus. Note: This bit is only valid when the USB is in Peripheral mode.

4.20.3.1.3 USB_DPDMCTL - USB DP/DM CONTROLLER REGISTER

This register is designed to allow a user to view and control the status of USB_DP and USB_DM pinout.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	RO	RO	R/W	R/W	R/W	R/W	R/W
			PD_PUPD		PM_PUPD		PHY_PWREN

Offset: 0x0002

Bits	Name	Type	Reset	Description
7:5	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
4:3	*PD_PUPD*	RW/R	0x0	pull up/pull down register 00: floatin 01: pull up 10: pull down 11: reserved
2:1	*PM_PUPD*	RW/R	0x0	pull up/pull down register 00: floatin 01: pull up 10: pull down 11: reserved
0	*PHY_PWREN*	RW/R	0x0	phy power enable

4.20.3.1.4 USB_SWCID - USB SOFTWARE CONTROL CID REGISTER

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	RO	RO	RO	RO	RO	R/W	R/W
						SWCID_MODE	SWCID_EN

Offset: 0x0003

Bits	Name	Type	Reset	Description
7:2	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
1	SWCID_MODE	RW/R	0x1	Select device or host mode 1: device mode 0: host mode
0	SWCID_EN	RW/R	0x1	Connect ID (CID) Signal Control 0: Controlled by software 1: Controlled by hardware

4.20.3.1.5 USB_SWVBUS - USB SOFTWARE CONTROL VBUS REGISTER

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	RO	RO	RO	R/W	R/W	R/W	R/W
				SW_VBUS VAL	SW_VBUS SES	SW_VBUSL O	SWVBUS_ EN

Offset: 0x0004

Bits	Name	Type	Reset	Description
7:4	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3	*SW_VBUSVAL*	RW/R	0x1	If SWVBUS_EN is controled by software, then it will check the setting value; if controled by hardware, settings will be ignored and switch to external control VBUS signal.
2	*SW_VBUSSES*	RW/R	0x1	If SWVBUS_EN is controled by software, then it will check the setting value; if controled by hardware, settings will be ignored and switch to external control VBUS signal.
1	*SW_VBUSLO*	RW/R	0x1	If SWVBUS_EN is controled by software, then it will check the setting value; if controled by hardware, settings will be ignored and switch to external control VBUS signal.
0	*SWVBUS_EN*	RW/R	0x1	VBUS Signal Control 0: Controlled by hardware 1: Controlled by software

4.20.3.1.6 USB_TMODE - USB TEST MODE REGISTER

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
WO	WO	WO	WO	WO	WO	WO	R/W
TMODE_RECOG							TMODE_EN

Offset: 0x0005

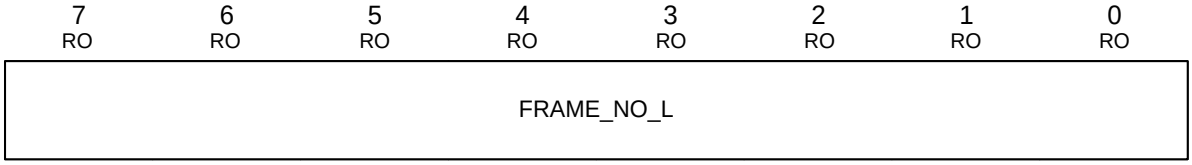
Bits	Name	Type	Reset	Description
7:1	*TMODE_RECOG*	WO	0x0	Protection Field Configure this field as 0x56 before writing *TMODE_EN* bit.
0	*TMODE_EN*	RW/R	0x0	Write 0xAD to enable Test mode Write 0xAC to disable Test mode 0: Normal Mode 1: Test Mode

4.20.3.1.7 USB_FRAME1 - USB FRAME NUMBER REGISTER 1

This read-only register that in Peripheral mode holds the lower 8 bits of the last received frame number and in Host mode holds the lower 8 bits of the current frame number.

Host mode:

Peripheral mode:



Offset: 0x000C

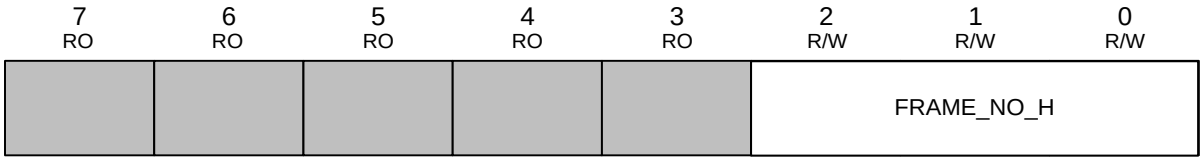
Bits	Name	Type	Reset	Description
7:0	*FRAME_NO_L*	R/RW	0x0	Lower 8 bits of Frame Number.

4.20.3.1.8 USB_FRAME2 - USB FRAME NUMBER REGISTER 2

Frame2 is a 3-bit read-only register that in Peripheral mode holds the upper 3 bits of the last received frame number and in Host mode holds the upper 3 bits of the current frame number.

Host mode:

Peripheral mode:



Offset: 0x000D

Bits	Name	Type	Reset	Description
7:3	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2:0	*FRAME_NO_H*	RW/R	0x0	Upper 3 bits of Frame Number.

4.20.3.1.9 USB_INDEX - USB INDEX REGISTER

Index is a 4-bit register that determines which endpoint control/status registers are accessed at addresses 10h to 1F. Each Tx endpoint and each Rx endpoint have their own set of control/status registers. Only one set of Tx control/status and one set of Rx control/status registers appear in the memory map at any one time. Before accessing an endpoint's control/status registers, the endpoint number should be written to the Index register to ensure that the correct control/status registers appear in the memory map.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	RO	RO	RO	R/W	R/W	R/W	R/W
				ENDPOINT			

Offset: 0x000E

Bits	Name	Type	Reset	Description
7:4	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
3:0	*ENDPOINT*	RW/R	0x0	Selected Endpoint.

4.20.3.1.10 USB_DEVCTRL - USB DEVICE CONTROL REGISTER

USB_DEVCTL is an 8-bit register that is used to select whether the USB is operating in Peripheral mode or in Host mode.

Host mode only:

7	6	5	4	3	2	1	0
RO	RO	RO	RO	RO	R/W	R/W	R/W
	FSDEV	LSDEV			HOST	HOSTREQ	SESSION

Offset: 0x000F

Bits	Name	Type	Reset	Description
7	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
6	*FSDEV*	RO	0	Full-Speed Device Detected 0: A full-speed device has not been detected on the port 1: A full-speed device has been detected on the port.
5	*LSDEV*	RO	0	Low-Speed Device Detected 0: A low-speed device has not been detected on the port 1: A low-speed device has been detected on the port.
4:3	*Reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read.
2	*HOST*	RO	0x0	Host Mode 0: The USB controller is acting as a Device 1: The USB controller is acting as a Host.
1	*HOSTREQ*	R/W	0	Host Request 0: No effect 1: Initiates the Host Negotiation when SUSPEND mode is entered.
0	*SESSION*	R/W	0	Session Start or End 0: When cleared by software, this bit ends a session. 1: When set by software, this bit starts a session.

4.20.3.2 INDEXED REGISTERSCOM

4.20.3.2.1 USB_TXMAXP - USB TX MAXIMUM PACKET SIZE REGISTER

USB_TXMAXP is an 8-bit register that holds the maximum packet size for transactions through the currently-selected Tx endpoint - in units of 8 bytes, except that a value of 128 sets the maximum packet size to 1023 (the maximum size for an Isochronous packet transferred in a full-speed transaction) rather than 1024.

Specification on packet sizes for Bulk, Interrupt and Isochronous transactions in full-speed operations.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
MAX_PACKET_SIZE/TRANSACTION							

Offset: 0x0010

Bits	Name	Type	Reset	Description
7:0	*MAX_PACKET_SIZE/TRANSACTION*	RW/R	0x0	There is a TxMaxP register for each configured Tx endpoint (except Endpoint 0).

The value written to this register should not exceed the Tx FIFO size or, where dynamic FIFO sizing is used, the maximum packet size specified in the TxFIFO2 register. If the value written to this register is less than, or equal to, half the Tx FIFO size, two packets can be buffered (except where dynamic FIFO sizing has been selected).

4.20.3.2.2 USB_CSR1 - USB MAIN CONTROL STATUS REGISTER

CSR0 is an 8-bit register that provides control and status bits for Endpoint 0. The interpretation of the register depends on whether the USB is acting as a peripheral or as a host.

Host mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W
SERVICEDSETUPEND	SERVICEDRXPKTRDY	SENDSTALL	SETUPEND	DATAEND	SENTSTALL	TXPKTRDY	RXPKTRDY

Offset: 0x0011

Bits	Name	Type	Reset	Description
7	SERVICEDSETUPEND	SET/R	0	The CPU writes a 1 to this bit to clear the SetupEnd bit. It is cleared automatically.
6	SERVICEDRXPKTRDY	SET/R	0	The CPU writes a 1 to this bit to clear the RxPktRdy bit. It is cleared automatically.
5	SENDSTALL	SET/R	0	The CPU writes a 1 to this bit to terminate the current transaction. The STALL handshake will be transmitted and then this bit will be cleared automatically.
4	SETUPEND	RO	0	This bit will be set when a control transaction ends before the DataEnd bit has been set. An interrupt will be generated and the FIFO flushed at this time. The bit is cleared by the CPU writing a 1 to the ServicedSetupEnd bit.
3	DATAEND	R/W	0	The CPU sets this bit: 1. When setting TxPktRdy for the last data packet. 2. When clearing RxPktRdy after unloading the last data packet. 3. When setting TxPktRdy for a zero length data packet. It is cleared automatically.
2	SENTSTALL	R/W	0	This bit is set when a STALL handshake is transmitted. The CPU should clear this bit.
1	TXPKTRDY	R/W	0	The CPU sets this bit after loading a data packet into the FIFO. It is cleared automatically when the data packet has been transmitted. An interrupt is generated when the bit is cleared.
0	RXPKTRDY	R/W	0	This bit is set when a data packet has been received. An interrupt is generated when this bit is set. The CPU clears this bit by setting the ServicedRxPktRdy bit.

Peripheral mode:

7	6	5	4	3	2	1	0
W1C	W1C	R/W	RO	R/W	R/W	R/W	RO
SETENDC	RXRDYC	STALL	SETEND	DATAEND	STALLED	TXRDY	RXRDY

Offset: 0x0011

Bits	Name	Type	Reset	Description
7	*SETENDC*	W1C	0	Setup End Clear Writing a 1 to this bit to clear the *SUPEND* bit.
6	*RXRDYC*	W1C	0	RXRDY Clear Writing a 1 to this bit to clear the *RXRDY* bit
5	*STALL*	R/W	0	Send Stall 0: No effect 1: Terminates the current transaction and transmits the STALL handshake.
4	*SETEND*	RO	0	Setup End 0: A control transaction has not ended or ended after the *DATAEND* bit was set. 1: A control transaction has ended before the *DATAEND* bit has been set.
3	*DATAEND*	R/W	0	Data End The CPU sets this bit: 1. When setting *TXRDY* for the last data packet. 2. When clearing *RxPktRdy* after unloading the last data packet. 3. When setting *TXRDY* for a zero length data packet. It is cleared automatically.
2	*STALLED*	R/W	0	Sent Stall 0: A STALL handshake has not been transmitted. 1: A STALL handshake has been transmitted.
1	*TXRDY*	R/W	0	Transmit Packet Ready 0: No transmit packet is ready. 1: Software sets this bit after loading a data packet into the Tx FIFO.
0	*RXRDY*	RO	0	Receive Packet Ready 0: No data packet has been received. 1: A data packet has been received.

4.20.3.2.3 USB_TXCSR1 - USB TX CONTROL STATUS REGISTER 1

TxCSR1 is an 8-bit register that provides control and status bits for transfers through the currently-selected Tx endpoint. There is a TxCSR1 register for each configured Tx endpoint (not including Endpoint 0).The interpretation of the register depends on whether the USB is acting as a peripheral or as a host.

Host mode:

7	6	5	4	3	2	1	0
RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	UNDERRUN	FIFONOTEMPTY	TXPKTRDY

Offset: 0x0011

Bits	Name	Type	Reset	Description
7	reserved	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	CLRDATATOG	SET/R_CLR	0	The CPU writes a 1 to this bit to clear the RxPktRdy bit. It is cleared automatically.
5	SENTSTALL	R_CLR/SET	0	This bit is set when a STALL handshake is transmitted. The FIFO is flushed and the TxPktRdy bit is cleared (see below). The CPU should clear this bit.
4	SENDSTALL	RW/R	0x0	The CPU writes a 1 to this bit to issue a STALL handshake to an IN token. The CPU clears this bit to terminate the stall condition. Note: This bit has no effect where the endpoint is being used for Isochronous transfers.
3	FLUSHFIFO	SET/R	0	The CPU writes a 1 to this bit to flush the next packet to be transmitted from the endpoint Tx FIFO. The FIFO pointer is reset and the TxPktRdy bit (below) is cleared. Note: FlushFIFO has no effect unless TxPktRdy is set. Also note that, if the FIFO is double-buffered, FlushFIFO may need to be set twice to completely clear the FIFO.
2	UNDERRUN	R_CLR/SET	0	The USB sets this bit if an IN token is received when the TxPktRdy bit not set. The CPU should clear this bit.
1	FIFONOTEMPTY	R_CLR/SET	0	The USB sets this bit when there is at least 1 packet in the Tx FIFO.
0	TXPKTRDY	R_SET/CLR	0	The CPU sets this bit after loading a data packet into the FIFO. It is cleared automatically when a data packet has been transmitted. An interrupt is generated (if enabled) when the bit is cleared.

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	CLRDT	STALLED	STALL	FLUSH	UNDRUN	FIFONE	TXRDY

Offset: 0x0011

Bits	Name	Type	Reset	Description
7	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	*CLRDT*	R/W	0	Clear Data Toggle Writing a 1 to this bit to reset the endpoint data toggle to 0.
5	*STALLED*	R/W	0	Endpoint Stalled 0: A STALL handshake has not been transmitted. 1: A STALL handshake has been transmitted. The FIFO is flushed and the *TXRDY* is cleared.
4	*STALL*	R/W	0	Send Stall 0: No effect 1: Issues a STALL handshake to an IN token.
3	*FLUSH*	R/W	0	Flush FIFO This bit should only be set when the *TXRDY* bit is clear. 0: No effect 1: Flushes the latest packet from the endpoint transmit FIFO. The FIFO pointer is reset and the *TXRDY* bit is cleared.
2	*UNDRUN*	R/W	0	Underrun 0: No underrun. 1: An IN token has been received when *TXRDY* is not set.
1	*FIFONE*	R/W	0	FIFO Not Empty 0: The FIFO is empty. 1: At least one packet is in the transmit FIFO
0	*TXRDY*	R/W	0	Transmit Packet Ready This bit is cleared automatically when a data packet has been transmitted. 0: No transmit packet is ready. 1: Software sets this bit after loading a data packet into the Tx FIFO

4.20.3.2.4 USB_CSR2 - USB MAIN CONTROL STATUS REGISTER 2

The CSR02 register comprises a single self-clearing bit which may be used to flush the Endpoint 0 FIFO.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	RO	RO	RO	RO	RO	RO	R/W
							FLUSHFIFO

Offset: 0x0012

Bits	Name	Type	Reset	Description
7:1	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
0	*FLUSHFIFO*	SLF_CLR/R	0	The CPU writes a 1 to this bit to flush the next packet to be transmitted/read from the Endpoint 0 FIFO. The FIFO pointer is reset and the TxPktRdy/RxPktRdy bit (above) is cleared. Note: FlushFIFO has no effect unless TxPktRdy/RxPktRdy is set.

4.20.3.2.5 USB_TXCSR2 - USB TX CONTROL STATUS REGISTER 2

TxCSR2 is an 8-bit register that provides further control bits for transfers through the currently-selected Tx endpoint. There is a TxCSR2 register for each configured Tx endpoint (not including Endpoint 0).

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	RO	R/W	RO	RO	RO
AUTOSET	ISO	MODE		FRCDATATOG			

Offset: 0x0012

Bits	Name	Type	Reset	Description
7	*AUTOSET*	RW/R	0	If the CPU sets this bit, TxPktRdy will be automatically set when data of the maximum packet size (value in TxMaxP) is loaded into the Tx FIFO. If a packet of less than the maximum packet size is loaded, then TxPktRdy will have to be set manually.
6	*ISO*	RW/R	0	The CPU sets this bit to enable the Tx endpoint for Isochronous transfers, and clears it to enable the Tx endpoint for Bulk or Interrupt transfers. Note: This bit only has any effect in Peripheral mode. In Host mode, it always returns zero.
5	*MODE*	RW/R	1	The CPU sets this bit to enable the endpoint direction as Tx, and clears it to enable the endpoint direction as Rx. Note: This bit only has any effect where the same endpoint FIFO is used for both Tx and Rx transactions.
4	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
3	*FRCDATATOG*	RW/R	0	The CPU sets this bit to force the endpoint data toggle to switch and the data packet to be cleared from the FIFO, regardless of whether an ACK was received. This can be used by Interrupt IN endpoints that are used to communicate rate feedback for Isochronous endpoints.
2:0	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

4.20.3.2.6 USB_RXMAXP - USB RX MAXIMUM PACKET SIZE REGISTER

USB_RXMAXP is an 8-bit register that holds the maximum packet size for transactions through the currently-selected Rx endpoint - in units of 8 bytes, except that a value of 128 sets the maximum packet size to 1023 (the maximum size for an Isochronous packet transferred in a full-speed transaction) rather than 1024. In setting this value, you should note the constraints placed by the USB

Specification on packet sizes for Bulk, Interrupt and Isochronous transactions in full-speed operations. There is an RxMaxP register for each configured Rx endpoint (except Endpoint 0).

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
MAX_PACKET_SIZE/TRANSACTION							

Offset: 0x0013

Bits	Name	Type	Reset	Description
7:0	*MAX_PACKET_SIZE/TRANSACTION*	RW/R	0x0	The Rx Max Package for each configured Rx endpoint (except Endpoint 0).

The total amount of data represented by the value written to this register must not exceed the Rx FIFO size or, where dynamic FIFO sizing is used, the maximum packet size specified in theRxFIFO2 register. If the value written to this register is less than, or equal to, half the Rx FIFO size, two packets can be buffered (except where dynamic FIFO sizing has been selected).

4.20.3.2.7 USB_RXCSR1 - USB RX CONTROL STATUS REGISTER 1

RxCSR1 is an 8-bit register that provides control and status bits for transfers through the currently-selected Rx endpoint. There is an RxCSR1 register for each configured Rx endpoint (not including Endpoint 0).

The interpretation of the register depends on whether the USB is acting as a peripheral or as a host.

Host mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	DATAERROR	OVERRUN	FIFOFULL	RXPKTRDY

Offset: 0x0014

Bits	Name	Type	Reset	Description
7	CLRDATATOG	SET/R_CLR	0	The CPU writes a 1 to this bit to reset the endpoint data toggle to 0.
6	SENTSTALL	R_CLR/SET	0	When a STALL handshake is received, this bit is set and an interrupt is generated. The CPU should clear this bit.
5	SENDSTALL	RW/R	0	The CPU writes a 1 to this bit to request an IN transaction. It is cleared when RxPktRdy is set.
4	FLUSHFIFO	SET/R	0	The CPU writes a 1 to this bit to flush the next packet to be read from the endpoint Rx FIFO. The FIFO pointer is reset and the RxPktRdy bit (below) is cleared. Note: FlushFIFO has no effect unless RxPktRdy is set. Also note that, if the FIFO is double-buffered, FlushFIFO may need to be set twice to completely clear the FIFO.
3	DATAERROR	R/SET	0	When operating in ISO mode, this bit is set when RxPktRdy is set if the data packet has a CRC or bit-stuff error and cleared when RxPktRdy is cleared. In Bulk mode, this bit will be set when the Rx endpoint is halted following the receipt of NAK responses for longer than the time set as the NAK Limit by the RxInterval register. The CPU should clear this bit to allow the endpoint to continue.
2	OVERRUN	R_CLR/SET	0	The USB sets this bit when 3 attempts have been made to receive a packet and no data packet has been received. The CPU should clear this bit. An interrupt is generated when the bit is set. Note: This bit is only valid when the IN endpoint is operating in Bulk or Interrupt mode. In ISO mode, it always returns zero.
1	FIFOFULL	R/SET	0	This bit is set when no more packets can be loaded into the Rx FIFO.
0	RXPKTRDY	R_CLR/SET	0	This bit is set when a data packet has been received. The CPU should clear this bit when the packet has been unloaded from the Rx FIFO. An interrupt is generated when the bit is set.

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	RO	R/W	RO	R/W
CLRDT	STALLED	STALL	FLUSH	DATAERR	OVERRUN	FULL	RXRDY

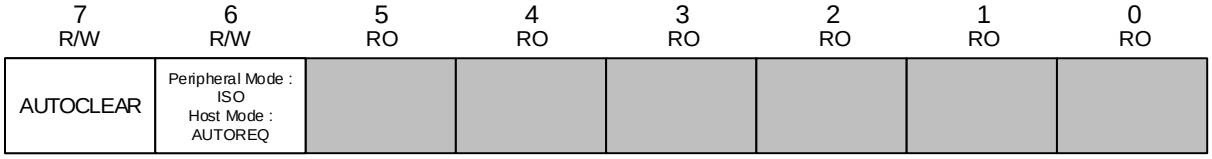
Offset: 0x0014

Bits	Name	Type	Reset	Description
7	CLRDT	R/W	0	Clear Data Toggle Writing a 1 to this bit to reset the endpoint data toggle to 0.
6	STALLED	R/W	0	Endpoint Stalled 0: A STALL handshake has not been transmitted. 1: A STALL handshake has been transmitted.
5	STALL	R/W	0	Send STALL 0: No effect 1: Issues a STALL handshake
4	FLUSH	R/W	0	Flush FIFO This bit should only be set when the *RXRDY* bit is set. 0: No effect 1: Flushes the next packet from the endpoint receive FIFO. The FIFO pointer is reset and the *RXRDY* bit is cleared.
3	DATAERR	RO	0	Data Error 0: Normal operation 1: Indicates the *RXRDY* is set and the data packet has a CRC or bit-stuff error.
2	OVERRUN	R/W	0	Overrun 0: No overrun error. 1: Indicates that an OUT packet cannot be loaded into the receive FIFO.
1	FULL	RO	0	FIFO Full 0: The receive FIFO is not full. 1: No more packets can be loaded into the receive FIFO.
0	RXRDY	R/W	0	Receive Packet Ready 0: No data packet has been received. 1: A data packet has been received.

4.20.3.2.8 USB_RXCSR2 - USB RX CONTROL STATUS REGISTER 2

RxCSR2 is an 8-bit register that provides further control bits for transfers through the currently-selected Rx endpoint. There is an RxCSR2 register for each configured Rx endpoint (not including Endpoint 0).

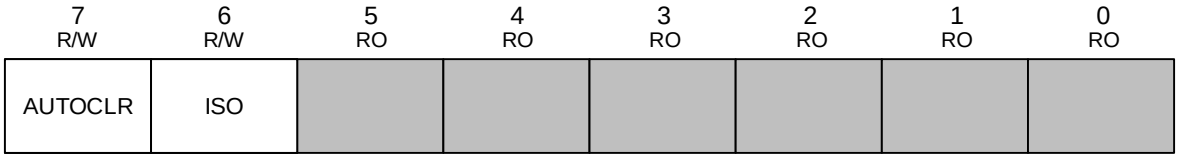
Host mode:



Offset: 0x0015

Bits	Name	Type	Reset	Description
7	AUTOCLEAR	SET/R_CLR	0	If the CPU sets this bit then the RxPktRdy bit will be automatically cleared when a packet of RxMaxP bytes has been unloaded from the Rx FIFO. When packets of less than the maximum packet size are unloaded, RxPktRdy will have to be cleared manually.
6	Peripheral Mode : ISO Host Mode : AUTOREQ	R_CLR/SET	0	In Peripheral mode, the CPU sets this bit to enable the Rx endpoint for Isochronous transfers, and clears it to enable the Rx endpoint for Bulk or Interrupt transfers. If the CPU sets this bit in Host mode, the ReqPkt bit will be automatically set when the RxPktRdy bit is cleared.
5:0	reserved	R/R	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

Peripheral mode:



Offset: 0x0015

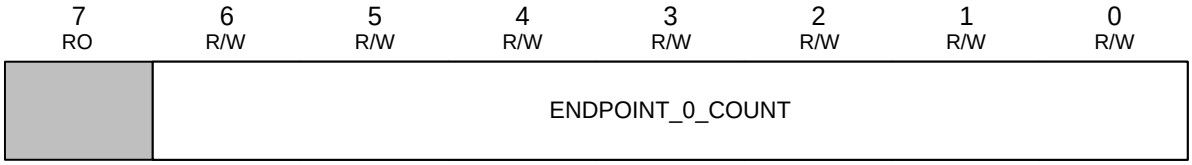
Bits	Name	Type	Reset	Description
7	AUTOCLR	R/W	0	Auto Clear 0: No effect 1: Enables the *RXRDY* bit to be automatically cleared when a packet of **USB_RXMAXP** bytes has been unloaded from the receive FIFO. When packets of less than the maximum packet size are unloaded, *RXRDY* must be cleared manually.
6	ISO	R/W	0	Isochronous Transfers 0: Enables the receive endpoint for isochronous transfers. 1: Enables the receive endpoint for bulk/interrupt transfers.
5:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

4.20.3.2.9 USB_COUNT - USB NUMBER OF RECEIVED BYTES REGISTER

Count0 is a 7-bit read-only register that indicates the number of received data bytes in the Endpoint 0 FIFO. The value returned is valid while **USB_CSR1**.*RXRDY* is set.

Host mode:

Peripheral mode:



Offset: 0x0016

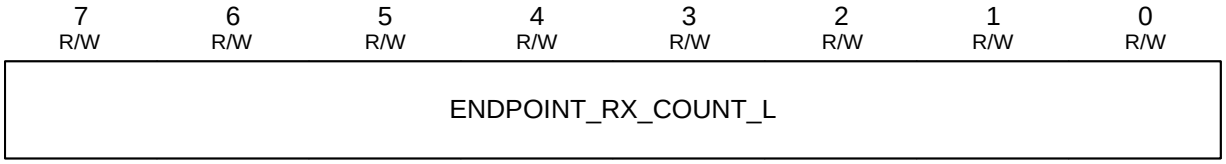
Bits	Name	Type	Reset	Description
7	reserved	N/A	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6:0	ENDPOINT_0_COUNT	R/W	0x0	That indicates the number of received data bytes in the Endpoint 0 FIFO. The value returned is valid while RxPktRdy (CSR0.D0) is set.

4.20.3.2.10 USB_RXCOUNT1 - USB RX NUMBER OF RECEIVED BYTES REGISTER 1

RxCount1 is an 8-bit read-only register that holds the lower 8 bits of the number of received data bytes in the packet in the FIFO associated with the currently-selected Rx endpoint.

Host mode:

Peripheral mode:



Offset: 0x0016

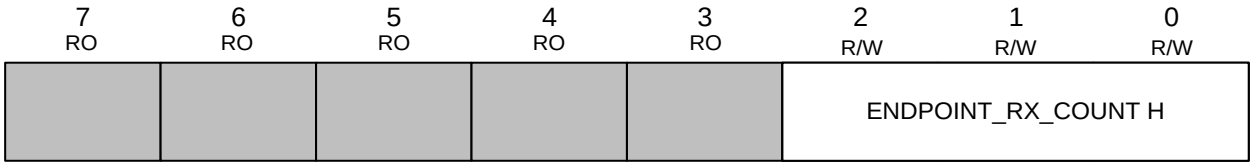
Bits	Name	Type	Reset	Description
7:0	ENDPOINT_RX_COUNT_L	R/W	0x0	Endpoint Rx Count - lower 8 bits

4.20.3.2.11 USB_RXCOUNT2 - USB RX NUMBER OF RECEIVED BYTES REGISTER 2

RxCount2 is a 3-bit read-only register that holds the upper 3 bits of the number of received data bytes in the packet in the FIFO associated with the currently-selected Rx endpoint. The value returned is valid while **USB_RXCSR1**.*RXRDY* is set.

Host mode:

Peripheral mode:



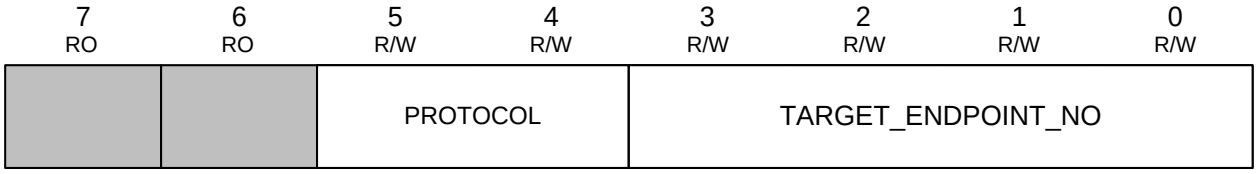
Offset: 0x0017

Bits	Name	Type	Reset	Description
7:3	reserved	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
2:0	ENDPOINT_RX_COUNT H	R/W	0x0	Endpoint Rx Count - upper 3 bits

4.20.3.2.12 USB_TXTYPE - USB TX PROTOCOL TYPE REGISTER

TxType is a 6-bit register that should be written with the endpoint number to be targeted by the endpoint in the lower 4 bits, and the transaction protocol to use for the currently-selected Tx endpoint in the upper 2 bits. There is a TxType register for each configured Tx endpoint (except Endpoint 0).

Host mode only:



Offset: 0x0018

Bits	Name	Type	Reset	Description
7:6	reserved	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
5:4	PROTOCOL	R/W	0x0	The CPU should set this to select the required protocol for the Tx endpoint: 00: Illegal. 01: Isochronous. 10: Bulk. 11: Interrupt.
3:0	TARGET_ENDPOINT_NO	R/W	0x0	The CPU should set this value to the endpoint number contained in the Tx endpoint descriptor returned to the USB during device enumeration.

4.20.3.2.13 USB_NAKLIMIT - USB NAK RESPONSE TIMEOUT REGISTER

USB_NAKLIMIT is an 8-bit register that sets the number of frames after which Endpoint 0 should timeout on receiving a stream of NAK responses. (Equivalent settings for other endpoints can be made through **USB_TXINTERVAL** and **USB_RXINTERVAL** registers.)

The number of frames selected may be between 2 and 255. If the host receives NAK responses from the target for more frames than the number represented by the Limit set in this register, the endpoint will be halted.

Note: A value of 0 or 1 disables the NAK timeout function.

Host mode only:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ENDPOINT_0_NAK_LIMIT							

Offset: 0x0019

Bits	Name	Type	Reset	Description
7:0	ENDPOINT_0_NAK_LIMIT	RW/R	0x0	That sets the number of frames after which Endpoint 0 should timeout on receiving a stream of NAK responses.

4.20.3.2.14 USB_TXINTERVAL - USB TX POLLING INTERVAL REGISTER

TxInterval is an 8-bit register that, for Interrupt and Isochronous transfers, defines the polling interval for the currently-selected Tx endpoint. For Bulk endpoints, this register sets the number of frames after which the endpoint should timeout on receiving a stream of NAK responses. There is a TxInterval register for each configured Tx endpoint (except Endpoint 0).

Host mode only:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
TX_POLLING_INTERVAL / NAK LIMIT (N)							

Offset: 0x0019

Bits	Name	Type	Reset	Description
7:0	TXPOLLING_INTERVAL / NAK LIMIT(N)	RW/R	0x0	The polling interval for the currently-selected Tx endpoint.

In each case the value that is set defines a number of frames, as follows:

Transfer Type	Valid values (n)	Interpretation
Interrupt or Isochronous	1-255	Polling interval is *n* frames.
Bulk	2-255	NAK Limit is *n* frames. *Note: A value of 0 or 1 disables the NAK timeout function.*

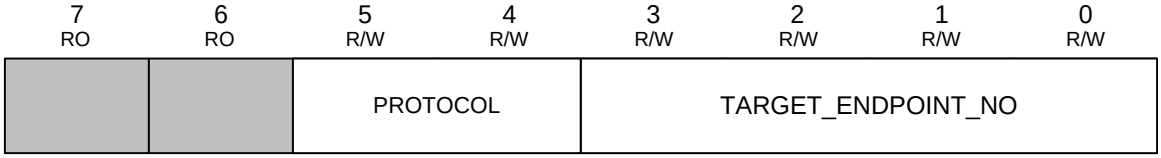
Note: The USB Specification allows polling intervals greater than 255ms for an isochronous endpoints. Where required, these must be implemented in software.

4.20.3.2.15 USB_RXTYPE - USB RX PROTOCOL TYPE REGISTER 142

RxType is a 6-bit register that should be written with the endpoint number to be targeted by the endpoint in the lower 4 bits, and the transaction protocol to use for the currently-selected Rx endpoint in the upper 2 bits.

There is an RxType register for each configured Rx endpoint (except Endpoint 0).

Host mode only:



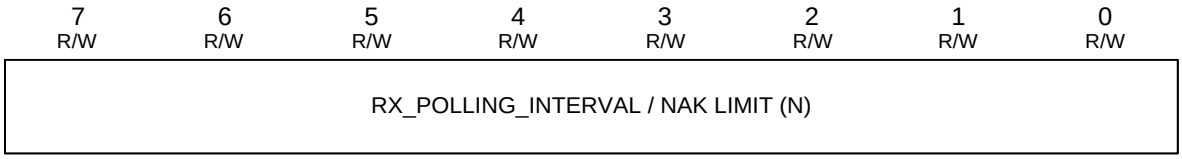
Offset: 0x001A

Bits	Name	Type	Reset	Description
7:6	*reserved*	N/A	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
5:4	PROTOCOL	RW/R	0x0	The CPU should set this to select the required protocol for the Rx endpoint : 00: Illegal. 01: Isochronous. 10: Bulk. 11: Interrupt.
3:0	TARGET_ ENDPOINT_NO	RW/R	0x0	The CPU should set this value to the endpoint number contained in the Rx endpoint descriptor returned to the USB during device enumeration.

4.20.3.2.16 USB_RXINTERVAL - USB RX POLLING INTERVAL REGISTER

RxInterval is an 8-bit register that, for Interrupt and Isochronous transfers, defines the polling interval for the currently-selected Rx endpoint. For Bulk endpoints, this register sets the number of frames after which the endpoint should timeout on receiving a stream of NAK responses. There is an RxInterval register for each configured Rx endpoint (except Endpoint 0).

Host mode only:



Offset: 0x001B

Bits	Name	Type	Reset	Description
7:0	RX_POLLING_INTERVAL / NAK LIMIT (N)	RW/R	0x0	The polling interval for the currently-selected Rx endpoint.

In each case the value that is set defines a number of frames, as follows:

Transfer Type	Valid values (n)	Interpretation
Interrupt or Isochronous	1-255	Polling interval is *n* frames.
Bulk	2-255	NAK Limit is *n* frames. *Note: A value of 0 or 1 disables the NAK timeout function.*

Note: The USB Specification allows polling intervals greater than 255ms for an isochronous endpoints. Where required, these must be implemented in software.

4.20.3.2.17 USB_TXFIFO1 - USB TX FIFO1 CONFIGURE REGISTER

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
AD[7:0]							

Offset: 0x001C

Bits	Name	Type	Reset	Description
7:0	AD[7:0]	RW/R	0x0	Start address of the Tx endpoint FIFO - low 8 bits. In units of 8 bytes as follows:

ADDRH+ADDRL	Start Address
000	0x0000
001	0x0008
002	0x0010
…	…
FFF	0x7FF8

4.20.3.2.18 USB_TXFIFO2 - USB TX FIFO2 CONFIGURE REGISTER

TxFIFO2 are 8-bit registers which together control the start address and the size of the selected Tx endpoint FIFO.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SZ[2:0]			DPB	AD[11:8]			

Offset: 0x001D

Bits	Name	Type	Reset	Description
7:5	SZ[2:0]	RW/R	0x0	Maximum packet Size SIZE Packet Size(Bytes) 0x0 8 0x1 16 0x2 32 0x3 64 0x4 128 0x5 256 0x6 512 0x7 1024 If DPB = 0, the FIFO will also be this size; if DPB = 1, the FIFO will be twice this size
4	DPB	RW/R	0	Double Packet Buffer Support 0: Only single-packet buffering is supported. 1: Double-packet buffering is supported.
3:0	AD[11:8]	RW/R	0x0	Start address of the Tx endpoint FIFO - high 4 bits

4.20.3.2.19 USB_RXFIFO1 - USB RX FIFO1 CONFIGURE REGISTER

RxFIFO1 are 8-bit registers which together control the start address and the size of the selected Rx endpoint FIFO.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
AD[7:0]							

Offset: 0x001E

Bits	Name	Type	Reset	Description
7:0	AD[7:0]	RW/R	0x0	Start address of the Rx endpoint FIFO - low 8 bits. In units of 8 bytes as follows:

ADDRH+ADDRL	Start Address
000	0x0000
001	0x0008
002	0x0010
…	…
FFF	0x7FF8

4.20.3.2.20 USB_RXFIFO2 - USB RX FIFO2 CONFIGURE REGISTER

RxFIFO2 are 8-bit registers which together control the start address and the size of the selected Rx endpoint FIFO.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SZ[2:0]			DPB	AD[11:8]			

Offset: 0x001F

Bits	Name	Type	Reset	Description
7:5	*SZ[2:0]*	RW/R	0x0	Maximum packet Size SZ[2:0] Packet Size(Bytes) SZ[2:0] Packet Size(Bytes) 0x0 8 0x1 16 0x2 32 0x3 64 0x4 128 0x5 256 0x6 512 0x7 1024 If DPB = 0, the FIFO will also be this size; if DPB = 1, the FIFO will be twice this size
4	*DPB*	R/W	0	Double Packet Buffer Support 0: Only single-packet buffering is supported. 1: Double-packet buffering is supported.
3:0	*AD[11:8]*	R/W	0x0	Start address Start address of the endpoint FIFO - high 8 bits

4.20.3.3 FIFO REGISTERS

4.20.3.3.1 USB_EP0FIFO - USB ENDPOINT 0 FIFO

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
FIFO0							

Offset: 0x0020

Bits	Name	Type	Reset	Description
7:0	*FIFO0*	R/W	0x0	Endpoint 0 FIFO

4.20.3.3.2 USB_EP0FIFO - USB ENDPOINT 0 FIFO

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
FIFO1							

Offset: 0x0024

Bits	Name	Type	Reset	Description
7:0	*FIFO0*	R/W	0x0	Endpoint 1 FIFO

4.20.3.3.3 USB_EP2FIFO - USB ENDPOINT 2 FIFO

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
FIFO2							

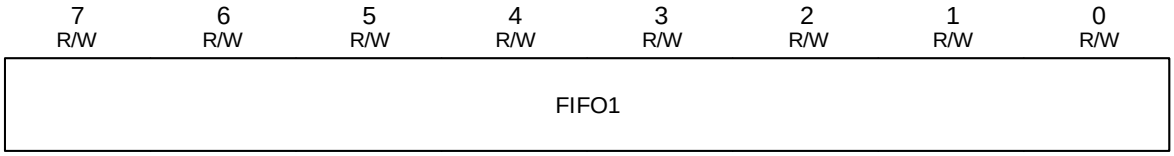
Offset: 0x0028

Bits	Name	Type	Reset	Description
7:0	*FIFO2*	R/W	0x0	Endpoint 2FIFO

4.20.3.3.4 USB_EP3FIFO - USB ENDPOINT 3 FIFO

Host mode:

Peripheral mode:



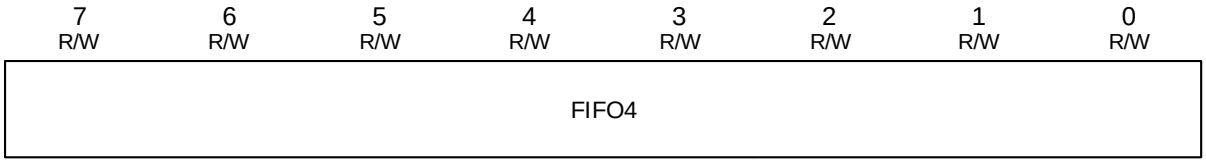
Offset: 0x002c

Bits	Name	Type	Reset	Description
7:0	*FIFO3*	R/W	0x0	Endpoint 3 FIFO

4.20.3.3.5 USB_EP4FIFO - USB ENDPOINT 4 FIFO

Host mode:

Peripheral mode:



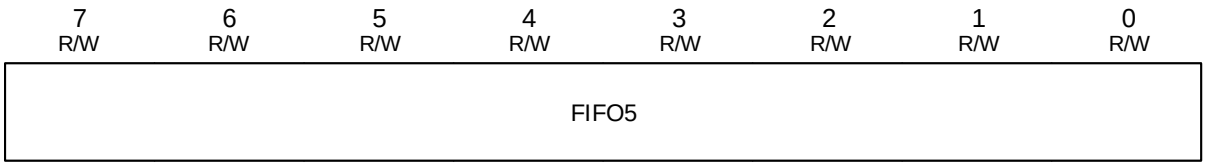
Offset: 0x0030

Bits	Name	Type	Reset	Description
7:0	*FIFO4*	R/W	0x0	Endpoint 4 FIFO

4.20.3.3.6 USB_EP5FIFO - USB ENDPOINT 5 FIFO

Host mode:

Peripheral mode:



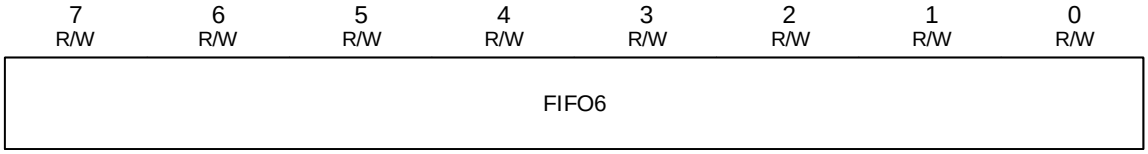
Offset: 0x0034

Bits	Name	Type	Reset	Description
7:0	*FIFO5*	R/W	0x0	Endpoint 5 FIFO

4.20.3.3.7 USB_EP6FIFO - USB ENDPOINT 6 FIFO

Host mode:

Peripheral mode:



Offset: 0x0038

Bits	Name	Type	Reset	Description
7:0	*FIFO6*	R/W	0x0	Endpoint 6 FIFO

4.20.3.4 INTERRUPTS REGISTERS

The interrupt in USB are controlled by a set of five registers.

- Interrupt Control (IER, IDR, IMR)

  USB Tx/Rx Interrupt Enable Register enables the interrupt request lines by writing a '1'. Similarly, **USB Tx/Rx Interrupt Disable Register** disables the interrupt request lines by writing a '1'. IER and IDR are write only registers which control the masking of interrupts. The overall result of these two registers can be shown by **USB Tx/Rx Interrupt Mask Register**. IMR is a read-only register using '1' or '0' to indicate if the interrupt request line is enabled/ or disabled.

- Interrupt Status Read (RIS)

  USB Tx/Rx Raw Interrupt Status is a read-only register to read all interrupt status of the module.

- Interrupt Clear (ISC)

  USB Tx/Rx Interrupt Status & Interrupt Clear Register is used to indicate the non-masked interrupt status of the module, since only now-masked interrupts are asserted to processor. Writing a '1' to the bit in this register can clear the corresponding interrupt status or disable the interrupt by writing 1 to IDR.

4.20.3.4.1 USB_TXIER - USB TX INTERRUPT ENABLE REGISTER

TX Endpoint 0 - 6 Interrupt Enable is a 7 bits register, set the bit will enable the TX Endpoint's interrupt.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	WO	WO	WO	WO	WO	WO	WO
	EP6TXIER	EP5TXIER	EP4TXIER	EP3TXIER	EP2TXIER	EP1TXIER	EP0TXIER

Offset: 0x0080

Bits	Name	Type	Reset	Description
7	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	EP6TXIER	WO	0	TX Endpoint 6 Interrupt Enable. 1: Interrupt Enable
5	EP5TXIER	WO	0	TX Endpoint 5 Interrupt Enable. 1: Interrupt Enable
4	EP4TXIER	WO	0	TX Endpoint 4 Interrupt Enable. 1: Interrupt Enable
3	EP3TXIER	WO	0	TX Endpoint 3 Interrupt Enable. 1: Interrupt Enable
2	EP2TXIER	WO	0	TX Endpoint 2 Interrupt Enable. 1: Interrupt Enable
1	EP1TXIER	WO	0	TX Endpoint 1 Interrupt Enable. 1: Interrupt Enable
0	EP0TXIER	WO	0	TX and RX Endpoint 0 Interrupt Enable. 1: Interrupt Enable

4.20.3.4.2 USB_RXIER - USB RX INTERRUPT ENABLE REGISTER

RX Endpoint 1-6 Interrupt Enable is a 6 bits register, set the bit will enable the RX Endpoint's interrupt.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	WO	WO	WO	WO	WO	WO	RO
	EP6RXIER	EP5RXIER	EP4RXIER	EP3RXIER	EP2RXIER	EP1RXIER	

Offset: 0x0082

Bits	Name	Type	Reset	Description
7	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	*EP6RXIER*	WO	0	RX Endpoint 6 Interrupt Enable. 1: Interrupt is enabled.
5	*EP5RXIER*	WO	0	RX Endpoint 5 Interrupt Enable. 1: Interrupt is enabled.
4	*EP4RXIER*	WO	0	RX Endpoint 4 Interrupt Enable. 1: Interrupt is enabled.
3	*EP3RXIER*	WO	0	RX Endpoint 3 Interrupt Enable. 1: Interrupt is enabled.
2	*EP2RXIER*	WO	0	RX Endpoint 2 Interrupt Enable. 1: Interrupt is enabled.
1	*EP1RXIER*	WO	0	RX Endpoint 1 Interrupt Enable. 1: Interrupt is enabled.
0	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

4.20.3.4.3 USB_TXIDR - USB RX INTERRUPT DISABLE REGISTER

TX Endpoint 0-6 Interrupt Disable is a 7 bits register, set the bit will disable the TX Endpoint's interrupt.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	WO	WO	WO	WO	WO	WO	R/W
	EP6TXIDR	EP5TXIDR	EP4TXIDR	EP3TXIDR	EP2TXIDR	EP1TXIDR	EP0TXIDR

Offset: 0x0084

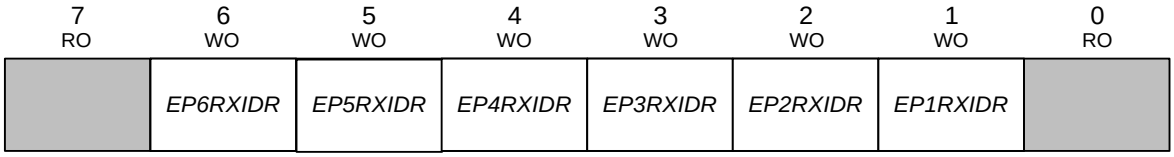
Bits	Name	Type	Reset	Description
7	Reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	EP6TXIDR	WO	0	TX Endpoint 6 Interrupt Disable. 1: Interrupt is disabled.
5	EP5TXIDR	WO	0	TX Endpoint 5 Interrupt Disable. 1: Interrupt is disabled.
4	EP4TXIDR	WO	0	TX Endpoint 4 Interrupt Disable. 1: Interrupt is disabled.
3	EP3TXIDR	WO	0	TX Endpoint 3 Interrupt Disable. 1: Interrupt is disabled.
2	EP2TXIDR	WO	0	TX Endpoint 2 Interrupt Disable. 1: Interrupt is disabled.
1	EP1TXIDR	WO	0	TX Endpoint 1 Interrupt Disable. 1: Interrupt is disabled.
0	EP0TXIDR	R/W	0	TX and RX Endpoint 0 Interrupt Disable. 1: Interrupt is disabled.

4.20.3.4.4 USB_RXIDR - USB RX INTERRUPT DISABLE REGISTER

RX Endpoint 1-6 Interrupt Disable is a 6 bits register, set the bit will disable the RX Endpoint's interrupt.

Host mode:

Peripheral mode:



Offset: 0x0086

Bits	Name	Type	Reset	Description
7	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	*EP6RXIDR*	WO	0	RX Endpoint 6 Interrupt Disable. 1: Interrupt is disabled.
5	*EP5RXIDR*	WO	0	RX Endpoint 5 Interrupt Disable. 1: Interrupt is disabled.
4	*EP4RXIDR*	WO	0	RX Endpoint 4 Interrupt Disable. 1: Interrupt is disabled.
3	*EP3RXIDR*	WO	0	RX Endpoint 3 Interrupt Disable. 1: Interrupt is disabled.
2	*EP2RXIDR*	WO	0	RX Endpoint 2 Interrupt Disable. 1: Interrupt is disabled.
1	*EP1RXIDR*	WO	0	RX Endpoint 1 Interrupt Disable. 1: Interrupt is disabled.
0	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

4.20.3.4.5 USB_TXIMR - USB TX INTERRUPT MASK STATUS REGISTER

TX Endpoint 0-6 Interrupt Mask is a 7 bits register, those bit will show the TX Endpoint's interrupt mask status.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	EP6TXIMR	EP5TXIMR	EP4TXIMR	EP3TXIMR	EP2TXIMR	EP1TXIMR	EP0TXIMR

Offset: 0x0088

Bits	Name	Type	Reset	Description
7	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	EP6TXIMR	R/W	0	TX Endpoint 6 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
5	EP5TXIMR	R/W	0	TX Endpoint 5 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
4	EP4TXIMR	R/W	0	TX Endpoint 4 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
3	EP3TXIMR	R/W	0	TX Endpoint 3 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
2	EP2TXIMR	R/W	0	TX Endpoint 2 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
1	EP1TXIMR	R/W	0	TX Endpoint 1 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
0	EP0TXIMR	R/W	0	TX and RX Endpoint 0 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked

4.20.3.4.6 USB_RXIMR - USB RX INTERRUPT MASK REGISTER

RX Endpoint 1-6 Interrupt Mask is a 6 bits register, those bit will show the RX Endpoint's interrupt mask status.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W	R/W	R/W	R/W	R/W	R/W	RO
	EP6RXIMR	EP5RXIMR	EP4RXIMR	EP3RXIMR	EP2RXIMR	EP1RXIMR	

Offset: 0x008A

Bits	Name	Type	Reset	Description
7	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	EP6RXIMR	R/W	0	RX Endpoint 6 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
5	EP5RXIMR	R/W	0	RX Endpoint 5 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
4	EP4RXIMR	R/W	0	RX Endpoint 4 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
3	EP3RXIMR	R/W	0	RX Endpoint 3 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
2	EP2RXIMR	R/W	0	RX Endpoint 2 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
1	EP1RXIMR	R/W	0	RX Endpoint 1 Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
0	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

4.20.3.4.7 USB_TXRIS - USB TX RAW INTERRUPT STATUS REGISTER

TX Endpoint 0-6 Raw Interrupt Status is a 7 bits register, those bit will show the TX Endpoint's raw interrupt status.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	EP6TXRIS	EP5TXRIS	EP4TXRIS	EP3TXRIS	EP2TXRIS	EP1TXRIS	EP0TXRIS

Offset: 0x008C

Bits	Name	Type	Reset	Description
7	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	EP6TXRIS	R/W	0	TX Endpoint 6 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
5	EP5TXRIS	R/W	0	TX Endpoint 5 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
4	EP4TXRIS	R/W	0	TX Endpoint 4 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
3	EP3TXRIS	R/W	0	TX Endpoint 3 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
2	EP2TXRIS	R/W	0	TX Endpoint 2 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
1	EP6TXRIS	R/W	0	TX Endpoint 1 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
0	EP0TXRIS	R/W	0	TX and RX Endpoint 0 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.20.3.4.8 USB_RXRIS - USB RX RAW INTERRUPT STATUS REGISTER

RX Endpoint 1-6 Raw Interrupt Status is a 6 bits register, those bit will show the RX Endpoint's raw interrupt status.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W	R/W	R/W	R/W	R/W	R/W	RO
	EP6RXRIS	EP5RXRIS	EP4RXRIS	EP3RXRIS	EP2RXRIS	EP1RXRIS	

Offset: 0x008E

Bits	Name	Type	Reset	Description
7	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	EP6RXRIS	R/W	0	RX Endpoint 6 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
5	EP5RXRIS	R/W	0	RX Endpoint 5 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
4	EP4RXRIS	R/W	0	RX Endpoint 4 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
3	EP3RXRIS	R/W	0	RX Endpoint 3 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
2	EP2RXRIS	R/W	0	RX Endpoint 2 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
1	EP1RXRIS	R/W	0	RX Endpoint 1 Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

4.20.3.4.9 USB_TXISC - USB TX INTERRUPT STATUS AND CLEAR REGISTER

TX Endpoint 0-6 Interrupt Status and Clear is a 7 bits register, those bit will show the TX Endpoint's interrupt status. Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
	EP6TXISC	EP5TXISC	EP4TXISC	EP3TXISC	EP2TXISC	EP1TXISC	EP0TXISC

Offset: 0x0090

Bits	Name	Type	Reset	Description
7	reserved	ROx	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	EP6TXISC	R/W1C	0	TX Endpoint 6 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled. Write '1' to clear this status
5	EP5TXISC	R/W1C	0	TX Endpoint 5 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled. Write '1' to clear this status
4	EP4TXISC	R/W1C	0	TX Endpoint 4 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled. Write '1' to clear this status
3	EP3TXISC	R/W1C	0	TX Endpoint 3 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled. Write '1' to clear this status
2	EP2TXISC	R/W1C	0	TX Endpoint 2 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled. Write '1' to clear this status
1	EP1TXISC	R/W1C	0	TX Endpoint 1 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled. Write '1' to clear this status
0	EP0TXISC	R/W1C	0	TX and RX Endpoint 0 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled. Write '1' to clear this status

4.20.3.4.10 USB_RXISC - USB RX INTERRUPT STATUS AND CLEAR REGISTER

RX Endpoint 1-6 Interrupt Status and Clear is a 6 bits register, those bit will show the RX Endpoint's interrupt status. Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.

Host mode:

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W1C	RO	R/W1C	R/W1C	R/W1C	R/W1C	RO
	EP6RXISC	EP5RXISC	EP4RXISC	EP3RXISC	EP2RXISC	EP1RXISC	

Offset: 0x0092

Bits	Name	Type	Reset	Description
7	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	EP6RXISC	R/W1C	0	RX Endpoint 6 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: An unmasked interrupt has occurred Write '1' to clear this status
5	EP5RXISC	R/W1C	0	RX Endpoint 5 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: An unmasked interrupt has occurred Write '1' to clear this status
4	EP4RXISC	R/W1C	0	RX Endpoint 4 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: An unmasked interrupt has occurred Write '1' to clear this status
3	EP3RXISC	R/W1C	0	RX Endpoint 3 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: An unmasked interrupt has occurred Write '1' to clear this status
2	EP2RXISC	R/W1C	0	RX Endpoint 2 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: An unmasked interrupt has occurred Write '1' to clear this status
1	EP1RXISC	R/W1C	0	RX Endpoint 1 Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: An unmasked interrupt has occurred Write '1' to clear this status
0	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read

4.20.3.4.11 USB_IER - USB INTERRUPT ENABLE REGISTER

Host mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
VBUS_ERROR_IER	SESS_REQ_IER	DISCON_IER	CONN_IER	SOF_IER	Peripheral mode : RESET_IER Host mode : BABBLE_IER	RESUME_IER	SUSPEND_IER

Offset: 0x00A0

Bits	Name	Type	Reset	Description
7	VBUS_ERROR_IER	R/W	0	VBus Error Interrupt Enable. 1: Interrupt Enable
6	SESS_REQ_IER	R/W	0	Session Request Interrupt Enable. 1: Interrupt Enable
5	DISCON_IER	R/W	0	Device Disconnet Interrupt Enable. 1: Interrupt Enable
4	CONN_IER	R/W	0	Device Connect Interrupt Enable. 1: Interrupt Enable
3	SOF_IER	R/W	0	Start of new Frame Interrupt Enable. 1: Interrupt Enable
2	Peripheral mode : RESET_IER Host mode : BABBLE_IER	R/W	0	Peripheral mode : Reset Interrupt Enable. 1: Interrupt Enable Host mode : Babble Interrupt Enable. 1: Interrupt Enable
1	RESUME_IER	R/W	0	Resume Interrupt Enable. 1: Interrupt Enable
0	SUSPEND_IER	R/W	0	Suspend Interrupt Enable. 1: Interrupt Enable

Peripheral mode:

7 RO	6 R/W	5 R/W	4 RO	3 R/W	2 R/W	1 R/W	0 R/W
	SESREQIE	DISCONIE		SOFIE	RESTIE	RESIE	SUSPDIE

Offset: 0x00A0

Bits	Name	Type	Reset	Description
7	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	*SESREQIE*	R/W	0	Session Request Interrupt Enable 1: The Session Request interrupt is enabled.
5	*DISCONIE*	R/W	0	Disconnect Interrupt Enable 1: The disconnect interrupt is enabled.
4	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
3	*SOFIE*	R/W	0	Start-of-Frame Interrupt Enable 1: The start-of-frame interrupt is enabled.
2	*RESTIE*	R/W	0	Reset Interrupt Enable 1: The reset interrupt is enabled.
1	*RESIE*	R/W	0	Resume Interrupt Enable 1: The resume interrupt is enabled.
0	*SUSPDIE*	R/W	0	Suspend Interrupt Enable 1: The suspend interrupt is enabled.

4.20.3.4.12 USB_IDR - USB INTERRUPT DISABLE REGISTER

Host mode:

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
VBUS_ERROR_IDR	SESS_REQ_IDR	DISCON_IDR	CONN_IDR	SOF_IDR	Peripheral mode : RESET_IDR Host mode : BABBLE_IDR	RESUME_IDR	SUSPEND_IDR

Offset: 0x00A4

Bits	Name	Type	Reset	Description
7	VBUS_ERROR_IDR	R/W	0	VBus Error Interrupt Disable. 1: Interrupt Disable
6	SESS_REQ_IDR	R/W	0	Session Request Interrupt Disable. 1: Interrupt Disable
5	DISCON_IDR	R/W	0	Device Disconnet Interrupt Disable. 1: Interrupt Disable
4	CONN_IDR	R/W	0	Device Connect Interrupt Disable. 1: Interrupt Disable
3	SOF_IDR	R/W	0	Start of new Frame Interrupt Disable. 1: Interrupt Disable
2	Peripheral mode : RESET_IDR Host mode : BABBLE_IDR	R/W	0	Peripheral mode : Reset Interrupt Disable. 1: Interrupt Disable Host mode : Babble Interrupt Disable. 1: Interrupt Disable
1	RESUME_IDR	R/W	0	Resume Interrupt Disable. 1: Interrupt Disable
0	SUSPEND_IDR	R/W	0	Suspend Interrupt Disable. 1: Interrupt Disable

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W	R/W	RO	R/W	R/W	R/W	R/W
	SESREQID	DISCONID		SOFID	RESTID	RESID	SUSPDID

Offset: 0x00A4

Bits	Name	Type	Reset	Description
7	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	*SESREQID*	R/W	0	Session Request Interrupt Disable 1: The Session Request interrupt is disabled.
5	*DISCONID*	R/W	0	Disconnect Interrupt Disable 1: The disconnect interrupt is disabled.
4	*reserved*	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
3	*SOFID*	R/W	0	Start-of-Frame Interrupt Disable 1: The start-of-frame interrupt is disabled.
2	*RESTID*	R/W	0	Reset Interrupt Disable 1: The reset interrupt is disabled.
1	*RESID*	R/W	0	Resume Interrupt Disable 1: The resume interrupt is disabled.
0	*SUSPDID*	R/W	0	Suspend Interrupt Disable 1: The suspend interrupt is disabled.

4.20.3.4.13 USB_IMR - USB INTERRUPT MASK REGISTER

Host mode:

7 R/W	6 R/W	5 R/W	4 R/W	3 R/W	2 R/W	1 R/W	0 R/W
VBUS_ERROR_IMR	SESS_REQ_IMR	DISCON_IMR	CONN_IMR	SOF_IMR	Peripheral mode : RESET_IMR Host mode : BABBLE_IMR	RESUME_IMR	SUSPEND_IMR

Offset: 0x00A8

Bits	Name	Type	Reset	Description
7	VBUS_ERROR_IMR	R/W	0	VBus Error Interrupt Mask Status. 0: Interrupt Disable 1: Interrupt Enable
6	SESS_REQ_IMR	R/W	0	Sess Req Interrupt Mask Status. 0: Interrupt Disable 1: Interrupt Enable.
5	DISCON_IMR	R/W	0	Device Disconnet Interrupt Mask Status. 0: Interrupt Disable 1: Interrupt Enable.
4	CONN_IMR	R/W	0	Device Connect Interrupt Mask Status. 0: Interrupt Disable 1: Interrupt Enable.
3	SOF_IMR	R/W	0	Start of new Frame Interrupt Mask Status. 0: Interrupt Disable 1: Interrupt Enable.
2	Peripheral mode : RESET_IMR Host mode : BABBLE_IMR	R/W	1	Peripheral mode : Reset Interrupt Mask. 0: Interrupt Disable 1: Interrupt Enable. Host mode : Babble Interrupt Mask. 0: Interrupt Disable 1: Interrupt Enable.
1	RESUME_IMR	R/W	1	Resume Interrupt Mask Status. 0: Interrupt Disable 1: Interrupt Enable.
0	SUSPEND_IMR	R/W	0	Suspend Interrupt Mask Status. 0: Interrupt Disable 1: Interrupt Enable.

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W	R/W	RO	R/W	R/W	R/W	R/W
	SESREQIM	DISCONIM		SOFIM	RESTIM	RESIM	SUSPDIM

Offset: 0x00A8

Bits	Name	Type	Reset	Description
7	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	SESREQIM	R/W	0	Session Request Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
5	DISCONIM	R/W	0	Disconnect Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
4	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
3	SOFIM	R/W	0	Start-of-Frame Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
2	RESTIM	R/W	1	Reset Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
1	RESIM	R/W	1	Resume Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked
0	SUSPDIM	R/W	0	Suspend Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked

4.20.3.4.14 USB_RIS - USB RAW INTERRUPT STATUS REGISTER

Host mode:

7 R/W	6 R/W	5 R/W	4 R/W	3 R/W	2 R/W	1 R/W	0 R/W
VBUS_ERROR_RIS	SESS_REQ_RIS	DISCONRI	CONRI	SOFRI	Peripheral mode : RESET_RIS Host mode : BABBLE_RIS	RESUME_RIS	SUSPEND_RIS

Offset: 0x00AC

Bits	Name	Type	Reset	Description
7	VBUS_ERROR_RIS	R/W	0	Set when VBus drops below the VBus Valid threshold during a session. Only valid when USB is 'A' device. 0: No interrupt occurs 1: An interrupt has occurred
6	SESS_REQ_RIS	R/W	0	Set when Session Request signaling has been detected. Only valid when USB is 'A' device. 0: No interrupt occurs 1: An interrupt has occurred
5	DISCON_RIS	R/W	0	Set in Host mode when a device disconnect is detected. Set in Peripheral mode when a session ends. 0: No interrupt occurs 1: An interrupt has occurred
4	CONN_RIS	R/W	0	Set when a device connection is detected. Only valid in Host mode. 0: No interrupt occurs 1: An interrupt has occurred
3	SOF_RIS	R/W	0	Set when a new frame starts. 0: No interrupt occurs 1: An interrupt has occurred
2	Peripheral mode : RESET_RIS Host mode : BABBLE_RIS	R/W	0	Peripheral mode : Set in Peripheral mode when Reset signaling is detected on the bus. 0: No interrupt occurs 1: An interrupt has occurred Host mode : Set in Host mode when babble is detected. 0: No interrupt occurs 1: An interrupt has occurred
1	RESUME_RIS	R/W	0	Set when Resume signaling is detected on the bus while the USB is in Suspend mode. 0: No interrupt occurs 1: An interrupt has occurred
0	SUSPEND_RIS	R/W	0	Set when Suspend signaling is detected on the bus. Only valid in Peripheral mode. 0: No interrupt occurs 1: An interrupt has occurred

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W	R/W	RO	R/W	R/W	R/W	R/W
	SESREQRI	DISCONRI		SOFRI	RESTRI	RESRI	SUSPDRI

Offset: 0x00AC

Bits	Name	Type	Reset	Description
7	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	SESREQRI	R/W	0	Session Request Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
5	DISCONRI	R/W	0	Disconnect Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
4	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
3	SOFRI	R/W	0	Start-of-Frame Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
2	RESTRI	R/W	0	Reset Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
1	RESRI	R/W	0	Resume Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
0	SUSPDRI	R/W	0	Suspend Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.20.3.4.15 USB_ISC - USB INTERRUPT STATUS AND CLEAR REGISTER

Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.

Host mode:

7	6	5	4	3	2	1	0
R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
VBUS_ERROR_ISC	SESS_REQ_ISC	DISCONIS	CONN_ISC	SOF_ISC	Peripheral mode : RESET_ISC Host mode : BABBLE_ISC	RESUME_ISC	SUSPEND_ISC

Offset: 0x00B0

Bits	Name	Type	Reset	Description
7	VBUS_ERROR_ISC	R/W1C	0	VBus Error Interrupt Status and Clear. 0: VBus Error Interrupt has not occurred or is masked 1: An unmasked VBus Error interrupt has occurred. Write '1' to clear this status
6	SESS_REQ_ISC	R/W1C	0	Session Request Interrupt Status and Clear. 0: Session Request Interrupt has not occurred or is masked 1: An unmasked Session Request interrupt has occurred. Write '1' to clear this status
5	DISCON_ISC	R/W1C	0	Device Disconnect Interrupt Mask Status and Clear. 0: Discon Interrupt has not occurred or is masked 1: An unmasked Device Disconnect interrupt has occurred. Write '1' to clear this status
4	CONN_ISC	R/W1C	0	Device Connect Interrupt Status and Clear. 0: Conn Interrupt has not occurred or is masked 1: An unmasked Device Connect interrupt has occurred. Write '1' to clear this status
3	SOF_ISC	R/W1C	0	Start of new Frame (SOF) Interrupt Status and Clear. 0: SOF Interrupt has not occurred or is masked 1: An unmasked SOF interrupt has occurred. Write '1' to clear this status
2	Peripheral mode : RESET_ISC Host mode : BABBLE_ISC	R/W1C	0	Peripheral mode : Reset Interrupt Status and Clear 0: Reset Interrupt has not occurred or is masked 1: An unmasked Reset interrupt has occurred. Write '1' to clear this status. Host mode : Babble Interrupt Status and Clear. 0: Babble Interrupt has not occurred or is masked 1: An unmasked Babble interrupt has occurred. Write '1' to clear this status
1	RESUME_ISC	R/W1C	0	Resume Interrupt Status and Clear. 0: Resume Interrupt has not occurred or is masked 1: An unmasked Resume interrupt has occurred. Write '1' to clear this status
0	SUSPEND_ISC	R/W1C	0	Suspend Interrupt Status and Clear. 0: Suspend Interrupt has not occurred or is masked 1: An unmasked Suspend interrupt has occurred. Write '1' to clear this status

Peripheral mode:

7	6	5	4	3	2	1	0
RO	R/W1C	R/W1C	RO	R/W1C	R/W1C	R/W1C	R/W1C
	SESREQIS	DISCONIS		SOFIS	RESTIS	RESIS	SUSPDIS

Offset: 0x00B0

Bits	Name	Type	Reset	Description
7	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
6	SESREQIS	R/W1C	0	Session Request Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
5	DISCONIS	R/W1C	0	Disconnect Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
4	reserved	RO	0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
3	SOFIS	R/W1C	0	Start-of-Frame Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
2	RESTIS	R/W1C	0	Reset Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
1	RESIS	R/W1C	0	Resume Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
0	SUSPDIS	R/W1C	0	Suspend Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.

4.21 HARDWARE DIVIDER (HDIV)

The hardware divider (HDIV) is useful to the high performance application. The hardware divider is a signed, integer divider with both quotient and remainder outputs.

HDIV Main Features:

- Signed (two's complement) integer calculation
- 32-bit dividend with 16-bit divisor calculation capacity
- 32-bit quotient and 32-bit remainder outputs (16-bit remainder with sign extends to 32-bit)
- Divided by zero warning flag
- 36 HCLK clocks taken for one cycle calculation
- Write divisor to trigger calculation

Waiting for calculation ready automatically when reading quotient and remainder

4.21.1 FUNCTIONAL DESCRIPTION

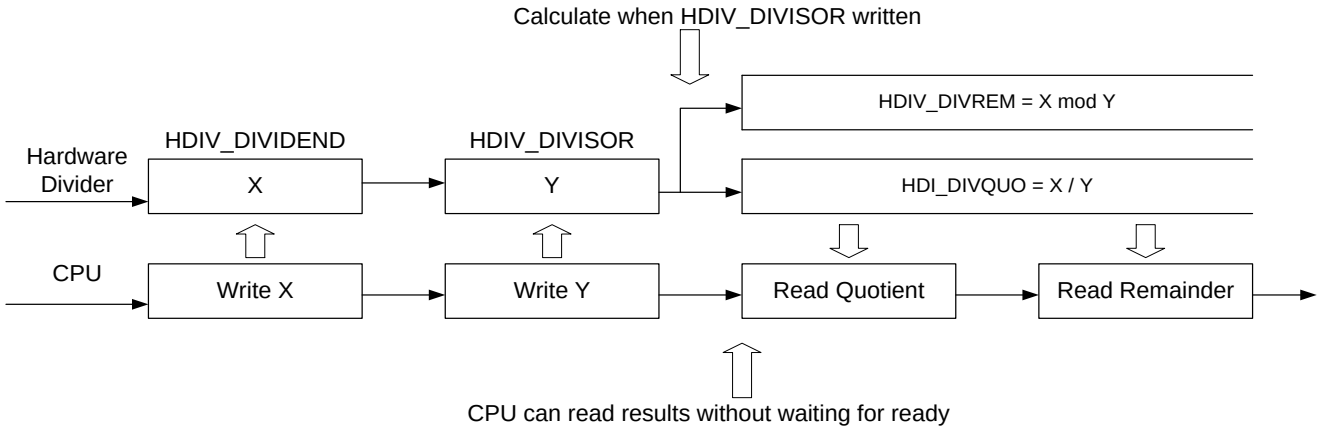
To use hardware divider, it needs to set dividend first. Then set divisor and the hardware divider will trigger calculation automatically after divisor written. The calculation results including the quotient and remainder could be got by reading **HDIV_DIVQUO** and **HDIV_DIVREM** register. If CPU reads **HDIV_DIVQUO** or **HDIV_DIVREM** before hardware divider calculation finishing, CPU will be held until hardware divider finishing the calculation. Therefore, CPU can always get valid results after trigger one hardware divider calculation without software delay.

DIV0 flag of **HDIV_DIVSTS** will be set if divisor is 0.

The dividend is 32-bit signed integer and divisor is 16-bit signed integer. The quotient is 32-bit signed integer and the remainder is 16-bit signed integer.

The following figure shows the operation flow of hardware divider. To calculation X / Y, CPU needs to write X to **HDIV_DIVIDEND** register, and then write Y to **HDIV_DIVISOR**. CPU can read **HDIV_DIVQUO** and **HDIV_DIVREM** registers to get calculation results after **HDIV_DIVISOR** been written.

Figure 4.21-1: Hardware Divider Operation Flow

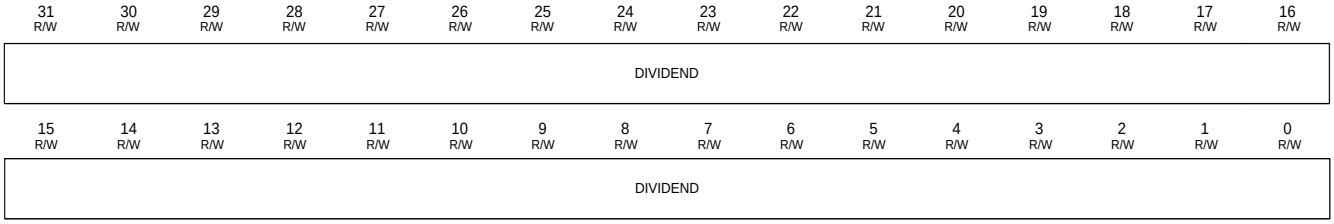


4.21.2 RESERVED HDIV REGISTER MAP

Base Address: 0x5005_0000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	HDIV_DIVIDEND	R/W	0x0000_0000	Dividend Source Register	542
0x0004	HDIV_DIVISOR	R/W	0x0000_FFFF	Divisor Source Resister	542
0x0008	HDIV_DIVQUO	R/W	0x0000_0000	Quotient Result Resister	543
0x000C	HDIV_DIVREM	R/W	0x0000_0000	Remainder Result Register	543
0x0010	HDIV_DIVSTS	RO	0x0000_0001	Divider Status Register	544

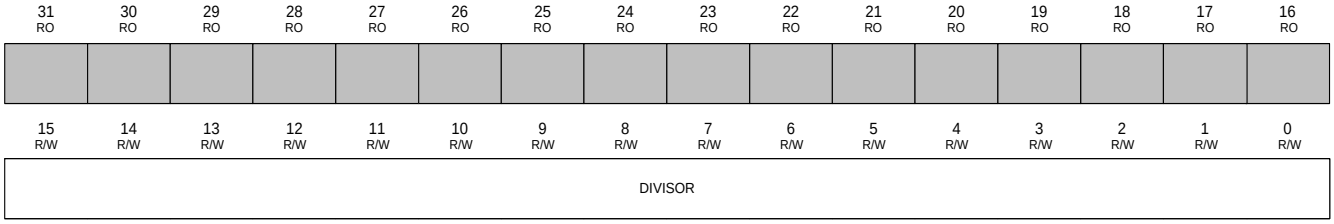
4.21.2.1 HDIV_DIVIDEND - DIVIDEND SOURCE REGISTER



Offset: 0x0000

Bits	Name	Type	Reset	Description
31: 0	*DIVIDEND*	R/W	0x0	Dividend Source This register is given the dividend of divider before calculation starting.

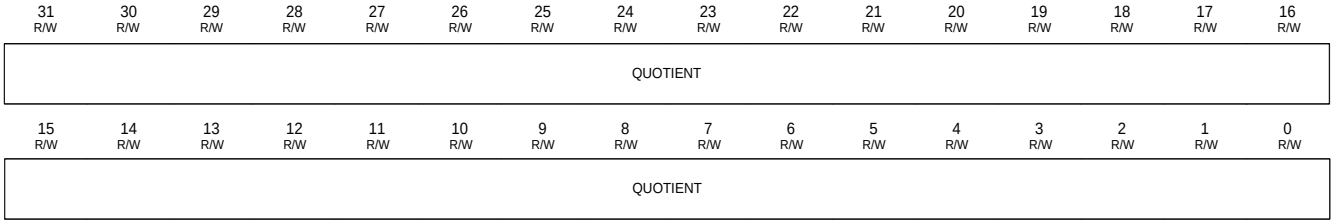
4.21.2.2 HDIV_DIVISOR - DIVISOR SOURCE REGISTER



Offset: 0x0004

Bits	Name	Type	Reset	Description
31: 16	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
15:0	*DIVISOR*	R/W	0x0	Divisor Source This register is given the divisor of divider before calculation starts. Note:When this register is written, hardware divider will start calculate

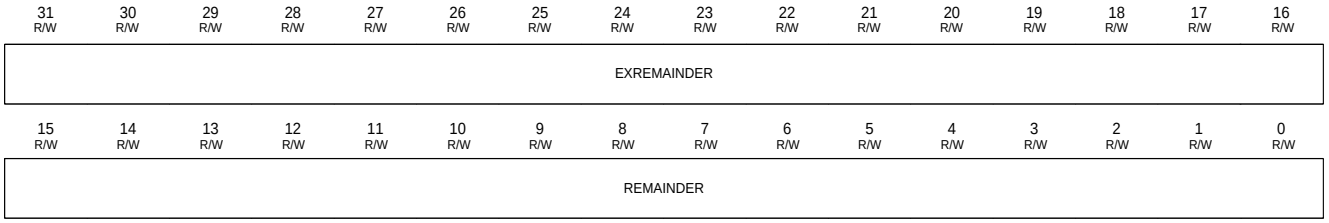
4.21.2.3 HDIV_DIVQUO - QUOTIENT RESULT RESISTER



Offset: 0x0008

Bits	Name	Type	Reset	Description
31: 0	*QUOTIENT*	R/W	0x0	Quotient Result This register holds the quotient result of divider after calculation complete.

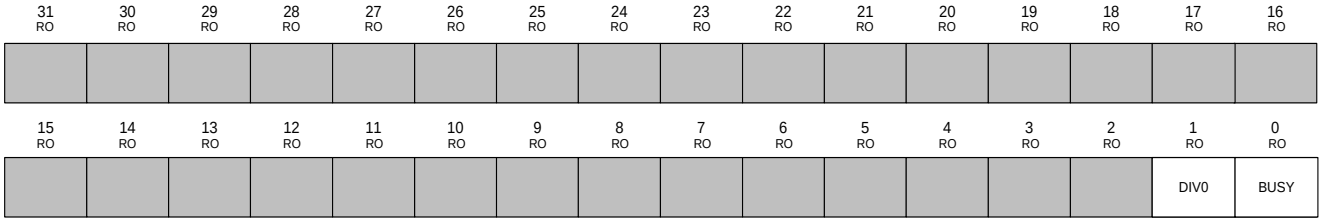
4.21.2.4 HDIV_DIVREM - REMAINDER RESULT REGISTER



Offset: 0x000C

Bits	Name	Type	Reset	Description
31: 16	*EXTREMAINDER*	R/W	0x0	Sign Extension of *EXTREMAINDER* The remainder of hardware divider is 16-bit sign integer (*REMAINDER*) with sign extension (*EXTREMAINDER*) to 32-bit integer.
15:0	*REMAINDER*	R/W	0x0	Remainder Result This register holds the remainder result of divider after calculation complete

4.21.2.5 HDIV_DIVSTS - DIVIDER STATUS REGISTER



Offset: 0x0010

Bits	Name	Type	Reset	Description
31: 2	*reserved*	RO	0x0	Software should not rely on the value of a reserved bit. Considering the compatibility with other products, the values of this should not be written or read
1	*DIV0*	RO	0	Divisor Zero Warning 0: The divisor is not 0. 1: The divisor is 0. Note: The DIV0 flag is used to indicate divide-by-zero situation and updated whenever **HDIV_DIVISOR** is written. This register is read only
0	*BUSY*	RO	0x0	Divider Busy Status 0: Divider is idle 1: Divider is busy

4.22 APPENDIX A CODE EXAMPLES

4.22.1 TIMER

4.22.1.1 INPUT CAPTURE MODE EXAMPLE

```
/* Select the active input TI0 (CC0S = 01),
   program the input filer for zero clock cycles (IC0F = 000),
   and prescaler at each valid transition ( IC0PS = 00)*/
/* Enable capture by setting CC0E */
/* Enable interrupt on Capture/Compare */
/* Enable counter */
TMR_CCMR= IC0_CAPTURE_TI0 | IC0_PSC_0 | IC0_DEBOUCE_0;
TMR_CCER= CH0_EN;
TMR_IER= INTR_CH0;
TMR_CR1|= ENABLE;
```

4.22.1.2 PWM INPUT MODE EXAMPLE

```
/* Select the active input TI0 (CC0S = 01),
   program the input filer for zero clock cycles (IC0F = 000),
   and prescaler at each valid transition ( IC0PS = 00)*/
/* Enable capture by setting CC0E */
/* Enable interrupt on Capture/Compare */
/* Enable counter */
TMR_CCMR= IC0_CAPTURE_TI0|IC1_CAPTURE_TI0;
TMR_SMCR= SLAVE_SEL_TI0|SLAVE_RST|SLAVE_ENABLE;
TMR_CCER=CH0_EN|IC0_RISE|CH1_ENIC1_FALL;
TMR_IER= INTR_SLV;
TMR_CR1|= ENABLE
```

5. ELECTRICAL CHARACTERISTICS

5.1 MAXIMUM RATINGS

The maximum ratings are the limits to which the device can be subjected without permanently damaging the device. Device reliability may be adversely affected by exposure to absolute-maximum ratings for extended periods.

Symbol	Parameter	Min.	Max.	Unit
V_{DD} -V_{SS}	V_{DD} supply voltage	2.0	3.6	V
V_I/V_O	Input Voltage		3.6	V
T_A	Operating ambient temperate	-40	85	℃
T_{STG}	Storage temperate	-40	125	℃

5.2 OPERATING CONDITIONS

All the electrical characteristics are applicable to the following conditions unless otherwise specified:

- Operating temperature range: T_A = -40°C to 85°C and for a junction temperature up to T_J = 100°C.
- Typical values are based on T_A = 25°c and VI= 3,3V unless otherwise specified

5.3 I/O PIN CHARACTERISTICS

Symbol	Parameter	Conditions		Min.	Typ.	Max.	Unit
R_{PU}	Pull-up resistor			34K	49K	74K	Ω
R_{PD}	Pull-down resistor			30K	47K	86K	Ω
I_{LKG}	Input leakage current					±1μ	A
V_{OL}	Output low-level voltage					0.4	V
V_{OH}	Output high-level voltage			2.4			V
V_{IL}	Input low-level voltage			-0.3		0.8	V
V_{IH}	Input high-level voltage			2		3.6	V
V_{T+}	Schmitt Trigger Low to High Threshold Point			1.53		1.66	V
V_{T-}	Schmitt Trigger High to Low Threshold Point			1.13		1.27	V
I_{OL}	Low Level Output Current	V_{OL}(Max)	Output Drive: 2mA	2.4	3.8	5.3	mA
			Output Drive: 4mA	4.7	7.6	10.6	
I_{OH}	High Level Output Current	V_{OH}(Min)	Output Drive: 2mA	3.4	7.0	11.6	mA
			Output Drive: 4mA	6.9	14.0	23.2	

5.4 ON-CHIP LOW DROP-OUT (LDO) REGULATOR CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power supply range	V_{DD}		2.0	3.3	3.6	V
Bandgap Reference	VREF		1.19	1.22	1.26	V
Output Current	Iout			200		mA
Power consumption	Current	Normal mode	13	16.3	26.2	μA
		Lpr mode	2.5	3.3	4.9	
Output Voltage	Vout	Corner Change. 18Ω Loading Vref=1.2V	1.76	1.82	1.87	V
Temperature coefficient	Tc	-40℃~125℃	233	396	578	ppm
Line Regulation		18Ω Loading		2.7		%/V
		1800Ω Loading		0.41		
Load Regulation		1mA ~100mA		0.0019		%/mA
Output settling time	Ts	99%	4	6.9	9.6	ms
Charge pump on threshold	Vthl	Vout change		1.776		V
Over voltage protect threshold	Vthh	Vout change		1.85		V

5.5 PHASE LOCKED LOOP CHARACTERISTICS

Parameter	Symbol	Condition	Min.	Typ.	Max.	Units
Power supply range	V_{DD}		2.0	3.3	3.6	V
PLL input clock	Fpll_in			4		MHz
PLL output clock	Fpll_out		4		72	MHz
PLL lock time	Tlock				400	μs
PLL frequency conversion time	Tconv				100	μs
PLL power consumption	current	VDD_A, Fpll_out=48MHz		260		μA
		VDD_D, Fpll_out=48MHz		170		μA

5.6 POWER-ON RESET CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power supply range	V_{DD}		1.8	3.3	3.6	V
Programmable voltage detector(PVD) level selection	VPVD	PLS[2:0]=000(rising edge)	2.15	2.2	2.25	V
		PLS[2:0]=000(falling edge)	2.05	2.1	2.15	V
		PLS[2:0]=001(rising edge)	2.24	2.3	2.36	V
		PLS[2:0]=001(falling edge)	2.15	2.2	2.25	V
		PLS[2:0]=010(rising edge)	2.34	2.4	2.46	V
		PLS[2:0]=010(falling edge)	2.24	2.3	2.36	V
		PLS[2:0]=011(rising edge)	2.44	2.5	2.56	V
		PLS[2:0]=011(falling edge)	2.34	2.4	2.46	V
		PLS[2:0]=100(rising edge)	2.54	2.6	2.66	V
		PLS[2:0]=100(falling edge)	2.44	2.5	2.56	V
		PLS[2:0]=101(rising edge)	2.63	2.7	2.77	V
		PLS[2:0]=101(falling edge)	2.54	2.6	2.66	V
		PLS[2:0]=110(rising edge)	2.73	2.8	2.87	V
		PLS[2:0]=110(falling edge)	2.63	2.7	2.77	V
		PLS[2:0]=111(rising edge)	2.83	2.9	2.97	V
		PLS[2:0]=111(falling edge)	2.73	2.8	2.87	V
PVD hysteresis	VPVDhyst			100		mV
Power on/Power down reset threshold	VPDR	Falling edge	1.85	1.89	1.94	V
		Rising edge	1.89	1.93	1.98	V
PDR hysteresis	VPDRhyst			50		mV
Reset temporization	Trsttempo		1.5	2.2	4.7	mS

5.7 NRST CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power supply range	V_{DD}		2.0	3.3	3.6	V
NRST Input low level voltage	$V_{IL(NRST)}$		-0.5		0.8	V
NRST Input high level voltage	$V_{IH(NRST)}$		2		V_{DD} + 0.5	
NRST Schmitt trigger voltage hysteresis	$V_{hys(NRST)}$			200		mV
Weak pull-up equivalent resistor	R_{PU}	$V_{IN} = V_{SS}$	30	40	50	kΩ
NRST Input filtered pulse	$V_{F(NRST)}$				100	ns
NRST Input not filtered pulse	$V_{NF(NRST)}$		300			ns
External load capacitance	C_L			0.1		μF
Native reset time	T_R	C_L =0.1u		2.7		ms

5.8 8 MHZ XTAL CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power supply range	V_{DD}		2.0	3.3	3.6	V
Oscillator frequency	f_{OSC_IN}			8		MHz
Feedback resistor	RF			200		kΩ
Recommended load capacitance versus equivalent serial resistance of the crystal (R_S)	C	R_S = 30 Ω			30	pF
Oscillator driving current	I_{VDD}	V_{DD} = 3.3 V			1	mA
Oscillator Transconductance	g_m	Startup	25			mA/V
startup time	t_{SU}	V_{DD} is stabilized		2		ms
User External clock source frequency	f_{HSE_ext}		1	8	25	MHz
OSC_IN input pin high level voltage	V_{HSEH}		0.7V_{DD}		V_{DD}	V
OSC_IN input pin low level voltage	V_{HSEL}		V_{SS}		0.3V_{DD}	
OSC_IN high or low time	$t_{w(HSE)}$ $t_{w(HSE)}$		16			ns
OSC_IN rise or fall time	$t_{r(HSE)}$ $t_{f(HSE)}$				20	
OSC_IN input capacitance	$C_{in(HSE)}$			5		pF
Duty cycle	$DuCy_{(HSE)}$		45		55	%
OSC_IN Input leakage current	I_L	$V_{SS} \le V_{IN} \le V_{DD}$			±1	μA

5.9 4 MHZ RCOSC CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power supply range	V_{DD}		2.0	3.3	3.6	V
Power consumption	Current	OSC_4M_A[7:0]= 80h	45	65	90	μA
Output CLK Frequency	Fout	OSC_4M_A[7:0]= 80h	3.13	4	6.2	MHz
Frequency trimming step	Fstep			14		KHz
Frequency trimming range	Frange	’	-40%		37.5	%
Frequency temperature coefficient	Fppm	-40℃~125℃		151		ppm/℃
Output startup time	Tsu				10	CLK
Trim settling time	Tst				5	CLK

5.10 32 KHZ XTAL

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power supply range	V_{DD}		2	3.3	3.6	V
Recommended load capacitance versus equivalent serial resistance of the crystal (R_S)	C_L	R_S = 30 kΩ		10		pF
Oscillator driving current	I_{vdd}	C3,C2,C1=1,1,0,		400		nA
startup time	t_{SU}	V_{DD} is stabilized			3	s
User External clock source frequency	f_{LSE_ext}	EN=1		32.768		kHz
Duty cycle	$DuCy_{(LSE)}$		30		70	%
OSC32_IN Input leakage current	I_L				±1	μA

5.11 A/D IO CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Input low voltage	V_{IL}		-0.3		0.8	V
Input high voltage	V_{IH}		2		3.6	V
Threshold point	V_T		1.33	1.4	1.47	V
Schemitt trigger low to high threshold point	V_{T+}		1.53	1.6	1.66	V
Schemitt trigger high to low threshold point	V_{T-}		1.13	1.2	1.27	V
Input leakage current	I_I	V_I=3.3V or 0V			±1μ	A
Tri-state output leakage current	I_{OZ}	V_O=3.3V or 0V			±1μ	A
Pull-up resistor	R_{PU}		34K	49K	74K	Ω
Pull-down resistor	R_{PD}		30K	47K	86K	Ω
Output low voltage	V_{OL}				0.4	V
Output high voltage	V_{OH}		2.4			V
Low level output current	I_{OL}	V_{OL}(max), 02mA	2.4	3.8	5.3	mA
		V_{OL}(max), 04mA	4.7	7.6	10.6	
High level output current	I_{OH}	V_{OH}(min), 02mA	3.4	7.0	11.6	mA
		V_{OH}(min), 04mA	6.9	14.0	23.2	

5.12 TEMPERATURE SENSOR CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power Supply Range	V_{DD}		2.0	3.3	3.63	V
Current Consumption	I_{DD}		70	100	160	μA
Standby Current	I_{STBY}	EN = LOW			1	μA
Temperature Range	T_A		-40		125	℃
Output Voltage	V_O	T_A = - 40℃	0.85	0.86	0.87	V
		T_A = +125℃	1.44	1.45	1.46	V
Start-Up Time	t_{start}				50	μs

5.13 ADC+ PGA CHARACTERISTICS

5.13.1 RECOMMENDED OPERATING CONDITIONS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Supply voltage	V_{DD}		2.4	3.3	3.6	V
Supply current(ADC+PGA)	Idd			3.5		mA
Clock frequency	Fclk			48		MHz
Operation temperature	Ta		-40		125	℃
Simulation condition	Tj		-40		125	℃

5.13.2 DC ACCURACY

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Resolution	Res			12		Bit
ENOB	ENOB		9.2	10.5	11.8	
Integral nonlinearity	INL				+/-5	LSB
Offset error	Eoff				5	%
Gain error	Egain				5	%

5.13.3 TIMING SPECIFICATION (REFER TO TIMING CHART)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Clock duty cycle	CLK_DUTY		45		55	%
ADCEN setup time to START"L"->"H"	t1		200			ns
ADSEL setup time to START"L"->"H"	t2		20			ns
ADCOUT hold time to START"L"->"H"	t3		20			ns
Conversion time[1]	Tc		1.12			μs
PGA Settling time	Tsettleing	Gain=4		3		μs

Note 1) Tc include time sampling input signal (the period START is "H"), 80ns

5.13.4 PGA CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Vswing(Vpp)	Vswin		0.3		3	V
PGA bandwidth	BW1	GAIN=8, -3dB Non-inverting		12		MHz
	BW2	GAIN=7, -3dB Inverting		8.5		
Total harmonic distortion	Thd	Fin=100khz		0.1		%

5.14 COMPARATOR CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power Supply Range	VDDH		2.0	3.3	3.6	V
Current Consumption	IDD	CAON<3:0> = 1111		200		μA
Standby Current	ISTBY	CAON<3:0> = 0000			1	μA
Propagation Delay	T_{PGD}			10		ns
Start time*	TS				5	μs

*: T_S is the start time that the comparator needs to have a stable time after CAON<3:0> is switched from 0000 to a different value.

5.15 RCOSC_32K CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power supply range	V_{DD}		2.0	3.3	3.6	V
Power consumption	Current	Corner Change	0.30	0.55	0.89	uA
Output CLK Frequency	Fout	Corner Change	28	32	53	kHz
Output startup time	Tsu	Corner Change		10		CLK
Output Frequency random error	σ(Fout)		1.04		2.05	%
Frequency temperature coefficient	Fppm	-40℃~125℃		2097		ppm/℃

5.16 DAC CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power supply range	V_{DD}		2.4	3.3	3.63	V
Reference Supply Range	V_{REF}	$V_{REF} \le V_{DD}$	2.4	3.3	3.63	V
Clock Frequency	freq			200		KHz
Current Consumption	I_{DD}			900		uA
Standby Current	I_{STBY}	EN = LOW			1	uA
Temperature Range	T_A		-40		125	℃
Output Voltage	V_O	DATA[11:0]=000		0.55		V
		DATA[11:0]=FFF		2.75		

5.17 USB PAD CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Power Supply Range	V_{DD}		2.4	3.3	3.63	V
Temperature Range	T_A	$V_{REF} \le V_{DD}$	-40		125	℃
Pull-up Resistance	R_{PU}			1.5		kΩ
Pull-down Resistance	R_{PD}			15		kΩ

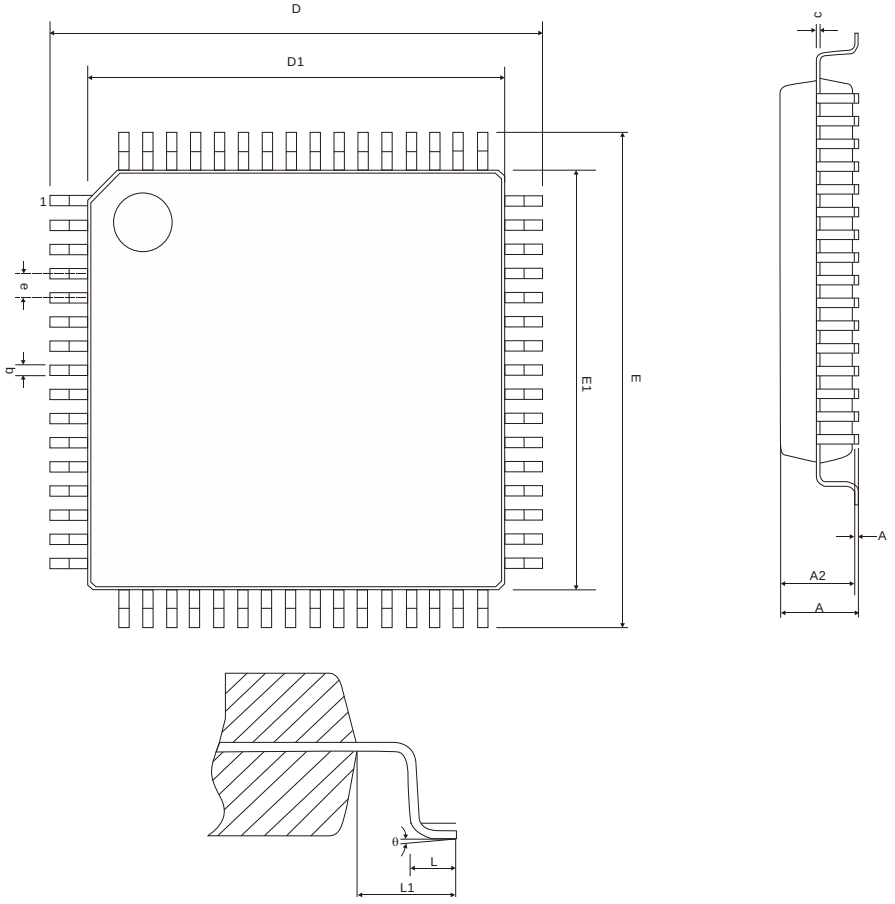
5.18 POWER CONSUMPTION TABLE

This table shows the power consumption data under three different modes: Standby Mode, Sleep Mode, and Normal Mode.

System Clock Source	Frequency	Run Mode		Sleep Mode, CPU Clock OFF				unit
		All IP Clock ON	All IP Clock OFF	All IP Clock ON		All IP Clock OFF		
				Eflash on	Eflash sleep	Eflash on	Eflash sleep	
LSI	32k	668 648	622 609	660 643	34 33.9	662 626	26.1 26.0	uA
LSE								
HSI	4M	2.6 2.53	1.49 1.45	1.92 1.84	1.29 1.26	0.867 0.826	0.23 0.228	mA
HSE	4M	2.84 2.72	1.74 1.69	2.14 2.06	1.52 1.49	1.02 0.98	0.417 0.414	mA
PLL+HSI	4M	3.28 3.19	2.06 2.02	2.51 2.43	1.84 1.81	1.43 1.39	0.80 0.79	mA
	8M	5.02 4.86	2.74 2.69	3.72 3.58	3.03 2.98	1.52 1.47	0.88 0.87	mA
	16M	8.71 8.38	4.07 3.93	6.08 5.86	5.44 5.20	1.71 1.66	1.08 1.06	mA
	32M	14.78 14.28	6.17 5.95	10.74 10.40	10.1 9.82	2.10 2.04	1.47 1.44	mA
	48M	20.32 19.68	7.67 7.36	15.37 14.86	14.65 14.24	2.38 2.33	1.75 1.70	mA
PLL+HSE	4M	3.38 3.29	2.36 2.34	2.72 2.65	2.11 2.08	1.61 1.58	1.01 1.00	mA
	8M	5.24 5.16	3.22 3.14	3.91 3.84	3.31 3.28	1.71 1.68	1.11 1.10	mA
	16M	8.94 8.65	4.76 4.63	6.38 6.18	5.72 5.65	1.89 1.87	1.31 1.30	mA
	32M	15.08 14.63	6.78 6.56	11.11 10.88	10.55 10.24	2.31 2.25	1.70 1.68	mA
	48M	20.66 20.12	7.79 7.51	15.74 15.34	15.08 14.76	2.58 2.51	1.98 1.93	mA
LSI	32k	614 591	16.1 16.0	610 588	14.4 14.3	1.9 1.8	1.1 1.1	uA
LSE								
HSI	4M	614 591	15.3 15.2	615 589	14.4 14.3	1.9 1.8	1.1 1.0	uA
HSE	4M	614 592	16.1 16.0	615 589	14.5 14.4	1.9 1.8	1.1 1.0	uA

6. PACKAGE INFORMATION

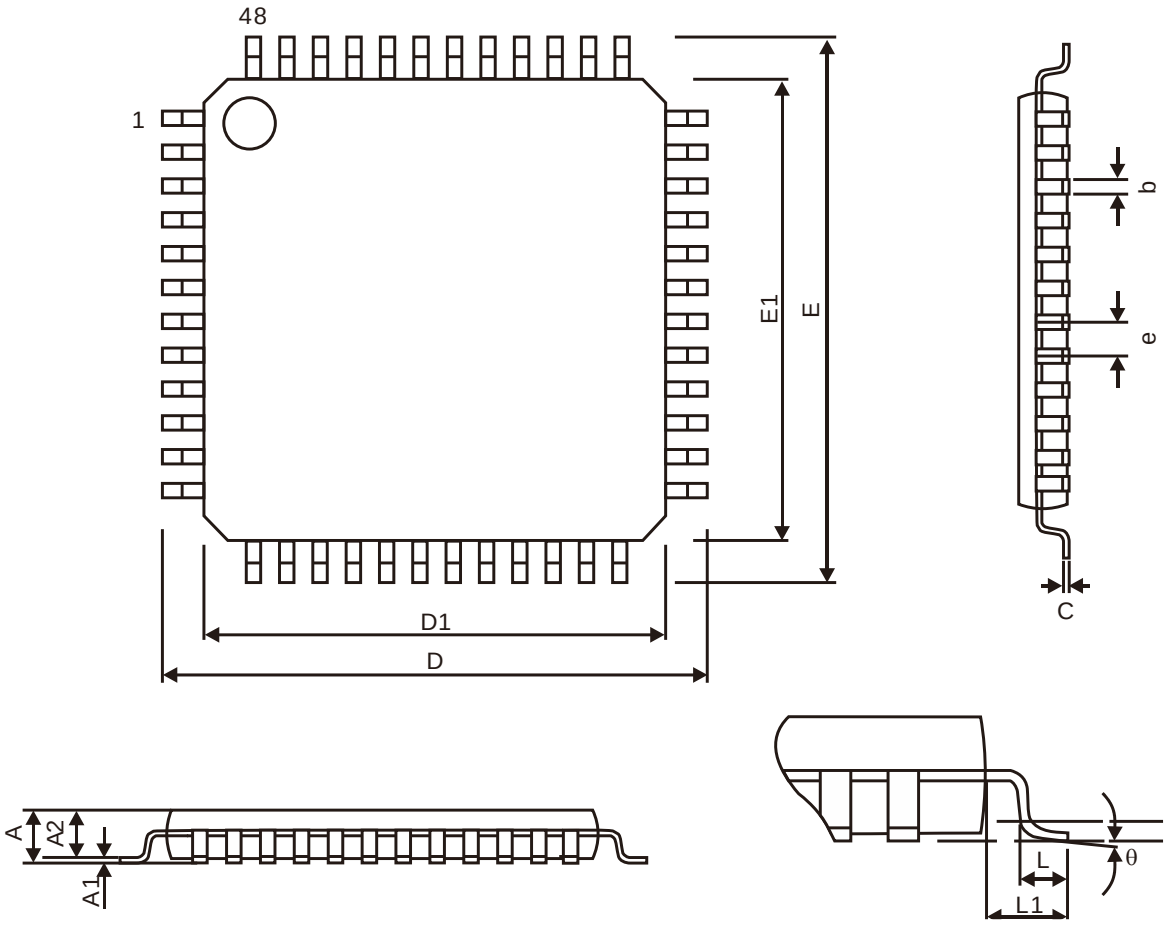
64-PIN, LQFP, 7X7



Symbol	Dimensions (mm)		
	Min.	Nom.	Max.
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
b	0.13	0.18	0.23
c	0.09	-	0.20
D	9.00 BSC		
D1	7.00 BSC		
E	9.00 BSC		
E1	7.00 BSC		
e	0.40 BSC		
L	0.45	0.60	0.75
L1	1.00 REF		
θ	0°	3.5°	7°

Note: Refer to JEDEC MS-026 BBD

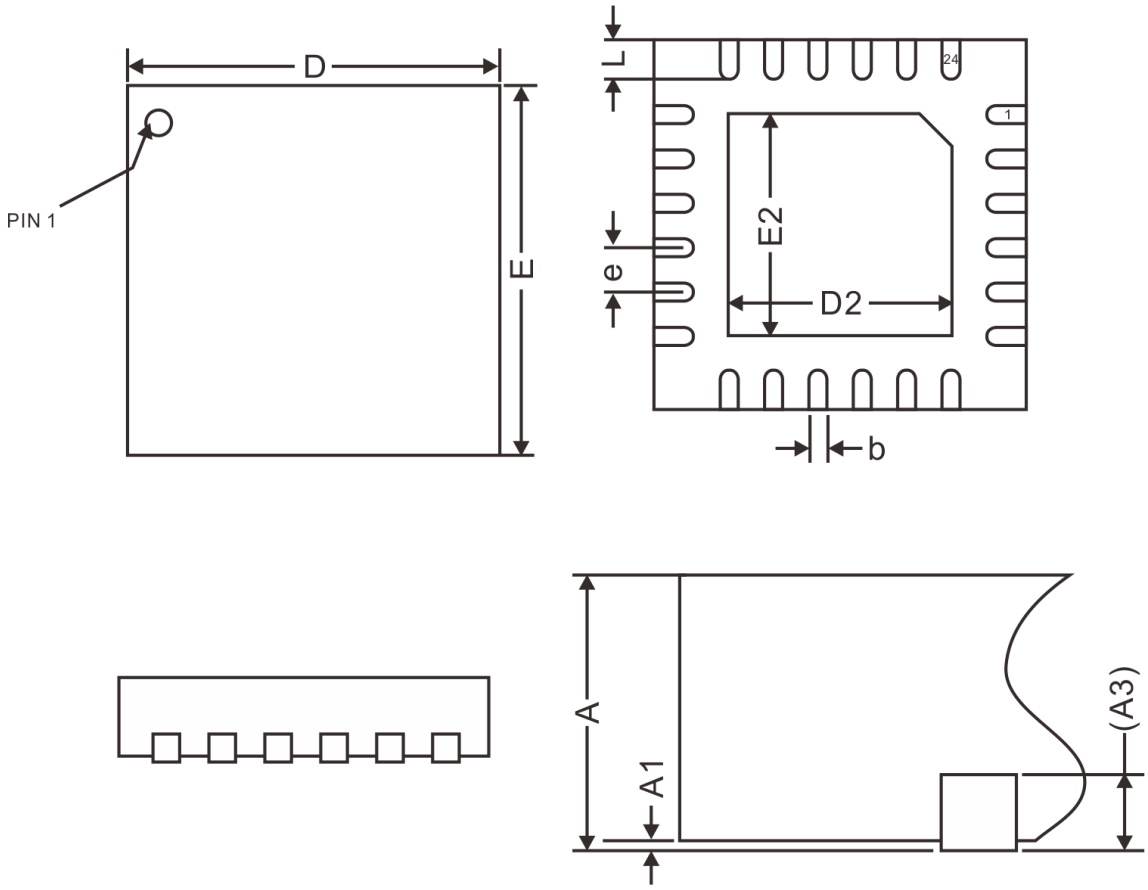
48-PIN, LQFP, 7X7



Symbol	Dimensions (mm)		
	Min.	Nom.	Max.
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
b	0.17	0.22	0.27
c1	0.09	-	0.20
D	9.00 BSC		
D1	7.00 BSC		
E	9.00 BSC		
E1	7.00 BSC		
e	0.50 BSC		
L	0.45	0.60	0.75
L1	1.00 REF		
θ	0°	3.5°	7°

Note: Refer to JEDEC MS-026 BBC

24-PIN, QFN



Symbol	Dimensions (mm)		
	Min.	Nom.	Max.
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A3		0.20 REF	
b	0.18	0.25	0.30
D		4.00 BSC	
D2	2.50	2.65	2.80
E		4.00 BSC	
E2	2.50	2.65	2.80
e		0.50 BSC	
L	0.35	0.40	0.45

Note: Refer to JEDEC MO-220 WGGD-6

IMPORTANT NOTICE

Princeton Technology Corporation (PTC) reserves the right to make corrections, modifications, enhancements, improvements, and other changes to its products and to discontinue any product without notice at any time.
PTC cannot assume responsibility for use of any circuitry other than circuitry entirely embodied in a PTC product. No circuit patent licenses are implied.

Princeton Technology Corp.
2F, 233-1, Baociao Road,
Sindian, Dist., New Taipei City 23145, Taiwan
Tel : 886-2-66296288
Fax: 886-2-29174598
http://www.princeton.com.tw