DESCRIPTION

The PT32U301 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 32 Kbytes of Flash and 4 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.

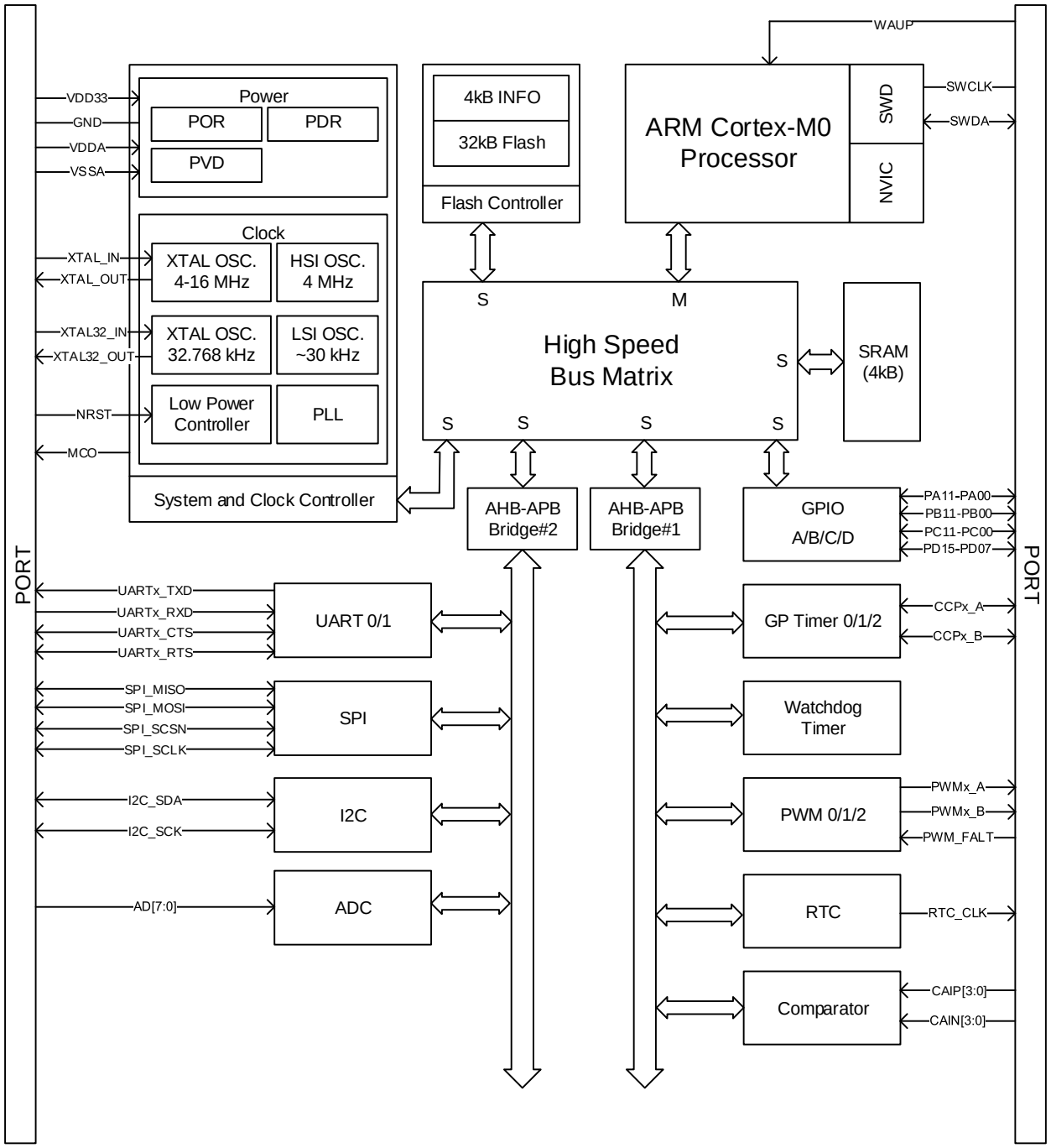
The PT32U301 MCUs give you any essential functionality as a General-purpose MCU. With its highly customizable peripherals, it eases the process of making your own ideal product.

PT32U301 microcontroller peripheral set also includes an External Bus Interface, 2x UARTs, 1X I2C, 1 SPI, 3x general-purpose 16/32-bit timers, an RTC, 3x PWM, an ADC and four analog comparators.

FEATURES

- ARM Cortex M0 Processor
- Performance up to 48 MHz
- Flash memory 32K-Byte
- System SRAM 4K-Byte
- Power control
  - POR/PDR
  - PVD
- Low power mode
  - Sleep mode
  - Stop mode
  - Standby mode
- Divers Peripherals
  - Up to 45 GPIOs
  - 16/32-bit timer * 6
  - WDT/RTC
  - 8 channel 12-bit ADC
  - 4 Comparator with window comparison
  - 2 UART
  - I2C with master/slave mode
  - SPI with variable bit length (4-16 bits)

BLOCK DIAGRAM



CONTENT

DESCRIPTION **1**
FEATURES **1**
BLOCK DIAGRAM **2**
CONTENT **3**
1. ORDER INFORMATION **5**
2. PIN CONFIGURATION **6**
3. PIN DESCRIPTION **9**
3.1 MULTIPLEXING PINS FUNCTION SELECTION 11
3.2 SIGNAL DESCRIPTION 12
4. FUNCTIONAL DESCRIPTION **14**
4.1 SYSTEM AND MEMORY OVERVIEW 14
4.1.1 PT32U301 Memory Mapping *15*
4.2 ARM ® CORTEX™-M0 CORE 17
4.2.1 CPU System Timer Control Register (SYST) *17*
4.2.2 SYST Register Maps *18*
4.2.3 CPU Nested Vectored Interrupt Controller (NVIC) *20*
4.2.4 Vector Table *22*
4.2.5 Operation Description *22*
4.2.6 NVIC Register Maps *22*
4.3 CPU SYSTEM CONTROL 33
4.3.1 CPU System Control Register Map *33*
4.4 SYSTEM CONTROL (SC) 38
4.4.1 Functional Description *38*
4.4.2 System Control Register Map *44*
4.5 FLASH CONTROLLER (FC) 70
4.5.1 Memory Organization *70*
4.5.2 Flash Controller Clocking Frequency *70*
4.5.3 Flash Programming/Erasing *71*
4.5.4 Flash Controller Register Map *71*
4.6 GENERAL PURPOSE I/O (GPIO) 74
4.6.1 Block Diagram *74*
4.6.2 Functional Description *75*
4.6.3 General-Purpose Input/Output Register Map *79*
4.7 UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART) 88
4.7.1 Block Diagram *89*
4.7.2 Functional Description *90*
4.7.3 UART Register Map *96*
4.8 PULSE WIDTH MODULATION (PWM) 115
4.8.1 PWM Block Diagram *116*
4.8.2 PWM Signal Generator Block Diagram *117*
4.8.3 PWM Functional Description *118*
4.8.4 PWM Register Map *124*
4.9 ANALOG TO DIGITAL CONVERTER (ADC) 154
4.9.1 Block Diagram *155*
4.9.2 Functional Description *156*
4.9.3 Initialization and Configuration *158*
4.9.4 ADC Register Map *159*
4.10 GENERAL PURPOSE TIMERS (GPT) 187
4.10.1 Block Diagram *188*
4.10.2 Functional Description *188*
4.10.3 GPTM Initialization and Configuration *195*
4.10.4 General-Purpose Timer Register Map *199*

4.11 ANALOG COMPARATOR (AC) 214
4.11.1 Block Diagram *214*
4.11.2 Functional Description *215*
4.11.3 Comparator Register Map *216*
4.12 WATCH DOG TIMER (WDT) 225
4.12.1 Block Diagram *225*
4.12.2 Operating Flowchart *226*
4.12.3 Functional Description *227*
4.12.4 WDT Register Map *227*
4.13 REAL TIME CLOCK (RTC) 233
4.13.1 Block Diagram *233*
4.13.2 Function Description *234*
4.13.3 RTC Register Map *236*
4.14 INTER INTEGRATED CIRCUIT (I2C) 256
4.14.1 Block Diagram *257*
4.14.2 Functional Description *258*
4.14.3 I2C Register Map *265*
4.15 SERIAL PERIPHERAL INTERFACE (SPI) 285
4.15.1 Block Diagram *286*
4.15.2 SPI Function Description *287*
4.15.3 SPI Programming Model *300*
4.15.4 SPI Register Map *303*
5. ELECTRICAL CHARACTERISTICS 320
5.1 MAXIMUM RATINGS 320
5.2 OPERATING CONDITIONS 320
5.3 I/O PIN CHARACTERISTICS 320
5.4 ON-CHIP LOW DROP-OUT (LDO) REGULATOR CHARACTERISTICS 321
5.5 PHASE LOCKED LOOP CHARACTERISTICS 321
5.6 POWER-ON RESET CHARACTERISTICS 322
5.7 NRST CHARACTERISTICS 323
5.8 8 MHZ XTAL CHARACTERISTICS 324
5.9 4 MHZ RCOSC CHARACTERISTICS 325
5.10 32 KHZ XTAL 326
5.11 TEMPERATURE SENSOR CHARACTERISTICS 326
5.12 ADC+ PGA CHARACTERISTICS 327
5.12.1 Recommended operating conditions *327*
5.12.2 DC accuracy *327*
5.12.3 Timing specification (refer to timing chart) *327*
5.12.4 PGA Characteristics *328*
5.13 COMPARATOR CHARACTERISTICS 328
5.14 RCOSC_32K CHARACTERISTICS 328
5.15 POWER CONSUMPTION TABLE 329
6. PACKAGE INFORMATION 330

1. ORDER INFORMATION

Valid Part Number	Package Type	Top Code
PT32U301	LQFP 48	PT32U301 HGMA
PT32U301	LQFP 32	PT32U301 HGWA
PT32U301	SSOP 24	PT32U301 HCPA
PT32U301	SOP 16	PT32U301 HXLA
PT32U301	SOP 8	PT32U301

P T 3 2 U 3 0 1 H G M A

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
6 : SiP

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
1 : Default Variant

Flash Memory Size
T : 64KB Eflash Memory, 8KB SRAM
H : 32KB Eflash Memory, 4KB SRAM
R : 64KB E flash Memory, 16KB SRAM

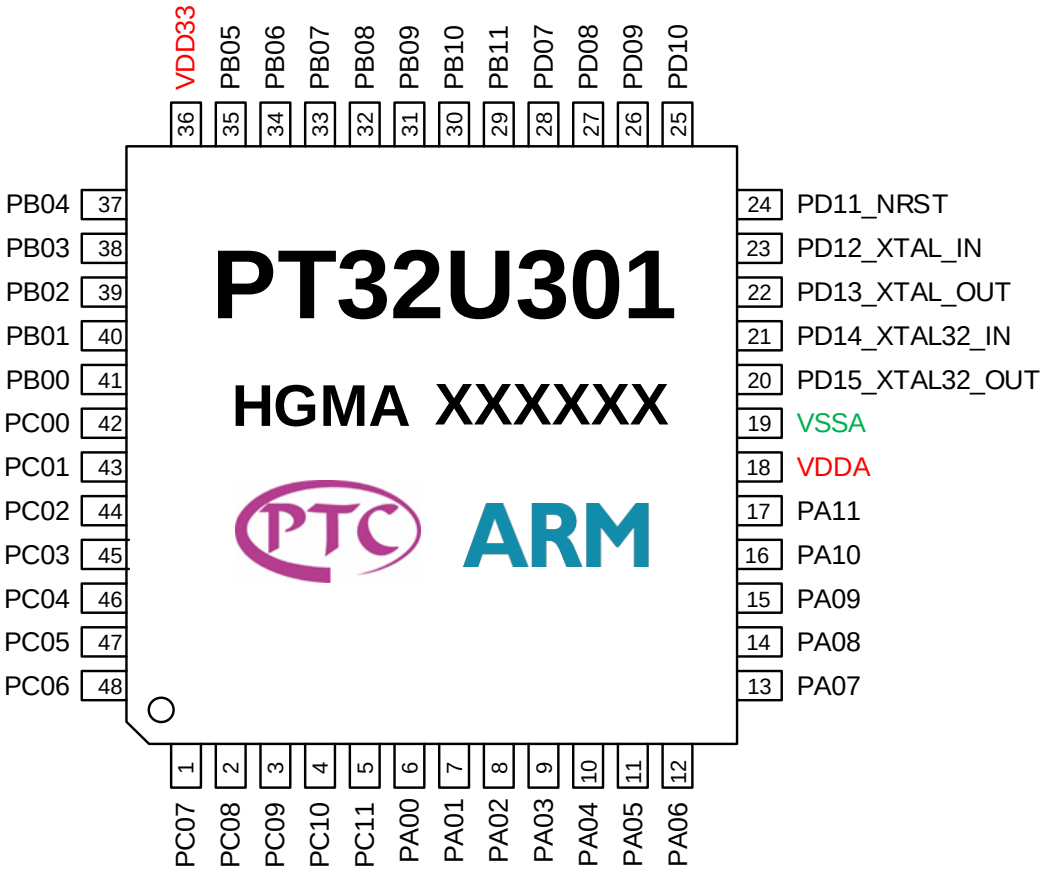
Package Type
G : LQFP, 7x7 mm^2
M : QFN, 4x4 mm^2
C : SSOP, 150 mil
X : SOP, 150 mil

Pin Count
M : 48 pins
R : 64 pins
W : 32 pins
P : 24 pins
B : 8 pins

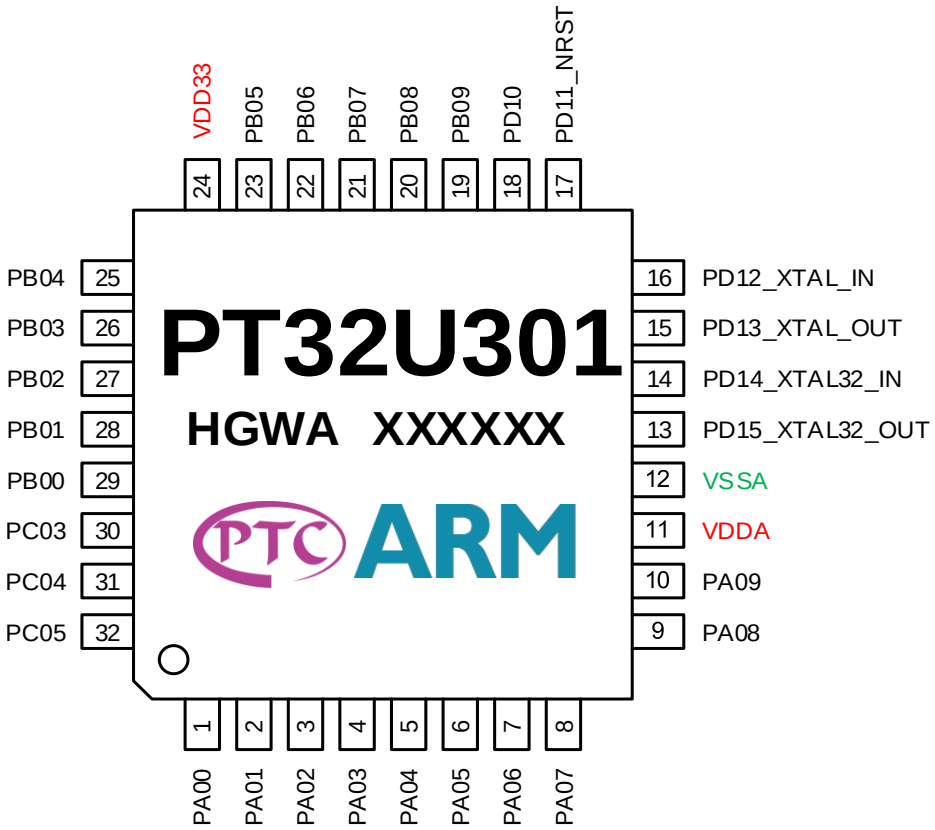
PKG Version

2. PIN CONFIGURATION

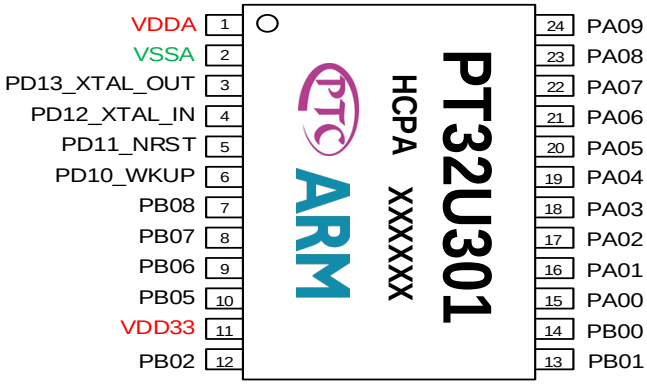
LQFP - 48



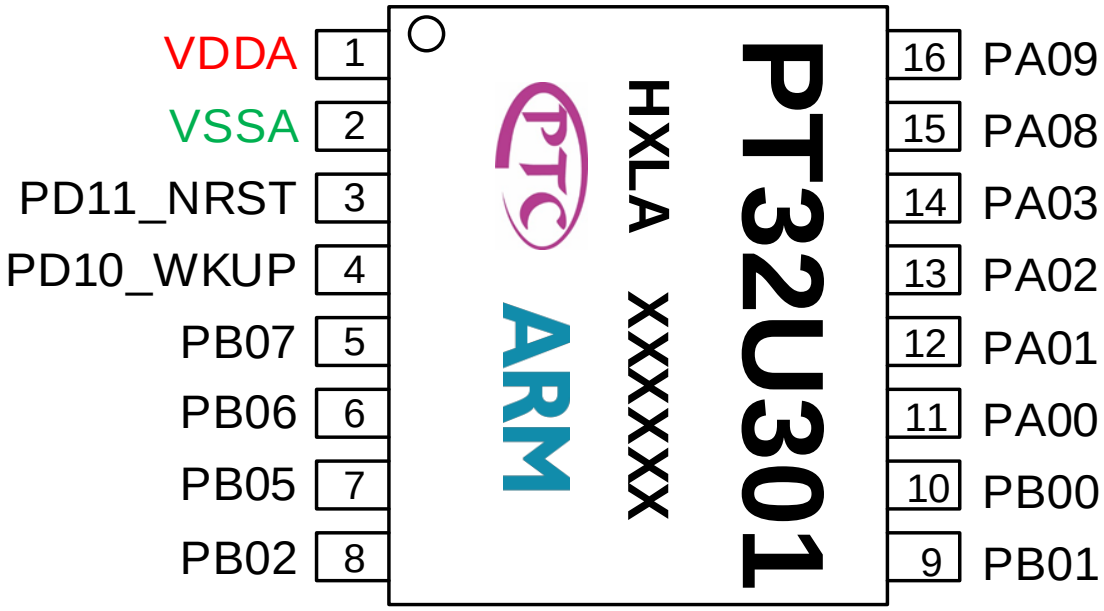
LQFP - 32



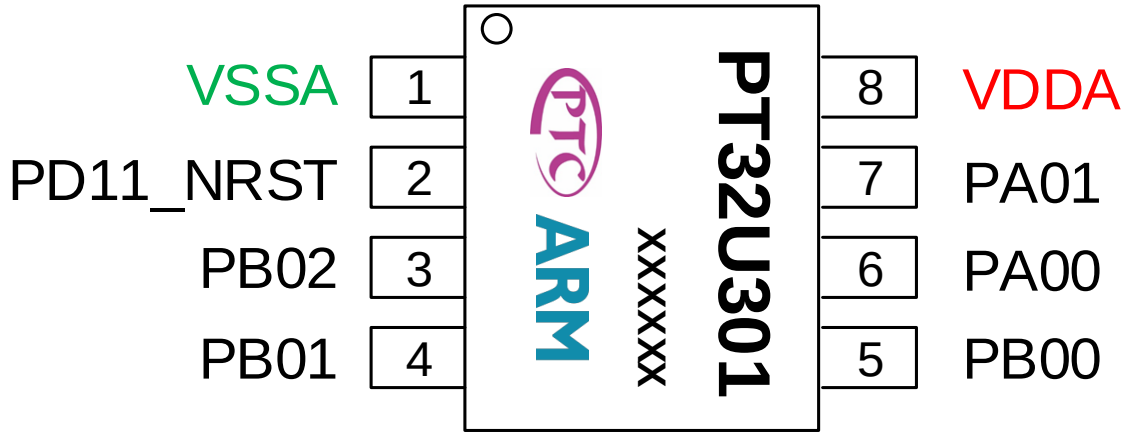
SSOP - 24



SOP - 16


VDDA 1
VSSA 2
PD11_NRST 3
PD10_WKUP 4
PB07 5
PB06 6
PB05 7
PB02 8
16 PA09
15 PA08
14 PA03
13 PA02
12 PA01
11 PA00
10 PB00
9 PB01
PT32U301
HXLA XXXXXX
ARM

SOP - 8


VSSA 1
PD11_NRST 2
PB02 3
PB01 4
8 VDDA
7 PA01
6 PA00
5 PB00
PT32U301
XXXXX
ARM

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.				
			LQFP48	LQFP32	SSOP24	SOP16	SOP8
PC07	I/O	General Purpose Digital I/O Pin	1				
PC08	I/O	General Purpose Digital I/O Pin	2				
PC09	I/O	General Purpose Digital I/O Pin	3				
PC10	I/O	General Purpose Digital I/O Pin	4				
PC11	I/O	General Purpose Digital I/O Pin	5				
PA00	I/O	General Purpose Digital I/O Pin	6	1	15	11	6
PA01	I/O	General Purpose Digital I/O Pin	7	2	16	12	7
PA02	I/O	General Purpose Digital I/O Pin	8	3	17	13	
PA03	I/O	General Purpose Digital I/O Pin	9	4	18	14	
PA04	I/O	General Purpose Digital I/O Pin	10	5	19		
PA05	I/O	General Purpose Digital I/O Pin	11	6	20		
PA06	I/O	General Purpose Digital I/O Pin	12	7	21		
PA07	I/O	General Purpose Digital I/O Pin	13	8	22	15	
PA08	I/O	General Purpose Digital I/O Pin	14	9	23	16	
PA09	I/O	General Purpose Digital I/O Pin	15	10	24		
PA10	I/O	General Purpose Digital I/O Pin	16				
PA11	I/O	General Purpose Digital I/O Pin	17				
VDDA	Supply	Analog Voltage Supply	18	11	1	1	8
VSSA	Ground	Analog Ground	19	12	2	2	1
PD15_XTAL32_OUT	I/O	General Purpose Digital I/O Pin/Crystal Output	20	13			
PD14_XTAL32_IN	I/O	General Purpose Digital I/O Pin/Crystal Input	21	14			
PD13_XTAL_OUT	I/O	General Purpose Digital I/O Pin/Crystal Output	22	15	3		
PD12_XTAL_IN	I/O	General Purpose Digital I/O Pin/Crystal input	23	16	4		
PD11_NRST	I/O	General Purpose Digital I/O Pin/Reset	24	17	5	3	2
PD10	I/O	General Purpose Digital I/O Pin	25	18	6	4	
PD09	I/O	General Purpose Digital I/O Pin	26	19			
PD08	I/O	General Purpose Digital I/O Pin	27	20			
PD07	I/O	General Purpose Digital I/O Pin	28				
PB11	I/O	General Purpose Digital I/O Pin	29				
PB10	I/O	General Purpose Digital I/O Pin	30				
PB09	I/O	General Purpose Digital I/O Pin	31				
PB08	I/O	General Purpose Digital I/O Pin	32		7		
PB07	I/O	General Purpose Digital I/O Pin	33	21	8	5	
PB06	I/O	General Purpose Digital I/O Pin	34	22	9	6	
PB05	I/O	General Purpose Digital I/O Pin	35	23	10	7	
VDD33	Supply	Voltage Supply	36	24	11		
PB04	I/O	General Purpose Digital I/O Pin	37	25			
PB03	I/O	General Purpose Digital I/O Pin	38	26			
PB02	I/O	General Purpose Digital I/O Pin	39	27	12	8	3

Pin Name	Pin Type	Description	Pin No.				
			LQFP48	LQFP32	SSOP24	SOP16	SOP8
PB01	I/O	General Purpose Digital I/O Pin	40	28	13	9	4
PB00	I/O	General Purpose Digital I/O Pin	41	30	14	10	5
PC00	I/O	General Purpose Digital I/O Pin	42				
PC01	I/O	General Purpose Digital I/O Pin	43				
PC02	I/O	General Purpose Digital I/O Pin	44				
PC03	I/O	General Purpose Digital I/O Pin	45	30			
PC04	I/O	General Purpose Digital I/O Pin	46	31			
PC05	I/O	General Purpose Digital I/O Pin	47	32			
PC06	I/O	General Purpose Digital I/O Pin	48				

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 AF6 functionality; In TSSOP 8-Pin Package, Respective PB [3:2] are defaulted to the AF6 functionality;

Table 3.1-1: Multiplexing Pins Function Selection

Pin Name	Alternate Functions								
	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	AF8
PC07								PWM1_B	
PC08								PWM0_B	
PC09								PWM2_A	
PC10								PWM1_A	
PC11								PWM0_A	
PA0	PA00			SPI_MISO	PWM1_A	CCP1_A			AD0_CAIP0
PA1	PA01	UART0_RTS	I2C_SDA	SPI_MOSI	PWM1_B	CCP1_B	UART1_RXD		AD1_CAIN0
PA2	PA02		UART1_TXD	SPI_MISO	PWM2_A	CCP2_A			AD2_CAIP1
PA3	PA03		UART1_RXD	SPI_MOSI	PWM2_B	CCP2_B			AD3_CAIN1
PA4	PA04	UART1_TXD			PWM1_A	CCP1_A			AD4
PA5	PA05	UART1_RXD			PWM1_B	CCP1_B			AD5
PA6	PA06	UART1_CTS			PWM2_A	CCP2_A			AD6
PA7	PA07	UART1_RTS			PWM2_B	CCP2_B			AD7
PA8	PA08	UART1_TXD	I2C_SDA		PWM0_A	CCP0_A			CAIP2
PA9	PA09		UART1_RTS	SPI_SCSN	PWM0_B	CCP0_B			CAIN2
PA10	PA10								CAIP3
PA11	PA11								CAIN3
VDDA									
VSSA									
PD15 XTAL32_OUT	PD15								XTAL32_OUT(1)
PD14_ XTAL32_IN	PD14								XTAL32_IN(1)
PD13_ XTAL_OUT	PD13								XTAL_OUT(1)
PD12_ XTAL_IN	PD12								XTAL_IN(1)
PD11_ NRST	PD11			SPI_SCSN	PWM_FALT	CCP1_B	PWM1_B		NRST(1)
PD10	PD10				PWM_FALT				WKUP_V33
PD09	PD09			SPI_SCSN					
PD08	PD08			SPI_SCKK					
PD07	PD07								
PB11	PB11								
PB10	PB10			SPI_SCKK				PWM2_B	
PB09	PB09			SPI_MISO				PWM1_B	
PB08	PB08		UART1_TXD	SPI_MOSI	PWM_FALT			PWM0_B	
PB07	PB07	I2C_SDA	UART1_RXD	SPI_SCSN	PWM2_B			PWM2_A	

Pin Name	Alternate Functions								
PB06	PB06	UART0_CTS	UART1_CTS	SPI_MOSI	PWM2_A	UART0_TXD	SWCLK(2)	PWM1_A	
PB05	PB05	UART0_RTS	UART1_RTS	SPI_MISO	PWM1_B	UART0_RXD	SWDA(2)	PWM0_A	
VDD33									
PB04	PB04			SPI_MISO		UART0_TXD			
PB03	PB03			SPI_MOSI		UART0_RXD	SWCLK(2)		
PB02	PB02	UART0_CTS	I2C_SCK	SPI_SCKK	PWM1_A	CCP1_A	SWDA(2)	MCO	
PB01	PB01	UART0_TXD	I2C_SCK	SPI_SCSN	PWM0_B	CCP0_B			
PB00	PB00	UART0_RXD	I2C_SDA	SPI_MISO	PWM0_A	CCP0_A			
PC00	PC00				PWM_FALT		RTC_1HZ	PWM2_B	
PC01	PC01							PWM1_B	
PC02	PC02							PWM0_B	
PC03	PC03							PWM2_A	
PC04	PC04							PWM1_A	
PC05	PC05							PWM0_A	
PC06	PC06							PWM2_B	

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
VDD33	Supply	Digital Domain 3.3V Power Supply
VSSA	Ground	Ground
VSS	Ground	Ground Voltage
Universal Asynchronous Receiver/Transmitter (UART0, UART1), x = 0, 1		
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 Intergrated Circuit (I2C)		
I2C_SDA	I/O	I2C Data
I2C_SCK	I/O	I2C Clock
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
General Purpose Input/Output (GPIOA, GPIOB, GPIOC, GPIOD)		
PA11-PA00	I/O	GPIO Port A
PB11-PB00	I/O	GPIO Port B

Signal Name	I/O	Function Description
PC11-PC00	I/O	GPIO Port C
PD15-PD07	I/O	GPIO Port D
Pulse Width Modulation (PWM0, PWM1, PWM2), x = 0, 1, 2		
PWMx_A	O	PWM x Signals
PWMx_B	O	PWM x Signals
PWM_FALT	I	PWM Fault Input
General-Purpose Timer (GPT0, GPT1, GPT2), x = 0, 1, 2		
CCPx_A	I/O	GPTimer x Compare and Capture A
CCPx_B	I/O	GPTimer x Compare and Capture B
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
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
WKUP	I	Wakeup
MCO	O	Microcontroller Clock Output
RTC_CLK	O	RTC Clock Output

4. FUNCTIONAL DESCRIPTION

4.1 SYSTEM AND MEMORY OVERVIEW

The PT32U301 microcontrollers is a series of low-power microcontrollers incorporating a high-performance ARM CortexTM-M0 32-bit RISC core operating at a 48 MHz frequency, high-speed embedded memories and an extensive range of enhanced peripherals and I/O s. A comprehensive set of power-saving modes allows it to be employed in low-power applications.

The PT32U301 MCUs give you any essential functionality as a General-purpose MCU. With its highly customizable peripherals, it eases the process of making your own ideal product.

This chapter introduces you to PT32U301 features, its system and memory structure.

4.1.1 PT32U301 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

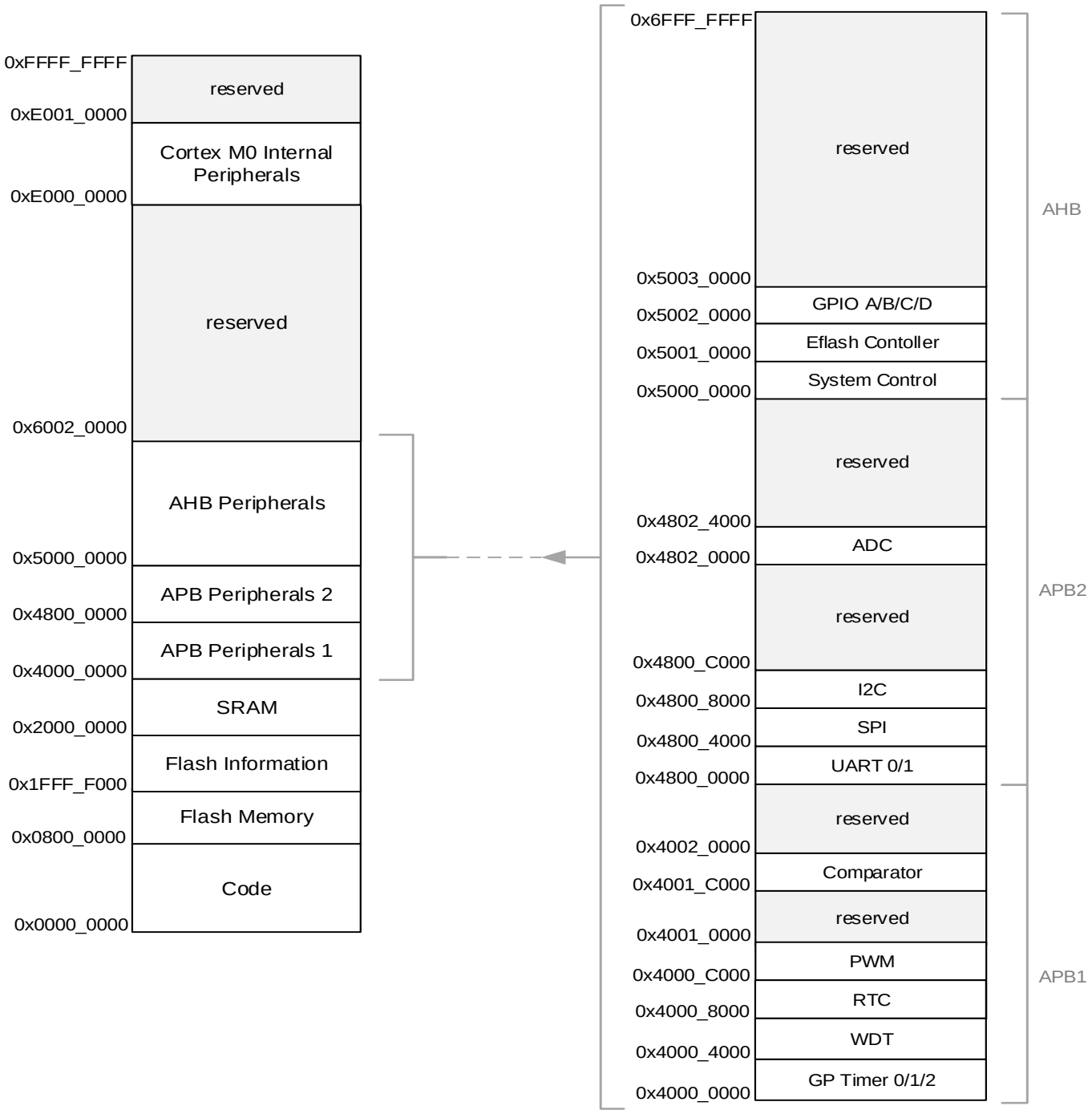


Table 4.1-1: Peripheral register boundary addresses

Boundary address		Depth (Byte)	Peripheral Description	Reference Section	BUS
Start	End				
0x0000_0000	0x1FFF_FFFF	6K	Mask ROM, Main Flash memory or System RAM depending on Booting Configuration	-	AHB
0x0800_0000	0x0800_7FFF	32K	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		
0x2000_1000	0x3FFF_FFFF	-	Reserved	-	
0x4000_0000	0x4000_3FFF		General-purpose Timer 0/1/2 Control Register		APB1
0x4000_4000	0x4000_7FFF		Watchdog Control Register		
0x4000_8000	0x4000_BFFF		RTC Control Register		
0x4000_C000	0x4000_FFFF		Pulse Width Modulation (PWM) Control Register		
0x4001_0000	0x4001_BFFF	-	Reserved	-	
0x4001_C000	0x4001_FFFF		Analog Comparator(AC) Control Register		
0x4002_0000	0x47FF_FFFF	-	Reserved	-	
0x4800_0000	0x4800_3FFF		UART 0/1 Control Register	4.6	APB2
0x4800_4000	0x4800_7FFF		SPI Control Register	4.15	
0x4800_8000	0x4800_BFFF		I2C Control Register	4.14	
0x4800_C000	0x4801_FFFF		Reserved	-	
0x4802_0000	0x4802_3FFF		ADC Control Register	4.8	
0x4802_4000	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	0x6FFF_FFFF		Reserved	-	
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 PT32U301, this processor configures following features:

- Built-in Nested Vectored Interrupt Controller (NVIC): 32 external Interrupt
- Little-endian
- Intergrated 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.

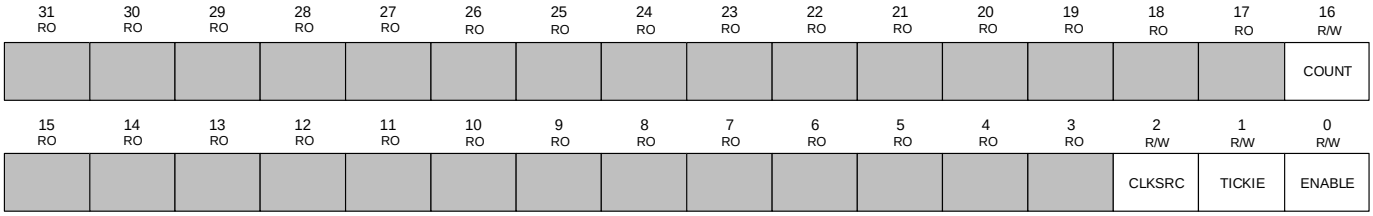
4.2.2 SYST REGISTER MAPS

Base Address: 0xE000_E000

Offset	Symbol	Type	Reset Value	Description	See page
0x0010	CSR	R/W	0x0000_0000	SysTick Control and Status Register	18
0x0014	RVR	R/W	-	SysTick Reload Value Register	19
0x0018	CVR	R/W	-	SysTick Current Value Register	19

4.2.2.1 SYST_CSR - SYSTICK CONTROL AND STATUS REGISTER

The SYST_CSR enables the SysTick features.

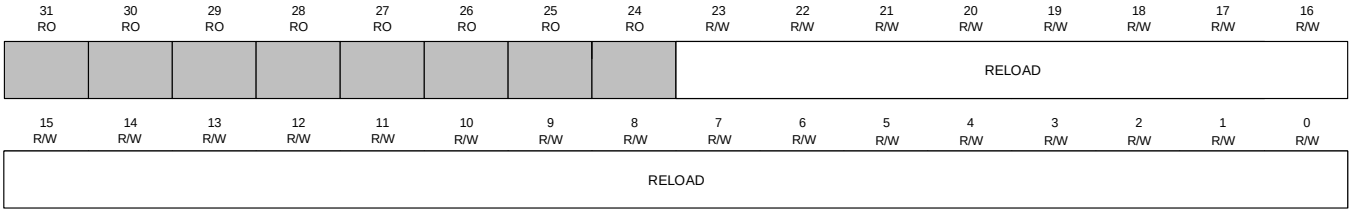


Offset: 0x0010

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.

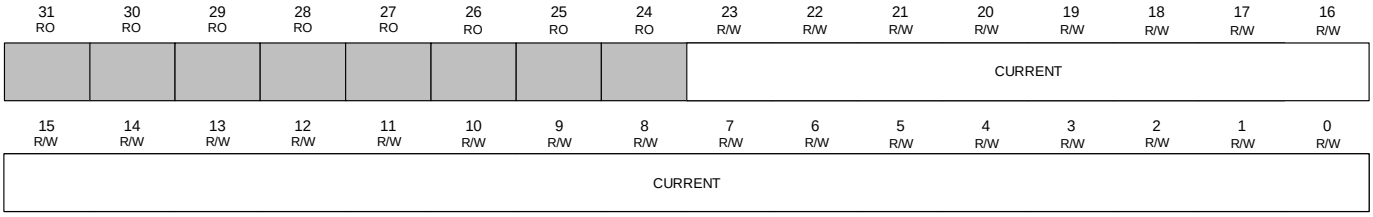


Offset: 0x0014

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
29-19	Reserved	-
18	ADC	ADC Interrupt
17	COMP	Analog Comparator Interrupt
16	PWM_FAULT	PWM Fault Interrupt
15	PWM2	PWM2Interrupt
14	PWM1	PWM1Interrupt
13	PWM0	PWM0 Interrupt
12	TIMER2	TIMER2 Interrupt
11	TIMER1	TIMER1 Interrupt
10	TIMER0	TIMER0 Interrupt
9	WDT	Watch Dog Interrupt
8	RTC	RTC Interrupt
7	I2C	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	23
0x0080	NVIC_ICER	R/W	0x0000_0000	IRQ Clear Enable Control Register	23
0x0100	NVIC_ISPR	R/W	0x0000_0000	IRQ Set Pending Control Register	24
0x0180	NVIC_ICPR	R/W	0x0000_0000	IRQ Clear Pending Control Register	24
0x0300	NVIC_IPR0	R/W	0x0000_0000	IRQ0 - IRQ3 Priority Control Register	25
0x0304	NVIC_IPR1	R/W	0x0000_0000	IRQ4 - IRQ7 Priority Control Register	26
0x0308	NVIC_IPR2	R/W	0x0000_0000	IRQ8 - IRQ11 Priority Control Register	27
0x030C	NVIC_IPR3	R/W	0x0000_0000	IRQ12 - IRQ15 Priority Control Register	28
0x0310	NVIC_IPR4	R/W	0x0000_0000	IRQ16 - IRQ19 Priority Control Register	29
0x0314	NVIC_IPR5	R/W	0x0000_0000	IRQ20 - IRQ23 Priority Control Register	30
0x0318	NVIC_IPR6	R/W	0x0000_0000	IRQ24 - IRQ27 Priority Control Register	31
0x031C	NVIC_IPR7	R/W	0x0000_0000	IRQ28 - IRQ31 Priority Control Register	32

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
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															

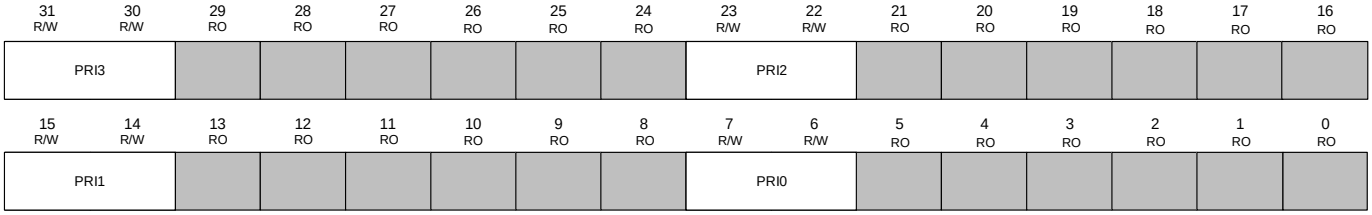
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	**R/W**	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
CLRPEND															

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

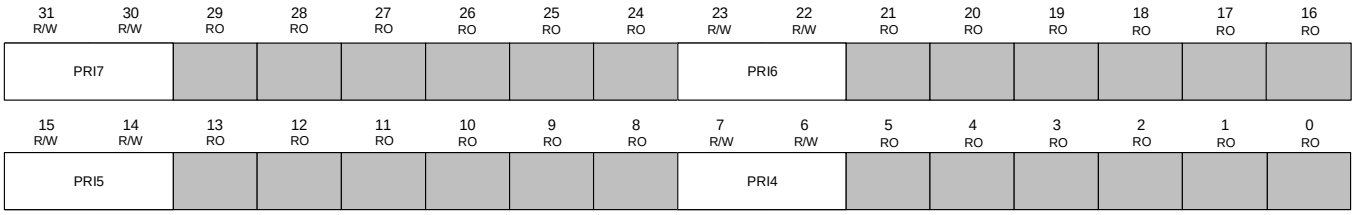


Offset: 0x0300

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.

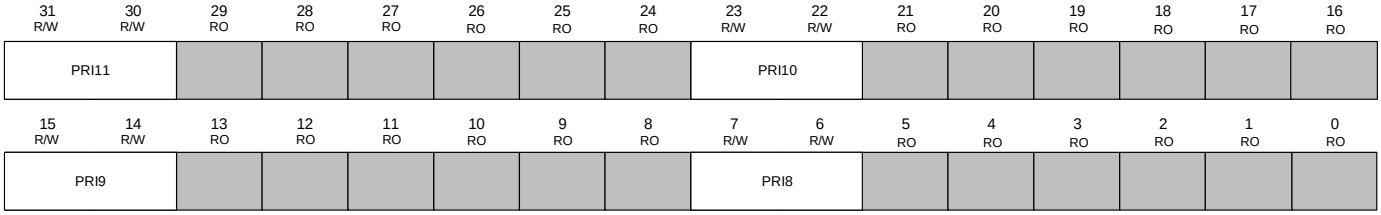


Offset: 0x0304

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

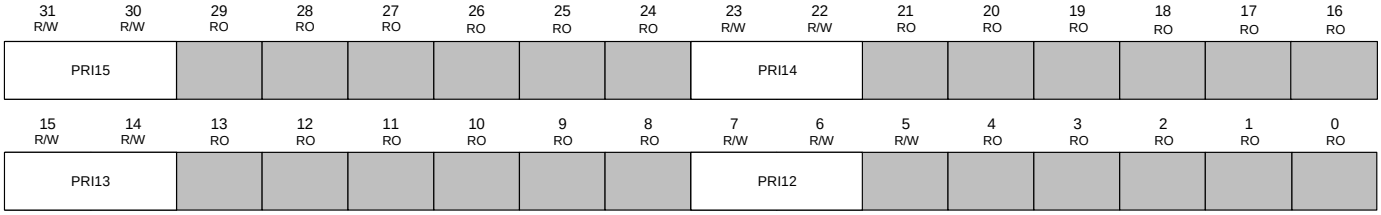


Offset: 0x0308

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

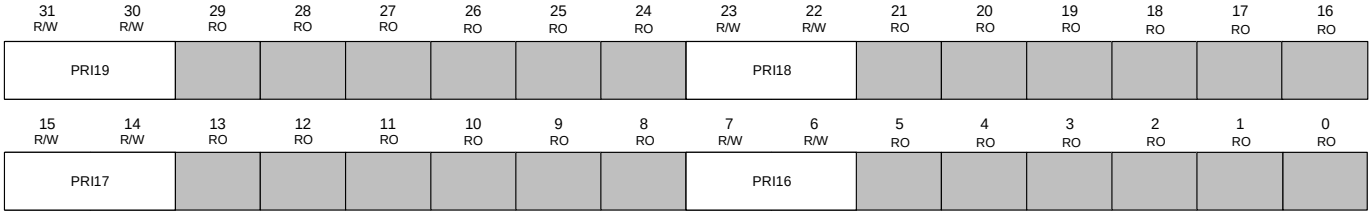


Offset: 0x030C

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

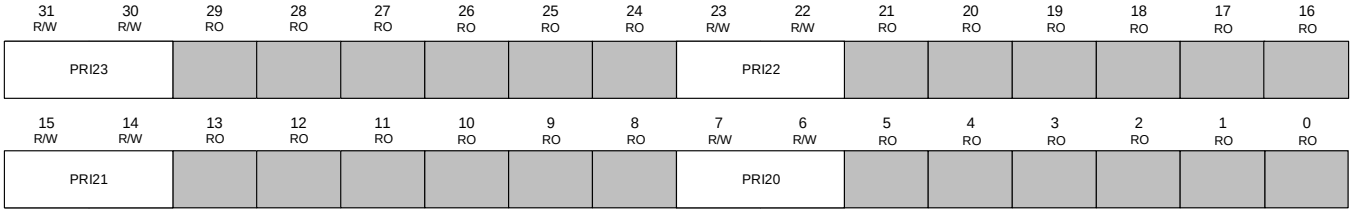


Offset: 0x0310

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.

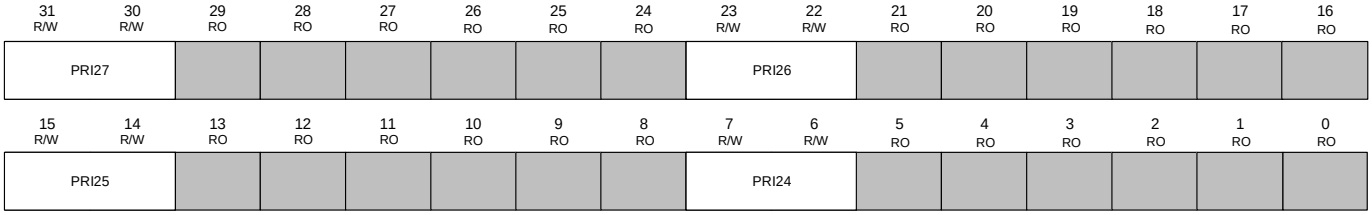


Offset: 0x0314

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.

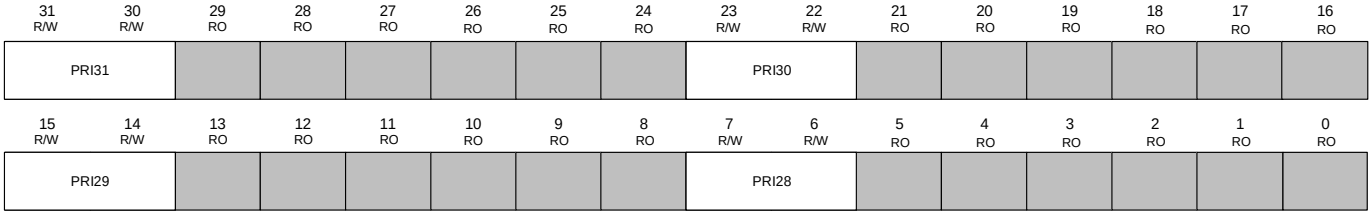


Offset: 0x0318

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.

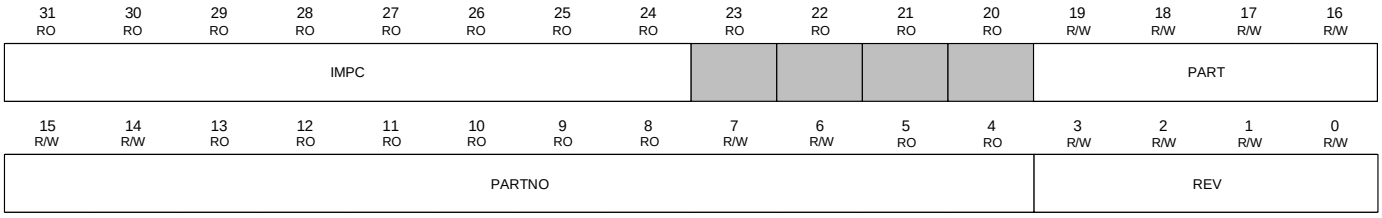
4.3.1 CPU SYSTEM CONTROL REGISTER MAP

Base Address 0xE000_ED00

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	SYS_CPUID	R/W	0x410C_C200	System CPUID Register	33
0x0004	SYS_ICSR	R/W	0x0000_0000	System Interrupt Control and State Register	34
0x000C	SYS_AIRCR	R/W	0xFA05_0000	System Application Interrupt and Reset Control Register	35
0x0010	SYS_SCR	R/W	0x0000_0000	System Control Register	36
0x001C	SYS_SHPR2	R/W	0x0000_0000	System Handler Priority Register 2	37
0x0020	SYS_SHPR3	R/W	0x0000_0000	System Handler Priority Register 3	37

4.3.1.1 SYS_CPUID – CPU ID REGISTER

This register provide identification for the processor.

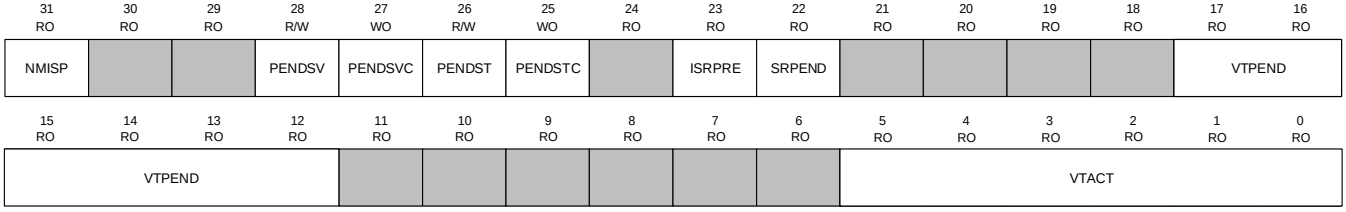


Offset: 0x0000

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.



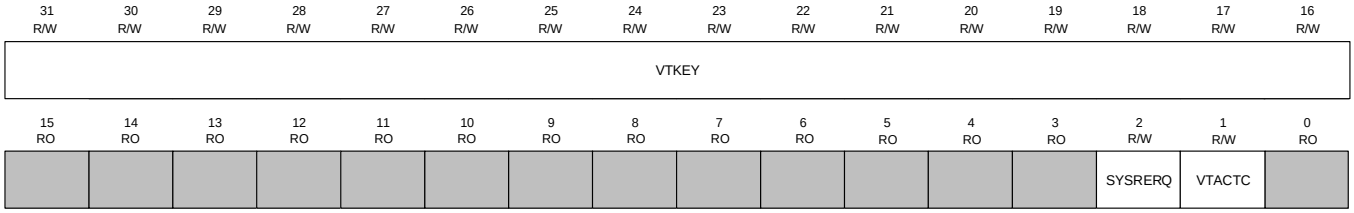
Offset: 0x0004

Bits	Name	Type	Reset	Description
31	*NMISP*	RO	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.

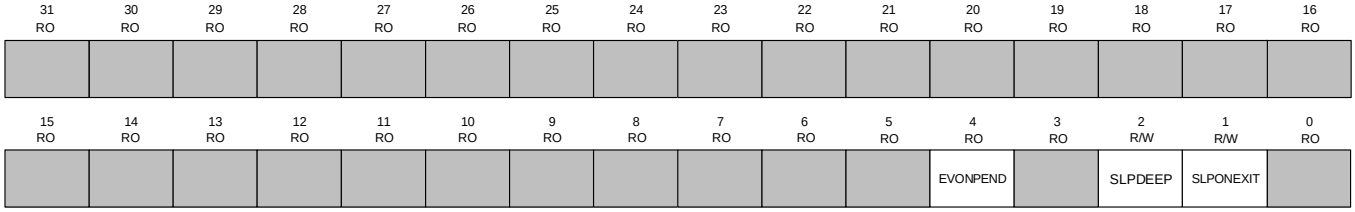


Offset: 0x000C

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

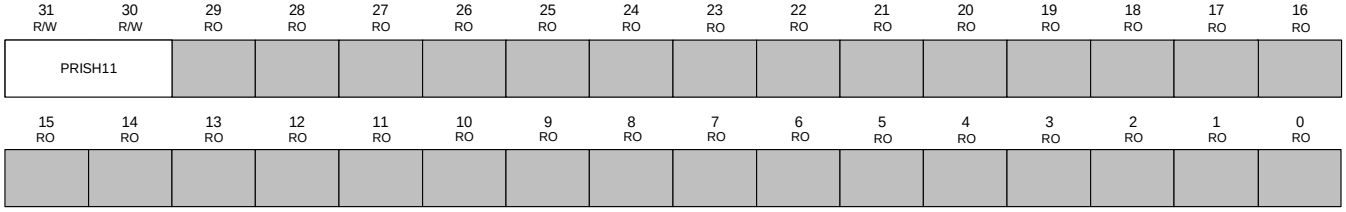
The SCR controls features of entry to and exit from low power state.



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*	RO	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	*SLPONEXIT*	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.

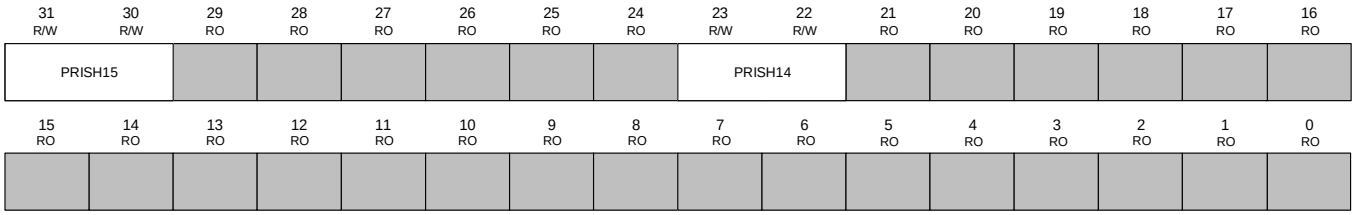
4.3.1.5 SYS_SHPR2 - SYSTEM HANDLER PRIORITY REGISTER 2



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

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_ICE.**

4.4.1.3 BOOTING CONFIGURATION

In the PT32U30x, 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	Mode*
REMAP bit	
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

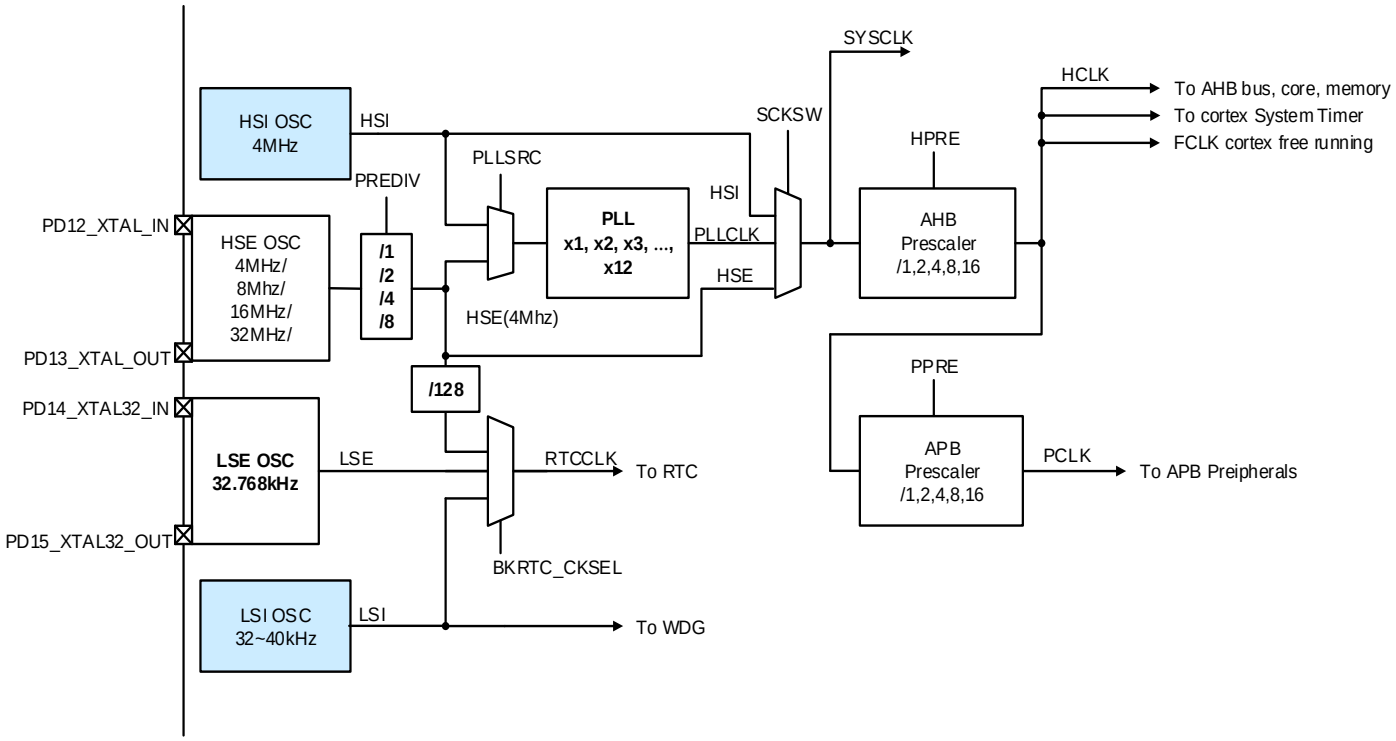
*: The boot mode configuration is sampled in a power-on reset or a system reset.

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

- Boot from Flash Information: the flash information memory is sliased 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), but still accessible from its original memory space (0x0800 0000).
- 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



4.4.1.5 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.
- HSE Clock
  - High-Speed External Clock (HSE) signal is generated from an external crystal oscillator. This clock can keep running in 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.

4.4.1.6 CLOCK DISTRIBUTION

The clock system on the PT32U301 consists of:

System Clock - SYSCLK

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

- HSI 4 MHz RC oscillator clock
- PLL clock
- HSE oscillator clock (4MHz)

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).**

- HSE 4Mhz divided by 128
- LSE 32.768kHz oscillator clock
- LSI RC oscillator clock

HCLK, PCLK

The AHB clock (HCLK) is derived from System clock through configurable prescalaer 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.

4.4.1.7 POWER

VOLTAGE REGULATOR

A voltage regulator is embedded in PT32U301 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

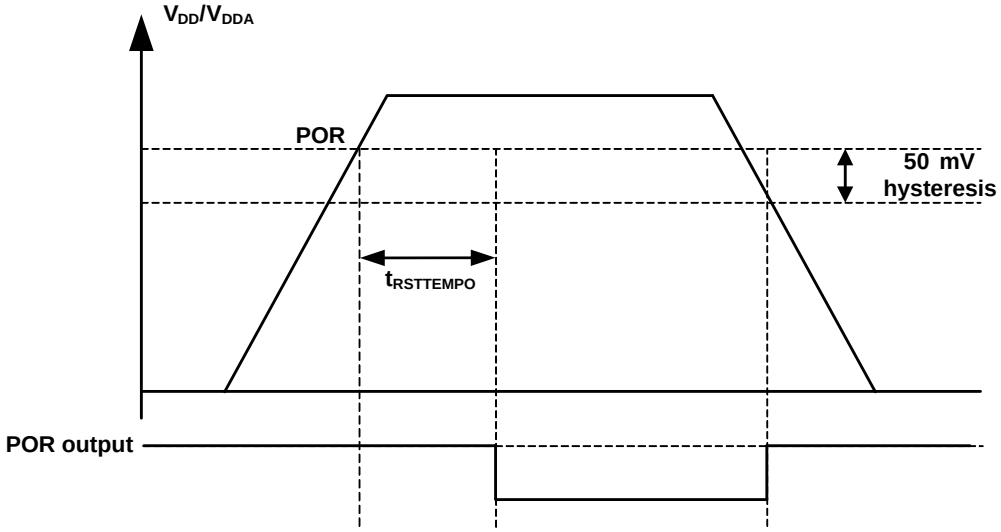
This device has three ways to monitor and reset the device:

- 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

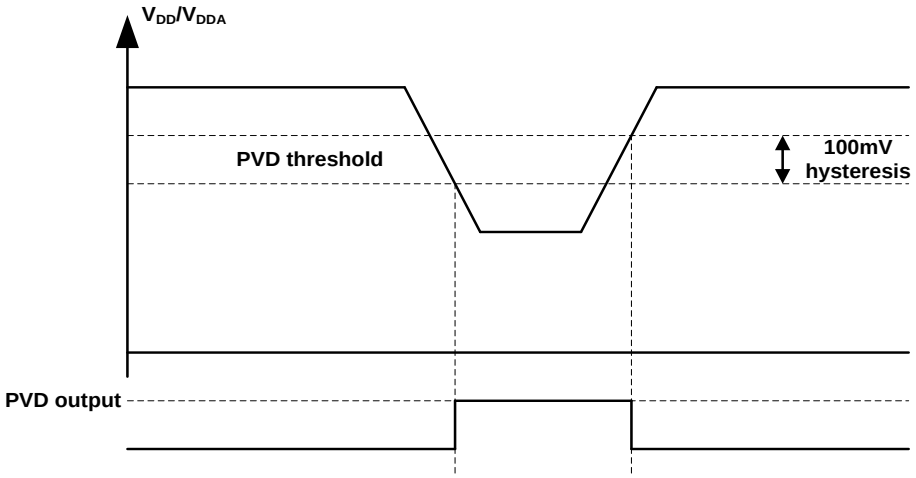


In PT32U301, the Power-on reset (NRST) must be provided externally (pin PD11_NRST). To guarantee proper power on and power down reset to the device, the PD11_NRST pin must be held low until V_{DD33} is stable or before turning off the supply. When V_{DD33} is stable, the reset state can be excited by putting the PD11_NRST pin in high impedance. The PD11_NRST pin has an internal pull-up connected to V_{DD}.

Programmable voltage detector (PVD)

PVD is used tp 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



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 low-power modes:

- Sleep mode (CPU clock off, all peripherals including Cortex®-M0 core peripherals like NVIC, SysTick, etc. are kept running)
- Stop mode (all clocks are stopped)
- Standby mode (1.8V domain powered-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, 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 or WFE 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.

Sleep mode

ENTERING SLEEP MODE

The Sleep mode is entered by executing the WFI (Wait for Interrupt) instruction. Two options are available to select the Sleep mode entry mechanism, depending on the SLPONEXIT bit in the Cortex®-M0 System Control register

- Sleep-now: if the SLPONEXIT bit is cleared, the MCU enters Sleep mode as soon as WFI instruction is executed.
- Sleep-on-exit: if the SLPONEXIT bit is set, the MCU enters Sleep mode as soon as it exits the lowest priority ISR.

In the Sleep mode, all I/O pins remain in the same state as in the Run mode

EXITING SLEEP MODE

If the WFI instruction is used to enter Sleep mode, any peripheral interrupt acknowledged by the nested vectored interrupt controller (NVIC) can wake up the device from Sleep mode.

STOP MODE

The Stop mode is based on the Cortex®-M0 **deepsleep** mode combined with peripheral clock gating. The voltage regulator can be configured either in **normal** or **low-power** mode. In Stop mode, all clocks can be switched off before CPU entering deepsleep mode. The system can save maximum power in this mode by switching of PLL/HSI/HSE/ADC off. And lower the system clock speed by switching the clock source to LSI or LSE. After clock has be switch to lower frequency, the system can also setup LDO into LDO Low Power mode. Without switching the *LDOLP* or *LDOOFF* bit, the Stop Mode is almost identical with Sleep Mode. The user must remember to switch off unnecessary peripheral before Stop the CPU.

ENTERING STOP MODE

To further reduce power consumption in Stop mode, the internal voltage regulator can be put in low-power mode. This is configured by the *LDOLP* bit of the **SC_BKSLP_CTRL**. The ADC can also consume power during Stop mode, unless they are disabled before entering this mode

EXITING STOP MODE

When the voltage regulator operates in low-power mode, an additional startup delay is incurred when waking up from Stop mode. By keeping the internal regulator ON during Stop mode, the consumption is higher although the startup time is reduced.

STANDBY MODE

The Standby mode allows to 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 RTC domain and Standby circuitry.

ENTERING STANDBY MODE

To further reduce power consumption in Stop mode, the internal voltage regulator can be switch off. This is configured by the *LDOOFF* bit of the **SC_BKSLP_CTRL**

EXITING STANDBY MODE

The PT32U301 exits the Standby mode when an external reset (NRST pin), a detected falling edge on the enabled WKUP pins (i.e. PD10) or an RTC event occurs. All registers are reset after wakeup from Standby except backup domain.

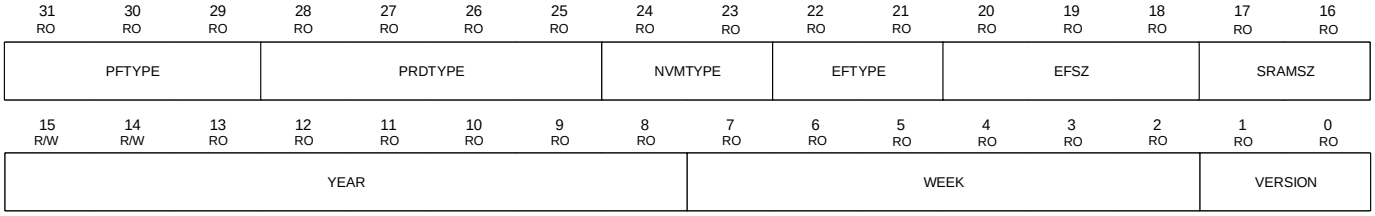
After waking up from Standby mode, program execution restarts in the same way as after a Reset. The *WKUPFLAG* status flag in the Backup RTC Sleep Controller Register (**SC_BK_STATUS**) indicates that the MCU was in Standby mode.

4.4.2 SYSTEM CONTROL REGISTER MAP

Base Address: 0x5000_0000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	SC_PID0	RO	0x0188_0F5C	Product ID0 number Register	45
0x0004	SC_PID1	R/W	0x0000_00E0	Product ID0 number Register	46
0x0008	SC_UID0	R/W	0x0000_0000	User define field ID0 Register	47
0x000C	SC_UID1	R/W	0x0000_0000	User define field ID1 Register	47
0x0010	SC_REMAP	R/W	0x0000_0F06	Internal remap by software configuration	48
0x0014	SC_ICE	R/W	0x0000_0000	ICE Protection	49
0x0018	SC_STCALIB	R/W	0x0100_0148	System Tick Timer Calibration Value	49
0x001C	SC_NVM_ACR	R/W	0x0000_0040	NVM (FLASH) Access Control Register _	50
0x0020	SC_GCLK_APB	R/W	0x0000_0000	APB Peripheral Clock Gating Enable Register	51
0x0024	SC_GCLK_AHB	R/W	0x0000_0000	AHB Peripheral Clock Gating Enable Register	52
0x0028	SC_RST_APB	R/W	0x0000_0000	APB Peripheral Reset Request Register	53
0x002C	SC_RST_AHB	R/W	0x0000_0000	AHB Peripheral Reset Request Register	55
0x0030	SC_CK_CTRL	R/W	0x0000_0001	Clock Control Register	56
0x0034	SC_CK_CONF	R/W	0x0000_000B	Clock Configuration Register	57
0x003C	SC_CK_STAT	RO	0x0000_0000	Clock Status Register	59
0x0040	SC_BK_CTRL	R/W	0x0000_0000	Backup Control Register	60
0x0048	SC_PWR_CTRL	R/W	0x0000_0000	Power Control Register	60
0x004C	SC_PVD_DET	R/W	0x0000_000E	PVD Detection	61
0x0050	SC_SLP_APB	R/W	0x0000_0000	Into Sleep APB Peripheral Clock Gating Enable Register	62
0x0054	SC_SLP_AHB	R/W	0x0000_0000	Into Sleep AHB Peripheral Clock Gating Enable Register	63
0x0058	SC_DSLP_APB	R/W	0x0000_0000	Into Deep Sleep APB Peripheral Clock Gating Enable Register	64
0x005C	SC_DSLP_AHB	R/W	0x0000_0000	Into Deep Sleep AHB Peripheral Clock Gating Enable Register	65
0x006C	SC_OSC_TRIM	R/W	0x0000_0000	Ring OSC Hardware Trimming Register	66
0x0080	SC_BKRTC_CTRL	R/W	0x00FF_7F00	Backup RTC Controller Register	67
0x0084	SC_BKSLP_CTRL	R/W	0x00FF_FFFF	Backup Sleep Controller Register	68
0x0088	SC_BK_STAT	RO	0x0000_0000	Backup Status Register	68
0x0090	SC_BK_REG0	R/W	0x0000_0000	Backup 0 Register	69
0x0094	SC_BK_REG1	R/W	0x0000_0000	Backup 1 Register	69
0x0098	SC_BK_REG2	R/W	0x0000_0000	Backup 2 Register	69
0x009C	SC_BK_REG3	R/W	0x0000_0000	Backup 3 Register	69
0x00A0	SC_BK_REG4	R/W	0x0000_0000	Backup 4 Register	69
0x00A4	SC_BK_REG5	R/W	0x0000_0000	Backup 5 Register	69
0x00A8	SC_BK_REG6	R/W	0x0000_0000	Backup 6 Register	69
0x00AC	SC_BK_REG7	R/W	0x0000_0000	Backup 7 Register	69

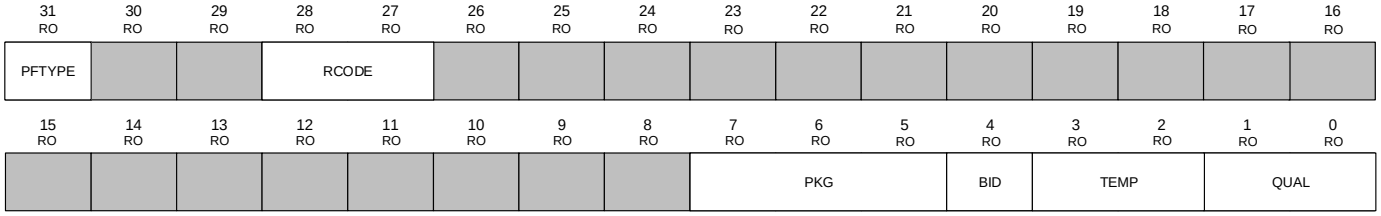
4.4.2.1 SC_PID0 - PRODUCT ID0 NUMBER REGISTER



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:29	*PFTYPE*	RO	0x0	Platform Type 0x0 : Cortex-M0 MCU Platform 0x1 : Cortex-M3 MCU Platform 0x2 : Cortex-M4 MCU Platform 0x3 : ASIC 0x4 : Others
28:25	*PDTYPE*	RO	0x2	Product Type 0x0 : General Propose MCU 0x1 : BLDC Application 0x2 : USB Application 0xE : EMC Application 0xF : TSC Application
24:23	*NVMTYPE*	RO	0x3	Non-Volatile Memory Type 0x0 : OTP 0x1 : MTP 0x2 : EEPROM 0x3 : Embedded Flash
22:21	*EFTYPE*	RO	0x0	Eflash Type 0: Single Embedded Flash Memory 1: Dual Embedded Flash Memory
20:18	*EFSZ*	RO	0x3	Embedded Flash Size 0x0 : 8K 0x1 : 16K 0x2 : 32K 0x3 : 64K Byte 0x4 : 128 Kbyte(Reserved)
17:16	*SRAMSZ*	RO	0x1	SRAM Size 4X(SRAM_SZ+1) K Byte
15:8	*YEAR*	RO	15	Release Year
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

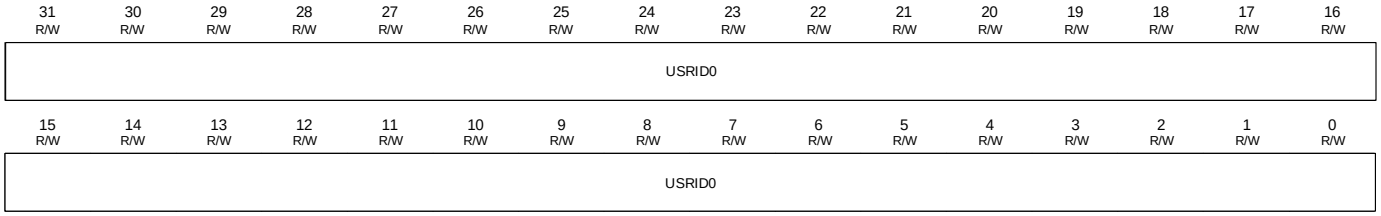


Offset: 0x0004

Bits	Name	Type	Reset	Description
31	*FPT*	WO	0	Flash Protect bit 0: Flash Data is protected 1: Flash Data is not being protected
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.
28:27	*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	0x0	Software should not 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*	RO	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*	RO	0	Bounding Wire Type 0: Gold 1: Copper
3:2	*TEMP*	RO	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*	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.

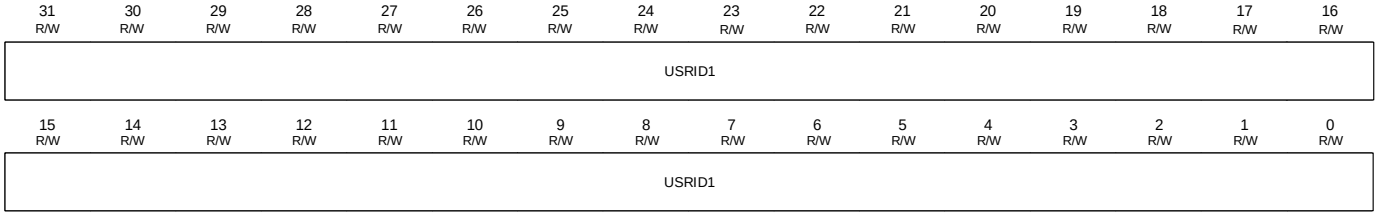


Offset: 0x0008

Bits	Name	Type	Reset	Description
31:0	*USRID0*	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.

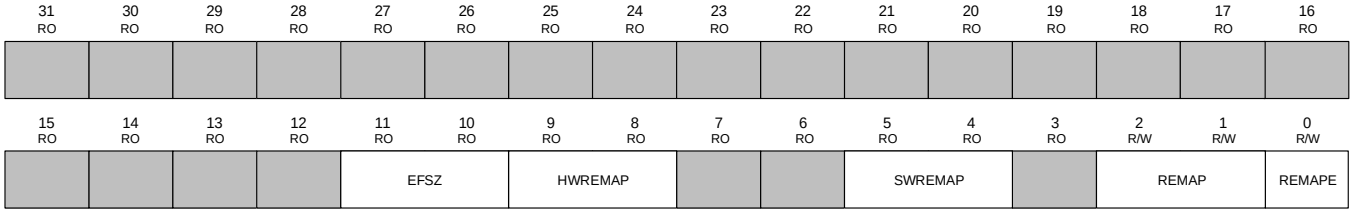


Offset: 0x000C

Bits	Name	Type	Reset	Description
31:0	*USRID1*	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. User can configure the software memory organization via the bit **REMAP** in this register.

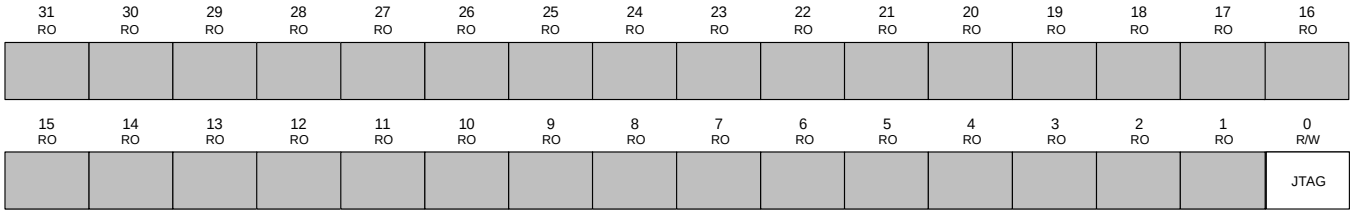


Offset: 0x0010

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.
11:10	*EFSZ*	RO	0x3	Maximum EFlash Size 0x0 : 16K 0x1 : 32K 0x3 : 64K
9:8	*HWREMAP*	RO	0x3	External Hardware Remap Status 0x2 : Eflash 0x3 : Mask ROM
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	*SWREMAP*	RO	0x3	Internal Remap Selected Status 0x0 : Eflash Information 0x1 : SRAM 0x2 : Eflash Memory 0x3 : Mask ROM
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.
2:1	*REMAP*	R/W	0x3	Internal Remap Select 0x0 : Eflash Information 0x1 : SRAM 0x2 : Eflash Memory 0x3 : Mask ROM
0	*REMAPE*	R/W	0	Internal Remap Enable 1: Enter the remapping flow This bit is self-cleared

4.4.2.6 SC_ICE – ICE PROTECTION

This register will be active, if Eflash Info 0x7F8:0 JTAG protect switch is disabled.

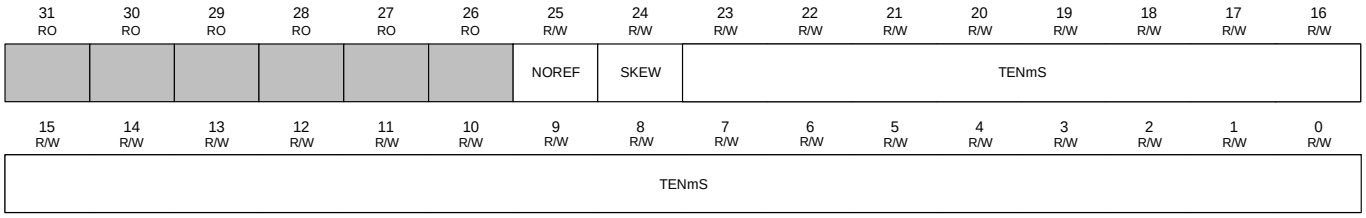


Offset: 0x0014

Bits	Name	Type	Reset	Description
31:1	*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.
0	*ICE*	R/W	0	Protect embedded and turn off JTAG function 0: disable JTAG protect 1: Enable JTAG protect (Disable JTAG)

4.4.2.7 SC_STCALIB - SYSTEM TICK TIMER CALIBRATION VALUE

The SC_STCALIB register indicates the SysTick calibration properties



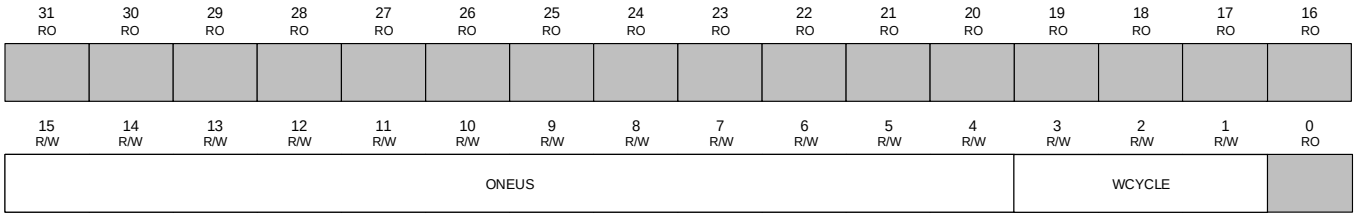
Offset: 0x0018

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.

In Run mode, the HCLK and PCLKx for individual peripherals and memories can been turn off 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 or WFE instructions.

4.4.2.8 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.



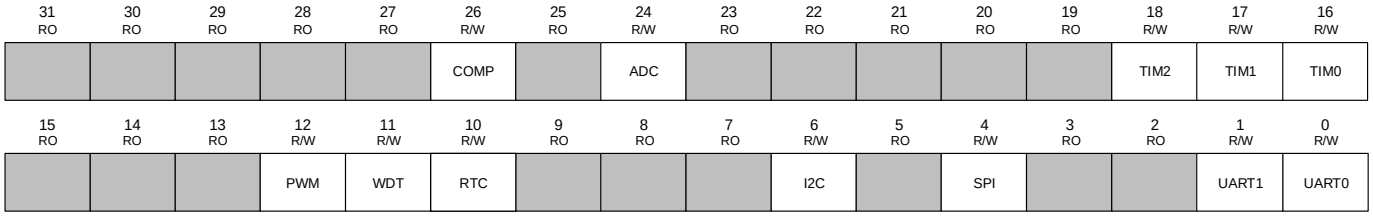
Offset: 0x001C

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.
3:1	*WCYCLE*	R/W	0x0	Waiting cycle to read data from embedded flash **HCLK Frequency (*f*)** / ***WCYCLE* Value** *f* ≤ 24MHz : 0x0 24MHz < *f* ≤ 48 MHz : 0x1 48 MHz < *f* ≤ 72 MHz : 0x2 *Note: The highest frequency of HCLK in PT32U301 is 72 MHz, therefore, the value of WCYCLE should not be larger than 0x2.*
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.

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.

4.4.2.9 SC_GCLK_APB - APB PERIPHERAL CLOCK GATING ENABLE REGISTER

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



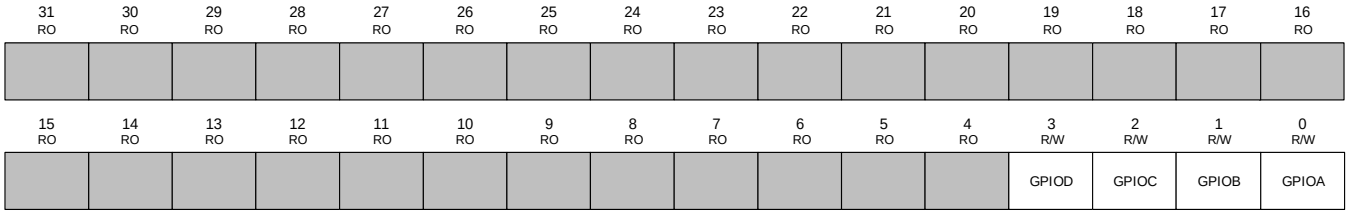
Offset: 0x0020

Bits	Name	Type	Reset	Description
31:27	*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	*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.
23:19	*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	*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.
15:13	*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.
12	*PWM*	R/W	0	PWM Clock Gating Control This bit controls the clock gating for PWM module 0.
11	*WDT*	R/W	0	WDT Clock Gating Control. This bit controls the clock gating for WDT module.
10	*RTC*	R/W	0	RTC Clock Gating Control This bit controls the clock gating for RTC module.
9:7	*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	*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_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.

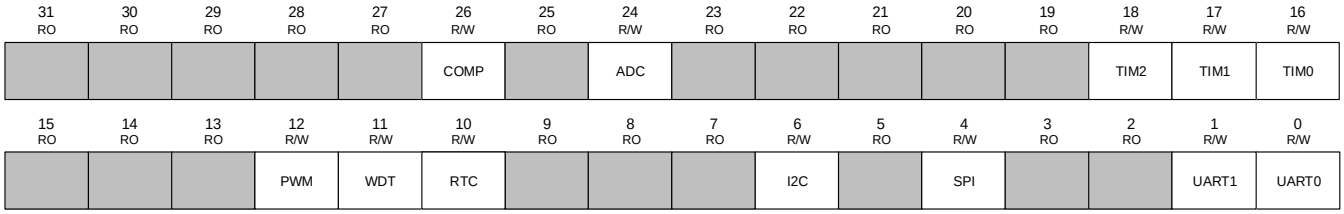


Offset: 0x0024

Bits	Name	Type	Reset	Description
31:4	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	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.

4.4.2.11 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

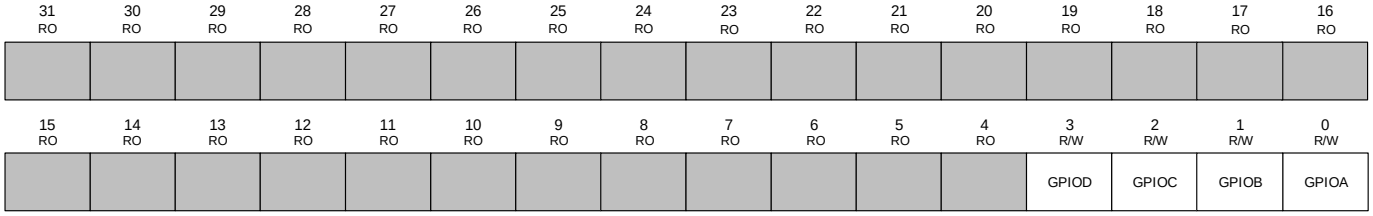


Offset: 0x0028

Bits	Name	Type	Reset	Description
31:27	*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	*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
23:19	*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	*TM2*	R/W	0	Timer 2 Reset Request 0: Manually clear after being set 1: Assert a reset signal to Timer 2
17	*TM1*	R/W	0	Timer 1 Reset Request 0: Manually clear after being set 1: Assert a reset signal to Timer 1
16	*TM0*	R/W	0	Timer 0 Reset Request 0: Manually clear after being set 1: Assert a reset signal to Timer 0
15:13	*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.
12	*PWM*	R/W	0	PWM Reset Request. 0: Manually clear after being set 1: Assert a reset signal to PWM
11	*WDT*	R/W	0	WDT Reset Request. 0: Manually clear after being set 1: Assert a reset signal to WDT
10	*RTC*	R/W	0	RTC Request 0: Manually clear after being set 1: Assert a reset signal to RTC
9:7	*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

Bits	Name	Type	Reset	Description
				or read.
6	*I2C*	R/W	0	I2C Reset Request. 0: Manually clear after being set 1: Assert a reset signal to I2C0
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.

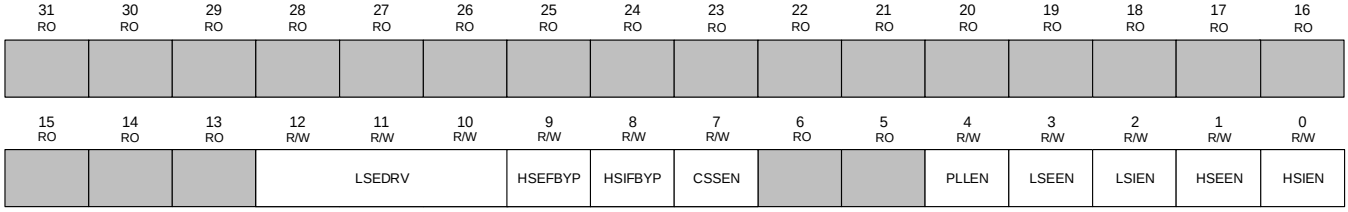
4.4.2.12 SC_RST_AHB - AHB PERIPHERAL RESET REQUEST REGISTER



Offset: 0x002C

Bits	Name	Type	Reset	Description
31:4	*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	*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

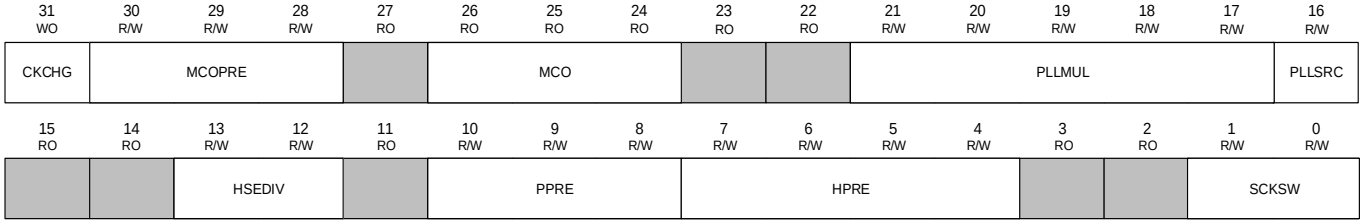
4.4.2.13 SC_CK_CTRL - CLOCK CONTROL REGISTER



Offset: 0x0030

Bits	Name	Type	Reset	Description
31:13	*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.
12:10	*LSEDRV*	R/W	0x1	LSE oscillator drive capability
9	*HSEFBYP*	R/W	1	HSE Clock Output Bypass 0: HSE 1: Bypass
8	*HSIFBYP*	R/W	1	HSI Clock Output Filter/Bypass 0: Filter 1: Bypass
7:5	*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	*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

4.4.2.14 SC_CK_CONF - CLOCK CONFIGURATION REGISTER



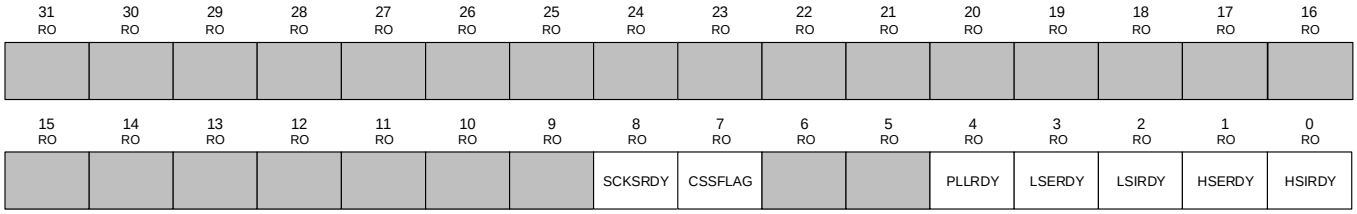
Offset: 0x0034

Bits	Name	Type	Reset	Description
31	*CKCHG*	WO	0	Change Clock Configuration
30:28	*MCOPRE*	R/W	0x0	Microcontroller Clock Output Prescaler It is highly recommended to change this prescaler only when the MCO output is disabled to avoid glitches. 0x0 : MCO is divided by 1 0x1 : MCO is divided by 2 0x2 : MCO is divided by 4 0x4 : MCO is divided by 8
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	*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 4-32 MHz (HSE) oscillator clock selected 0x5 : System clock selected 0x6 : AHB clock selected 0x7 : APB clock selected
23:22	*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.
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
15:14	*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
13:12	*HSEDIV*	R/W	0x0	PREDIV Division Factor These bits can be written only when the PLL is disabled. 0x0 : PREDIV input clock not divided 0x1 : PREDIV input clock divided by 2 0x2 : PREDIV input clock divided by 4 0x3 : PREDIV input clock divided by 8
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 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_CK_STAT - CLOCK SOURCE STATUS REGISTER

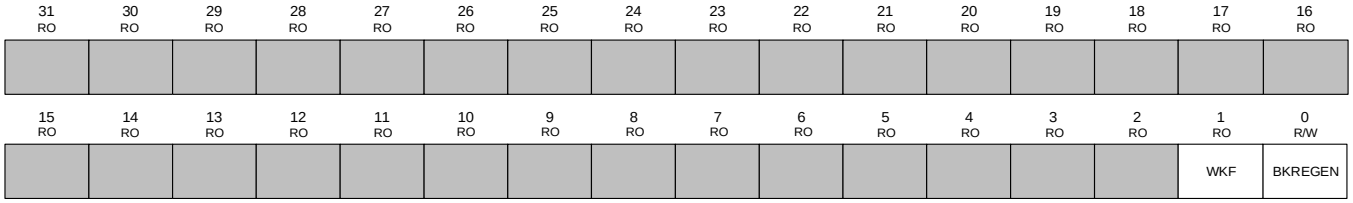
Bits in this register are set when the corresponding clock source are ready.



Offset: 0x003C

Bits	Name	Type	Reset	Description
31:9	*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.
8	*SCKSRDY*	RO	0	System Clock Switch Ready Flag 0: System Clock Switch is not yet ready 1: System Clock Switch is ready
7:5	*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	*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

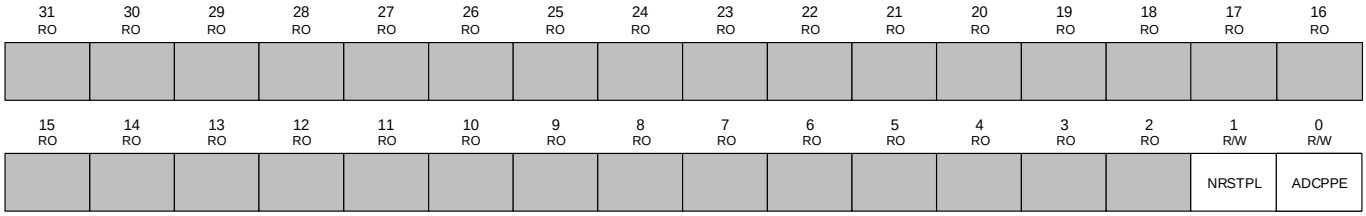
4.4.2.16 SC_BK_CTRL - BACKUP CONTROL REGISTER



Offset: 0x0040

Bits	Name	Type	Reset	Description
31:2	*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	*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

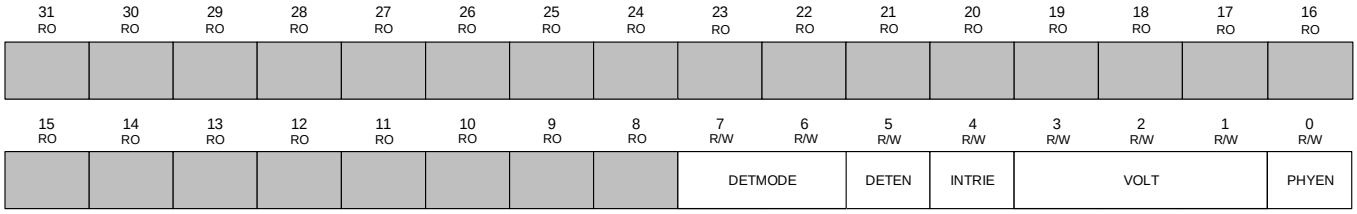
4.4.2.17 SC_PWR_CTRL - POWER CONTROL REGISTER



Offset: 0x0048

Bits	Name	Type	Reset	Description
31:2	*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	*NRSTPL*	R/W	0	NRST Pull Low, active high 0: No function 1: Pull Low
0	*reserved*	RO	0	ADC/Comparator LDO enbale 0: Disable LDO 1: Enable LDO

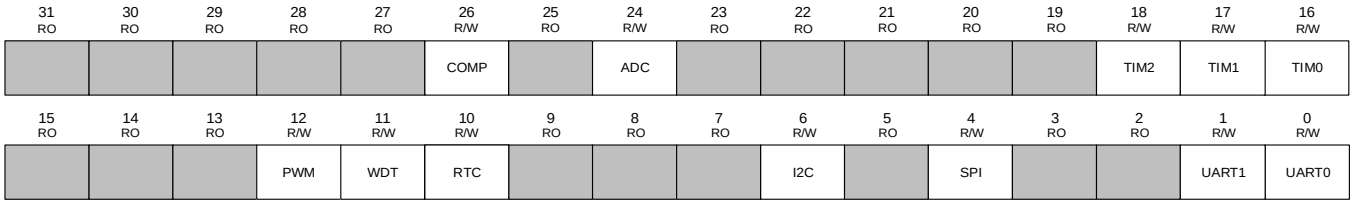
4.4.2.18 SC_PVD_DET - PVD DETECTION REGISTER



Offset: 0x004C

Bits	Name	Type	Reset	Description
31:8	*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.
7:6	*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

4.4.2.19 SC_SLP_APB - INTO SLEEP APB PERIPHERAL CLOCK GATING ENABLE REGISTER

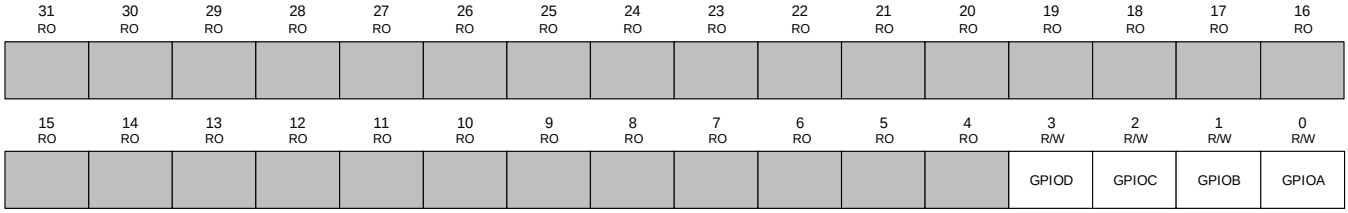


Offset: 0x0050

Bits	Name	Type	Reset	Description
31:27	*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	*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.
23:19	*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	*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.
15:13	*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.
12	*PWM*	R/W	0	Sleep Mode PWM Clock Gating Control This bit controls the clock gating for PWM module 0 in the sleep mode.
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	*RTC*	R/W	0	Sleep Mode RTC Gating Control This bit controls the clock gating for RTC module in the sleep mode.
9:7	*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	*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.

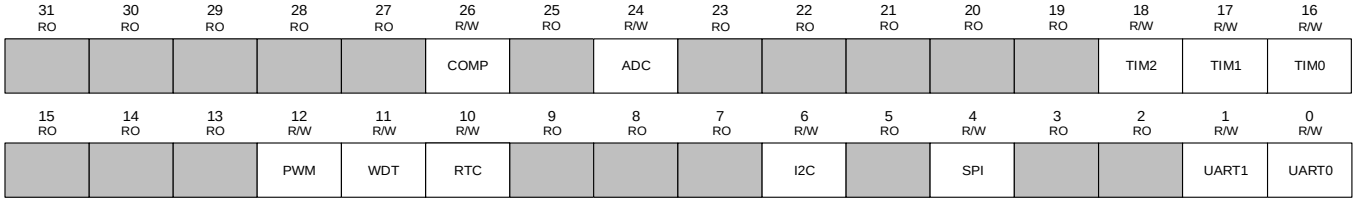
4.4.2.20 SC_SLP_AHB – INTO SLEEP AHB PERIPHERAL CLOCK GATING ENABLE REGISTER



Offset: 0x0054

Bits	Name	Type	Reset	Description
31:4	*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	*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.

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

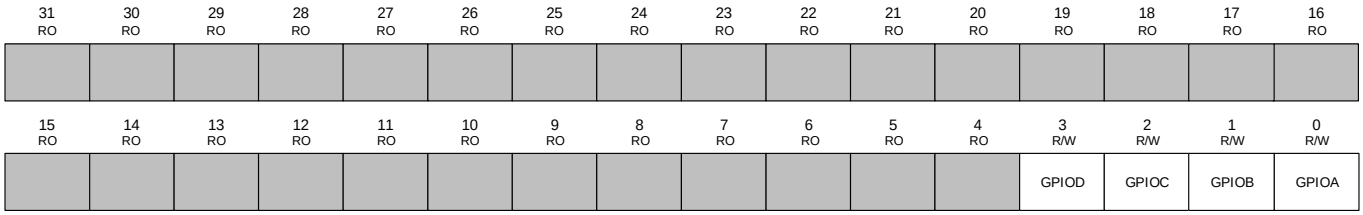


Offset: 0x0058

Bits	Name	Type	Reset	Description
31:27	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	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	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	Deep Sleep Mode ADC Clock Gating Control. This bit controls the clock gating for ADC module in the deep sleep mode.
23:19	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	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.
15:13	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.
12	PWM	R/W	0	Deep Sleep Mode PWM Clock Gating Control This bit controls the clock gating for PWM module 0 in the deep sleep mode.
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	RTC	R/W	0	Deep Sleep Mode RTC Gating Control This bit controls the clock gating for RTC module in the deep sleep mode.
9:7	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	I2C	R/W	0	Deep Sleep Mode I2C0 Clock Gating Control This bit controls the clock gating for I2C module 0 in the deep sleep

Bits	Name	Type	Reset	Description
				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	Deep Sleep Mode SPI Clock Gating Control This bit controls the clock gating for SPI module 0 in the deep 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	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.

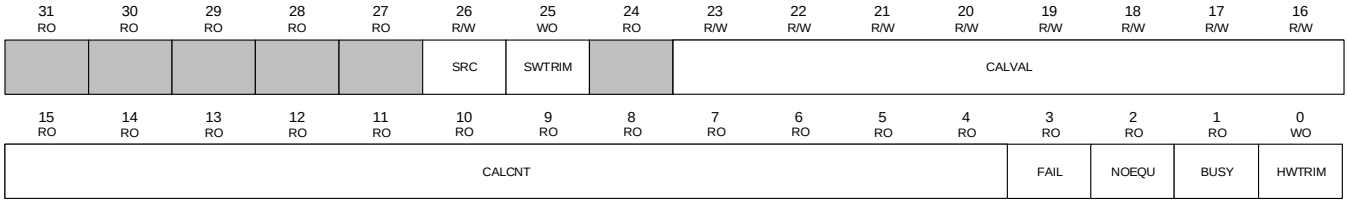
4.4.2.22 SC_DSLP_AHB - INTO DEEP SLEEP AHB PERIPHERAL CLOCK GATING ENABLE REGISTER



Offset: 0x005C

Bits	Name	Type	Reset	Description
31:4	*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	*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.

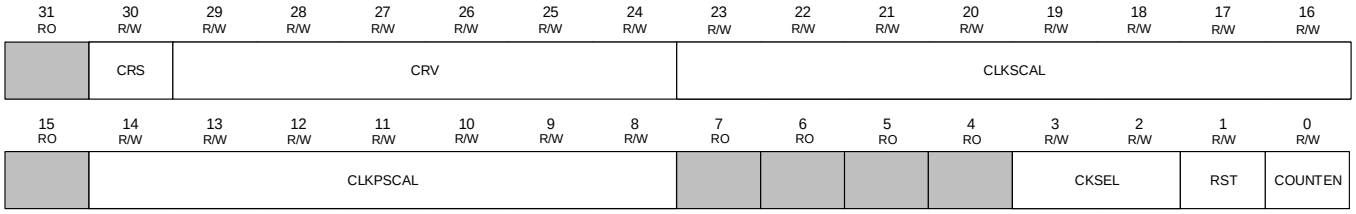
4.4.2.23 SC_OSC_TRIM - RING OSC HARDWARE TRIMMING REGISTER



Offset: 0x006C

Bits	Name	Type	Reset	Description
31:27	*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	*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*	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.
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.

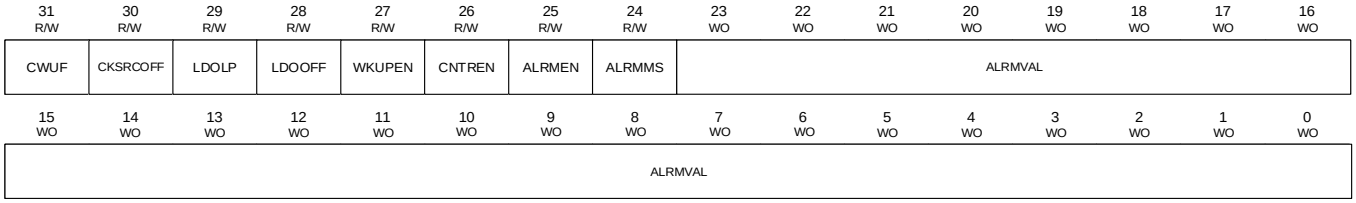
4.4.2.24 SC_BKRTC_CTRL - BACKUP RTC CONTROLLER REGISTER



Offset: 0x0080

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	*CRS*	R/W	0	Calibration Value Control 1: Decrease Calibration Value, CRV [5:0] 0: Increase Calibration Value, CRV[5:0]
29:24	*CRV*	R/W	0	RTC clock calibration value
23:16	*CLKSCAL*	R/W	0xFF	RTC clock scale, maximum divider is 256
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.
14:8	*CLKPSCAL*	R/W	0x7F	RTC clock prescale, maximum divider is 128
7:4	*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.
3:2	*CKSEL*	R/W	0x0	RTC clock source selection 00: No clock 01: LSE oscillator clock used as RTC clock 10: LSI oscillator clock used as RTC clock 11: HSE oscillator clock divided by 128 used as RTC clock
1	*RST*	R/W	0	RTC Counter Reset Request 0: No reset signal to be sent 1: Send a reset signal
0	*COUNTEN*	R/W	0	RTC Counting Enable 0: Disable RTC counting 1: Enable RTC counting

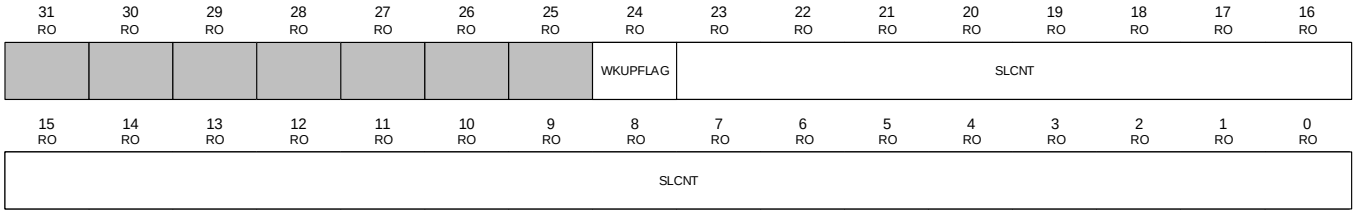
4.4.2.25 SC_BKSLP_CTRL - BACKUP RTC SLEEP CONTROLLER REGISTER



Offset: 0x0084

Bits	Name	Type	Reset	Description
31	*CWUF*	R/W	0	Clear Wakeup Flag Read: This bit is set by hardware to indicate that the device is waken up from Low Power mode. Write: Set this bit to clear the wakeup flag and *WKUPFLAG* bit in SC_BK_STATUS.
30	*CKSRCOFF*	R/W	0	In Deep Sleep mode, turn off all clock sources
29	*LDOLP*	R/W	0	In Deep Sleep, turn on LDO 1.8V into the low power
28	*LDOOFF*	R/W	0	In Deep Sleep, turn off LDO 1.8V power
27	*WKUPEN*	R/W	0	Enable the external pin to wake up system 1: Enable the system to be waken up by external pin
26	*CNTREN*	R/W	0	Backup RTC sleep counter enable 1: Enable RTC sleep counter
25	*ALRMEN*	R/W	0	Backup RTC sleep alarm enable 1: Enable RTC sleep alarm
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_STATUS - BACKUP STATUS REGISTER

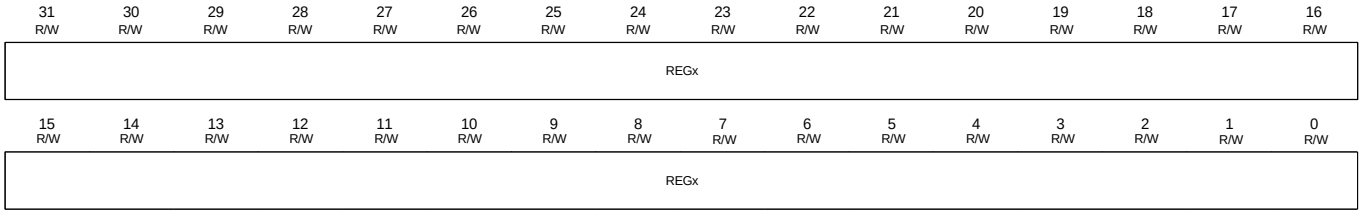


Offset: 0x0088

Bits	Name	Type	Reset	Description
31:25	*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	*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

4.4.2.27 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

4.5 FLASH CONTROLLER (FC)

The PT32U301 embedded flash has 4kB Information Block and 32kB 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 - 0x08007FFF / 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 example, if the current operating frequency is 48 MHz (20.8 ns) then 1 wait cycles are added in order to meet the read access time. To do this write 1 to “wait cycle” field of **SC_NVMACR** register.

4.5.3 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:

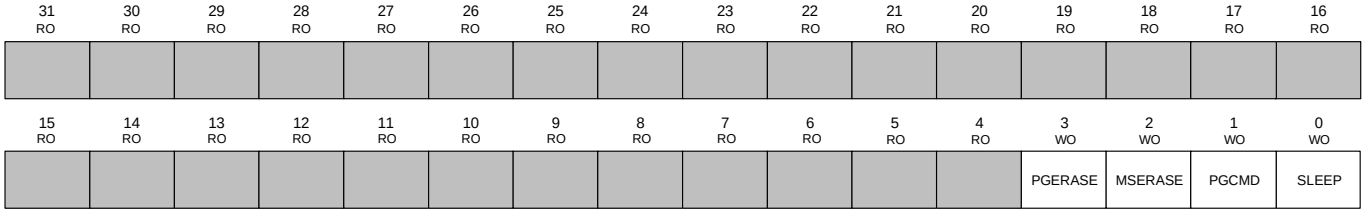
1. Write the MASERASE bit to the FC_CMD register.
2. Poll the FC_CMD register until the MASERASE bit is automatically cleared

4.5.4 FLASH CONTROLLER REGISTER MAP

Base Address: 0x5001_0000

Offset	Symbol	Type	Reset Value	Description	See page
0x0004	FC_CMD	WO	0x0000_0000	Flash Command Register	72
0x0008	FC_PDATA	R/W	0x0000_0000	Flash Program Data Register	73
0x0010	FC_PADDR	R/W	0x0000_0000	Flash Program Address Register	73

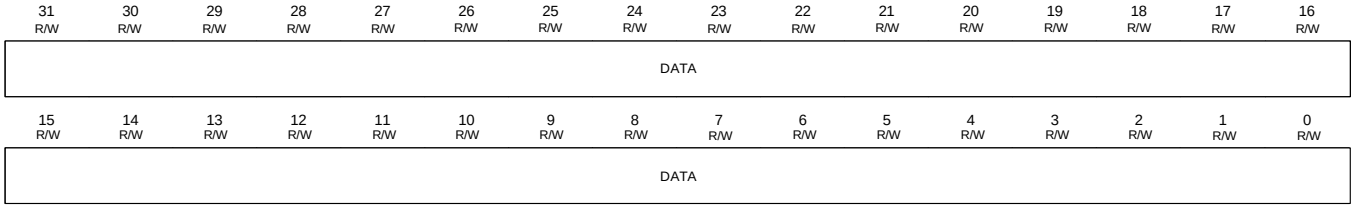
4.5.4.1 FC_CMD - EMBEDDED COMMAND REGISTER



Offset: 0x0004

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	*PGERASE*	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	*MSERASE*	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.

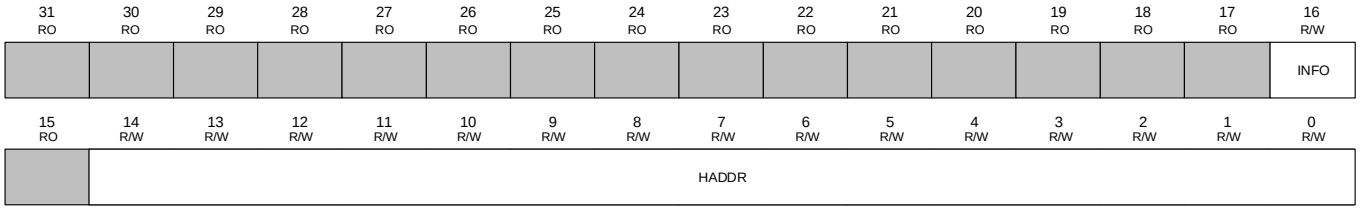
4.5.4.2 FC_PDATA - EMBEDDED PROGRAM DATA REGISTER



Offset: 0x0008

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

4.5.4.3 FC_PADDR - EMBEDDED PROGRAM ADDRESS REGISTER



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	*INFO*	R/W	0	Eflash Information Block control 1: Enable Eflash Information Block Control to be written or erased.
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.
14:0	*HADDR*	R/W	0x0	Embedded Program/Erase Address Register. (Page Size=512 Bytes) Mass Erase: HADDR[14:0] don't care Page Erase: HADDR[8:0] don't care; HADDR[14:9] is page address.(64 page/32K Bytes) Info. Erase: HADDR [8:0] & HADDR [14:12] don't care. HADDR [11:9] is info. page address.(8 page/4K Bytes) Program: HADDR [1:0] don't care. HADDR[14:2] is double word address

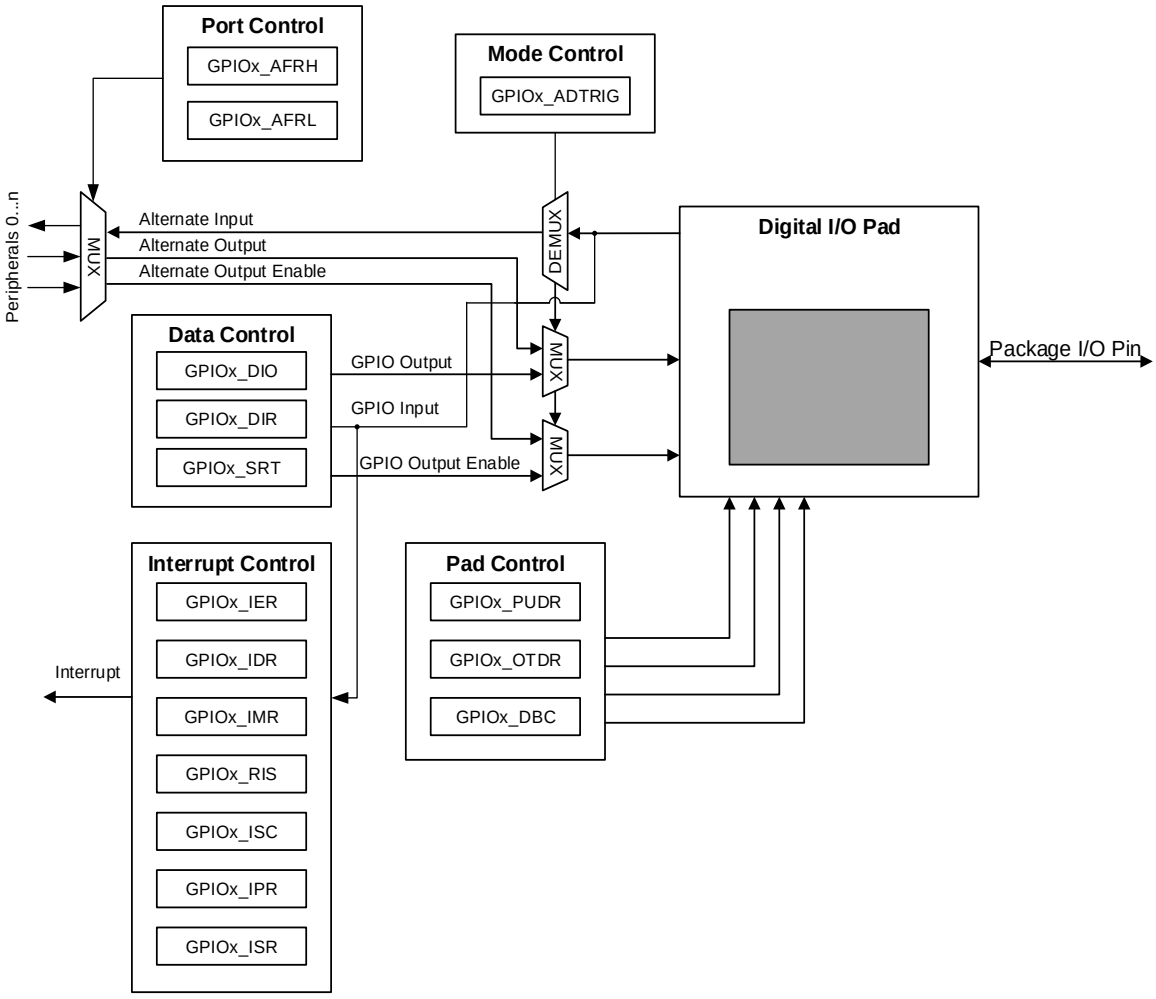
4.6 GENERAL PURPOSE I/O (GPIO)

In PT32U301, the GPIO module has up to 45 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 45 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

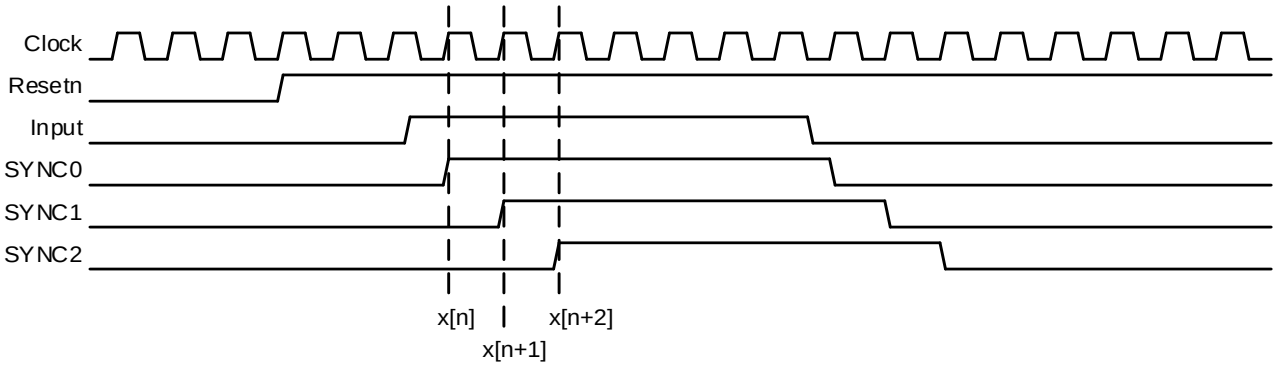
The pad control registers in GPIO include the GPIO (A/B/C/D) Port Pull-Up/Pull-Down Register (**GPIOx_PUDR**) and GPIO (A/B/C/D) Port Output Type and Drive Register **(GPIOx_OTDR)**. These register control the pull-up and pull-down resistor, output type and drive strength.

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 $x\ [n+1]\ \times T_s$.

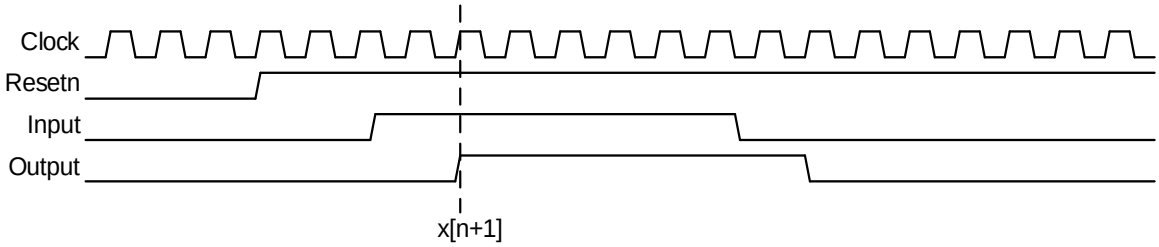
If open the debounce module, SYNC2 signal will go into the debounce circuit, then it will be sampled and counted times (sample frequency and count times value are setting by user). The time can be express as $[(SPT+1)\times(FILTCNT+1)]\times T_s$. The SPT is sample frequency value, FILTCNT is the count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

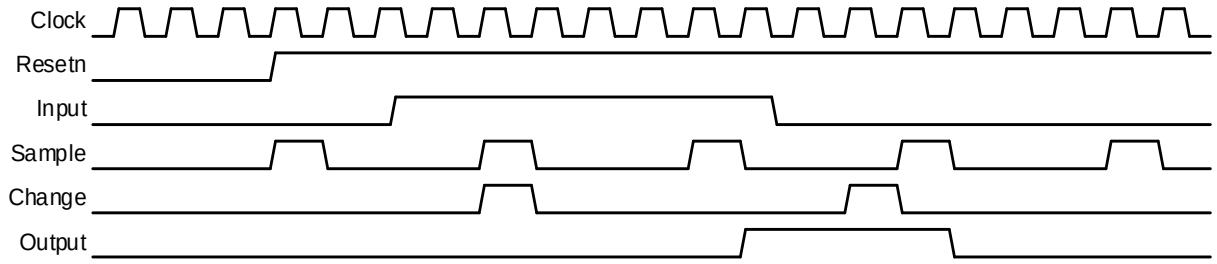


In the above diagram, the debounce function was be disabled, so the output will happen **$x\ [n+1]\ \times T_s$**.

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



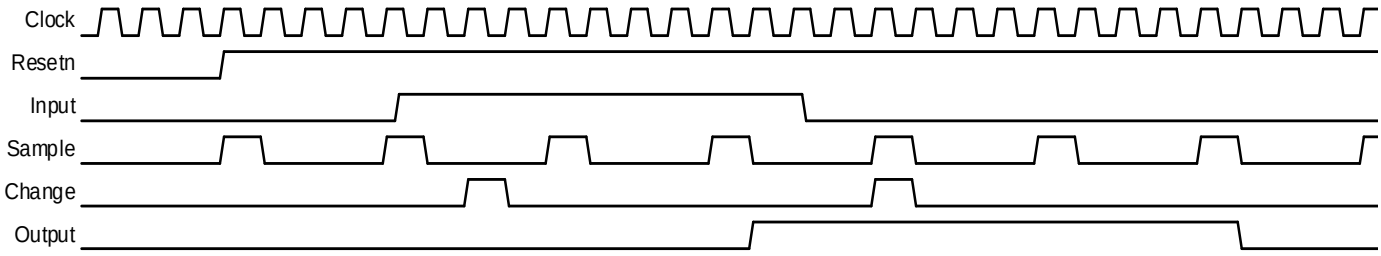
In the above diagram, debounce module is enabled, “Sample” is the sample frequency, “Change” is represented for SYNC1 and SYNC2 are not the same. When “Change” is high, it means count times value will be cleared to 0, so the “Output” will be set after “Change” was happened and SYNC2 signal is sampling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



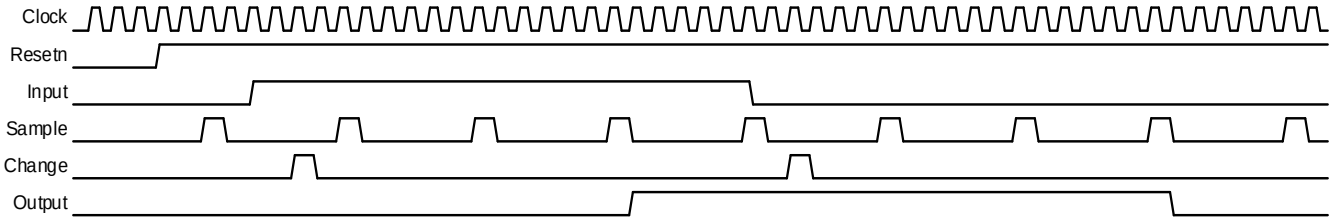
In the above diagram, debounce module is enabled, “**Sample**” is the sample frequency, “**Change**” is represented for **SYNC1** and **SYNC2** are not the same. When “**Change**” is high, it means count times value will be cleared to 0, so the “**Output**” will be set after “**Change**” was happened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

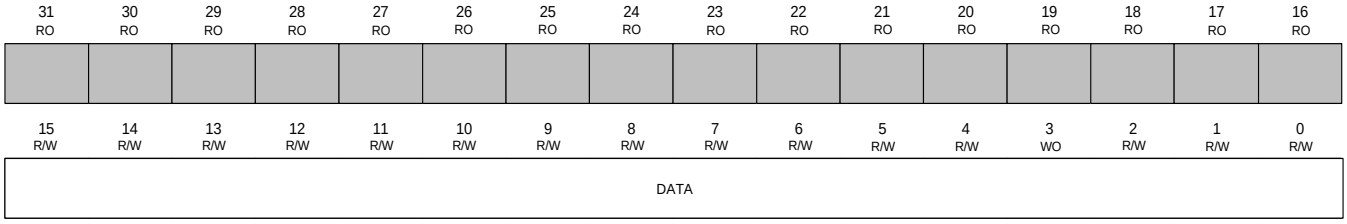
4.6.3 GENERAL-PURPOSE INPUT/OUTPUT REGISTER MAP

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	80
0x0004	GPIOx_DIR	R/W	0x0000_FFFF	GPIOx Data Direction Register	80
0x0008	GPIOx_SRT	WO	0x0000_0000	GPIOx Set/Rest Register	81
0x000C	GPIOx_DBC	R/W	0x0000_0000	GPIOx Debounce Count Register	81
0x0010	GPIOx_IPR	R/W	0x0000_0000	GPIOx Interrupt Polarity Register	82
0x0014	GPIOx_ISR	R/W	0x0000_0000	GPIOx Interrupt Sense Register	82
0x0018	GPIOx_IER	WO	0x0000_0000	GPIOx Interrupt Enable Register	83
0x001C	GPIOx_IDR	WO	0x0000_0000	GPIOx Interrupt Disable Register	83
0x0020	GPIOx_IMS	RO	0x0000_0000	GPIOx Interrupt Mask Status Register	84
0x0024	GPIOx_RIS	RO	0x0000_0000	GPIOx Raw Interrupt Status Register	84
0x0028	GPIOx_ISC	RW/1C	0x0000_0000	GPIOx Interrupt Masked Status & Interrupt Clear Register	85
0x0030	GPIOx_PUDR	R/W	0x0000_0000	GPIOx Port Pull-Up/Pull-Down Register	85
0x0034	GPIOx_OTSR	R/W	0x0000_0000	GPIOx Port Output Type and Drive Register	86
0x0038	GPIOx_AFRL	R/W	0x0000_0000	GPIOx Alternate Function Low Register	86
0x003C	GPIOx_AFRH	R/W	0x0000_0000	GPIOx Alternate Function High Register	87
0x0040	GPIOx_ADTRIG	R/W	0x0000_0000	GPIOx Trigger ADC Register	87

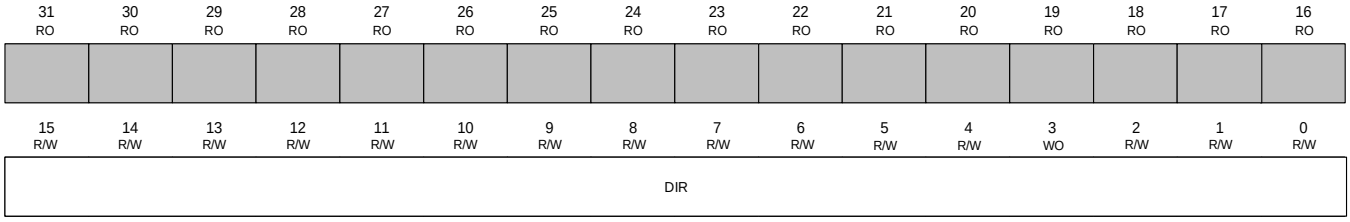
4.6.3.1 GPIOX_DIO - GPIOX INPUT AND OUTPUT REGISTER



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	*DATA*	R/W	Read: 0x03FF Write: 0xFFFF	GPIO Data GPIO data output when the GPIO Port's Pins are set to output : GPIO data input when the GPIO Port's Pins are set to input

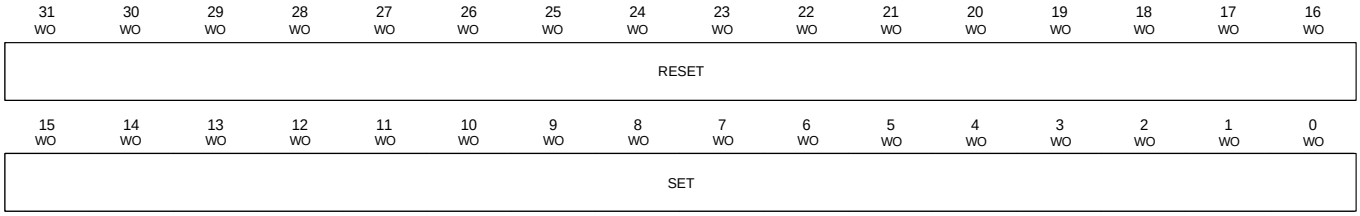
4.6.3.2 GPIOX_DIR - GPIOX DATA DIRECTION REGISTER



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



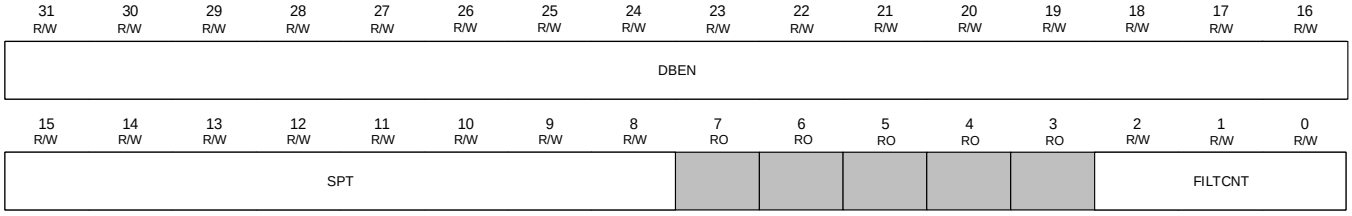
Offset: 0x0008

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.

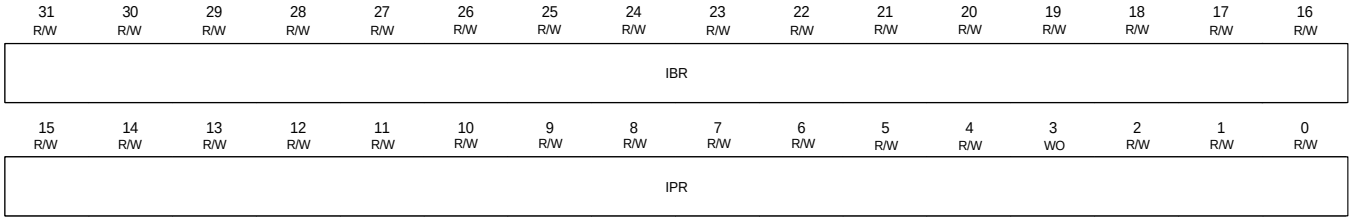
Derivation of the formula (system frequency / *SPT* / *FILTCNT*) = maximum data transfer effective frequency. For example: If system clock is 48 MHz, the debounce output of frequency is below 4 MHz (48/6/2 = 4 MHz); If the *SPT* and *FILTCNT* value are maximum, the valid frequency of input data is 23.437 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*	RO	0x0	Software should not 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

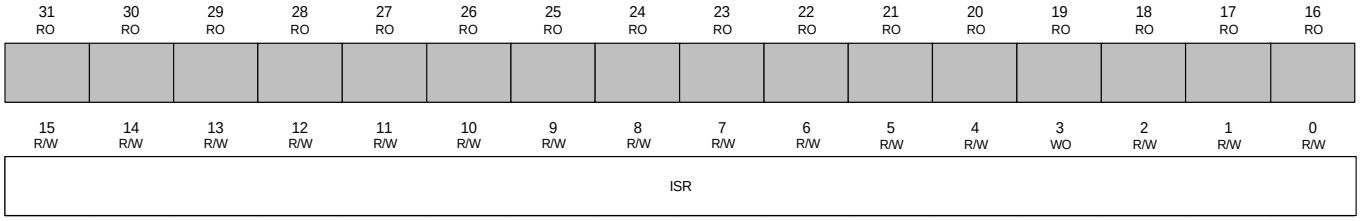
4.6.3.5 GPIOX_IPR - GPIOX INTERRUPT POLARITY REGISTER



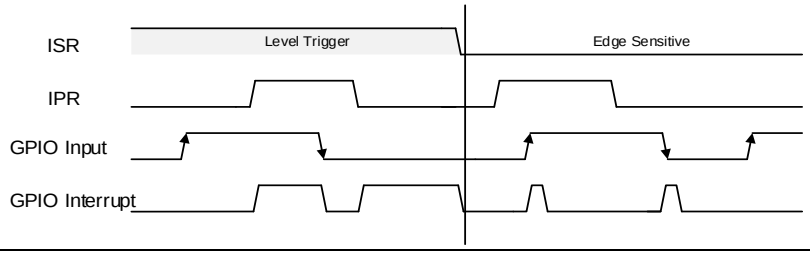
Offset: 0x0010

Bits	Name	Type	Reset	Description
31:16	*IBR*	R/W	0x0	0: According to the IPR bits to generate interrupt trigger. 1: Detect both edge to generate interrupt trigger.
15:0	*IPR*	R/W	0x0	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

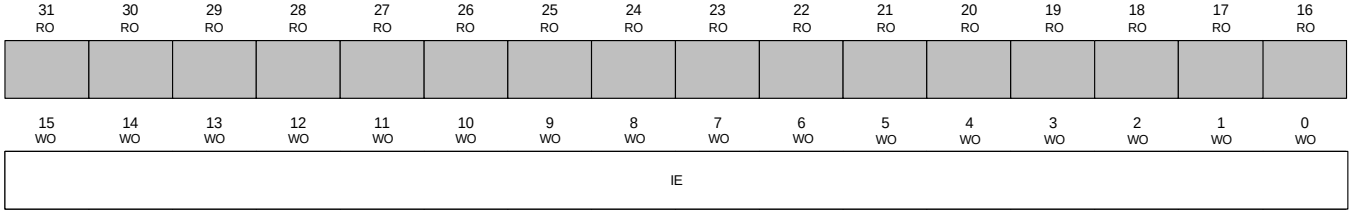
4.6.3.6 GPIOX_ISR - GPIOX INTERRUPT SENSE TYPE REGISTER



Offset: 0x0014

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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 

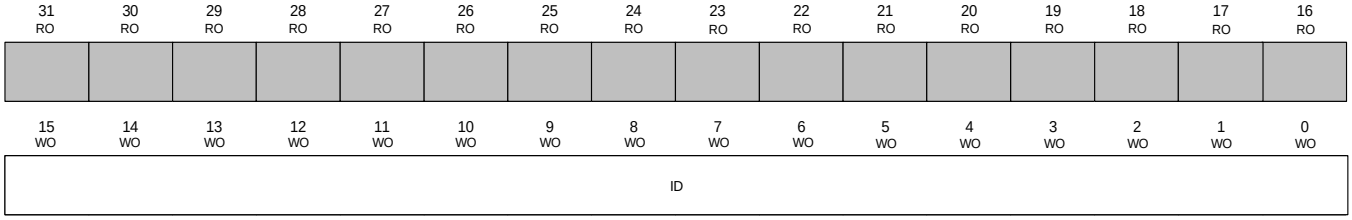
4.6.3.7 GPIOX_IER - GPIOX INTERRUPT ENABLE REGISTER



Offset: 0x0018

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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

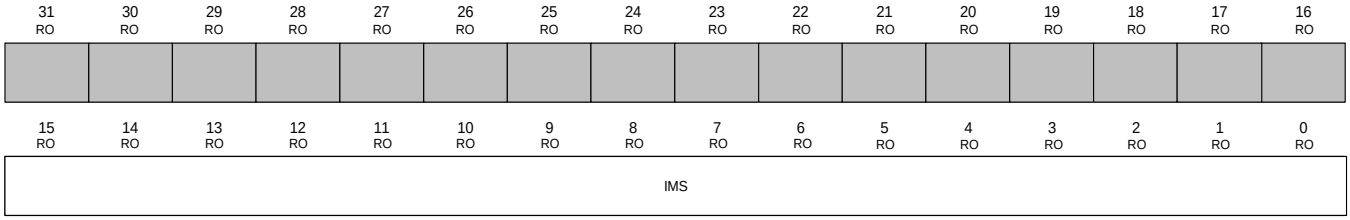
4.6.3.8 GPIOX_IDR - GPIOX INTERRUPT DISABLE REGISTER



Offset: 0x001C

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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

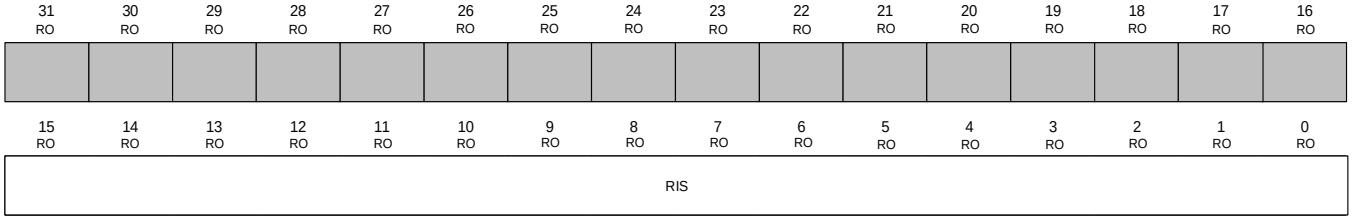
4.6.3.9 GPIOX_IMR - GPIOX INTERRUPT MASK STATUS REGISTER



Offset: 0x0020

Bits	Name	Type	Reset	Description
31:16	reserved	RO	0x0	Software should not 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	IMS	RO	0x0	GPIOx Interrupt Mask Status 0 : Corresponding pin interrupt is masked 1 : Corresponding pin interrupt is not masked

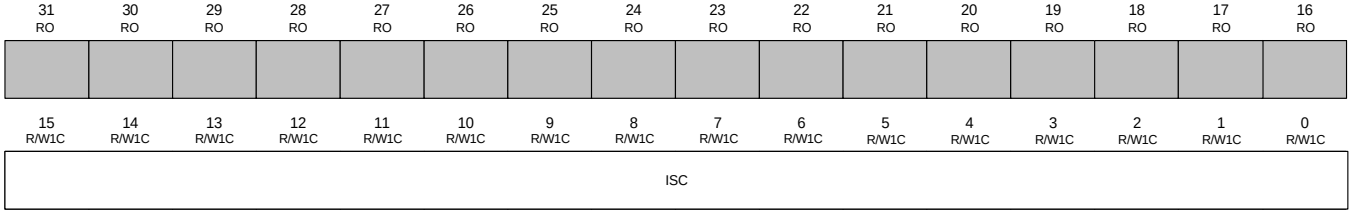
4.6.3.10 GPIOX_RIS - GPIOX RAW INTERRUPT STATUS REGISTER



Offset: 0x0024

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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

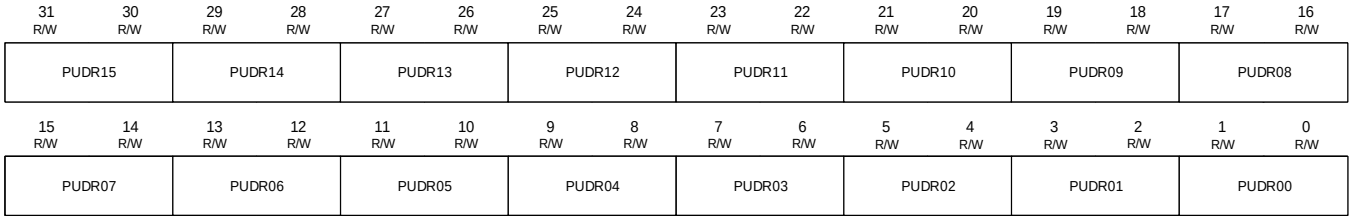
4.6.3.11 GPIOX_ISC - GPIOX INTERRUPT STATUS & INTERRUPT CLEAR REGISTER



Offset: 0x0028

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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

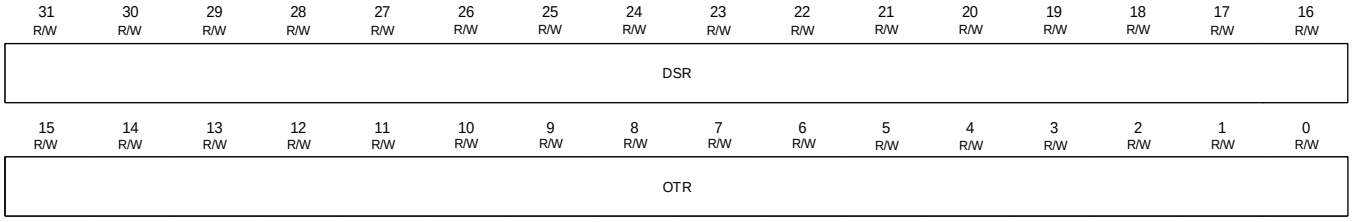
4.6.3.12 GPIOX_PUDR - GPIOX PORT PULL-UP/PULL-DOWN REGISTER



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

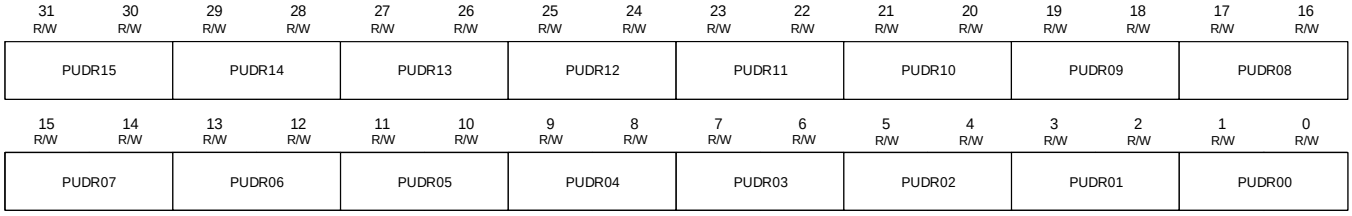


Offset: 0x0034

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

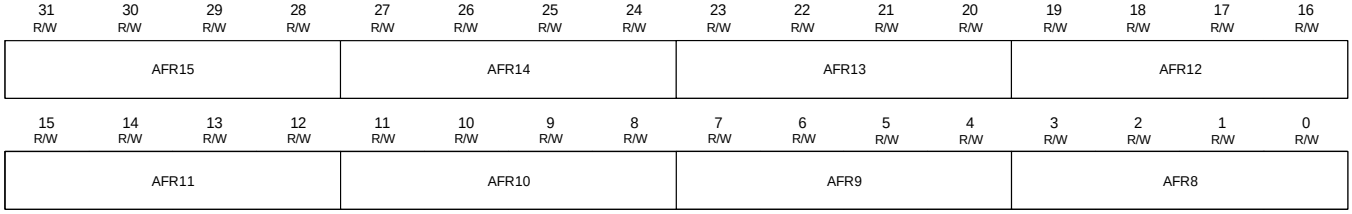
GPIO provides multiple functions of IO options, select other functions through **GPIOx_AFRL** and **GPIOx_AFRH**, classified functions. For example, if the pin PA00 whose default function is GPIO (PA00), and to configure its functions to AD0_CAIP0 (ADCIN0 & Comparator 0 Positive input), you must setting the register **GPIOA_AFR0** is 8. For further description about the alternate functions in PT32U301, 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

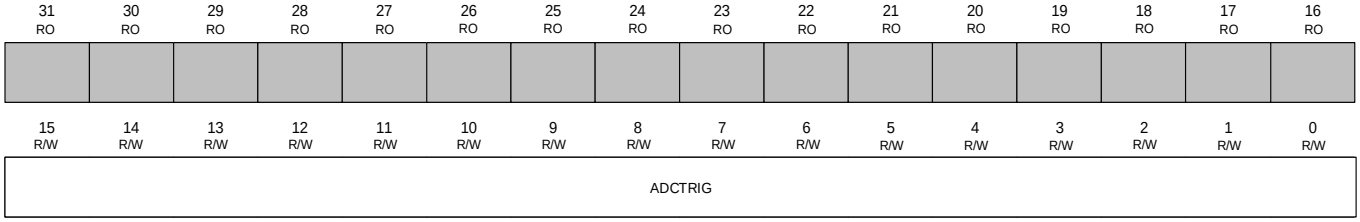


Offset: 0x003C

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*	RO	0x0	Software should not 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.

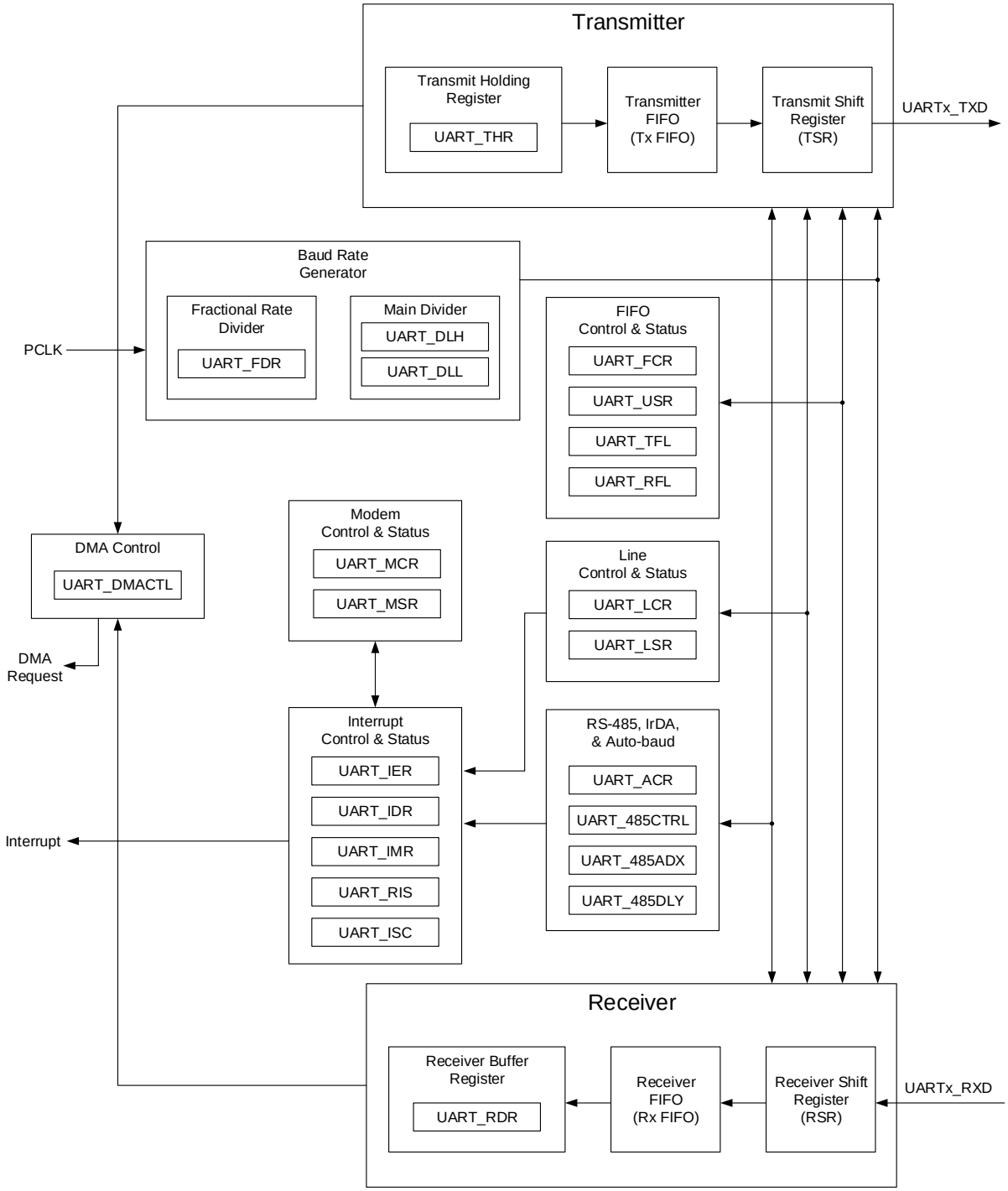
- Full Duplex, Asynchronous Communication.
- 16-byte Receive and Transmit FIFOs.
- Compatible with 16C550 standard.
- Programmable receiver buffer trigger level.
- Programmable baud-rate generator for each channel individually.
- Supports auto baud rate detect function.
- Eight interrupt sources
- Built-in 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.

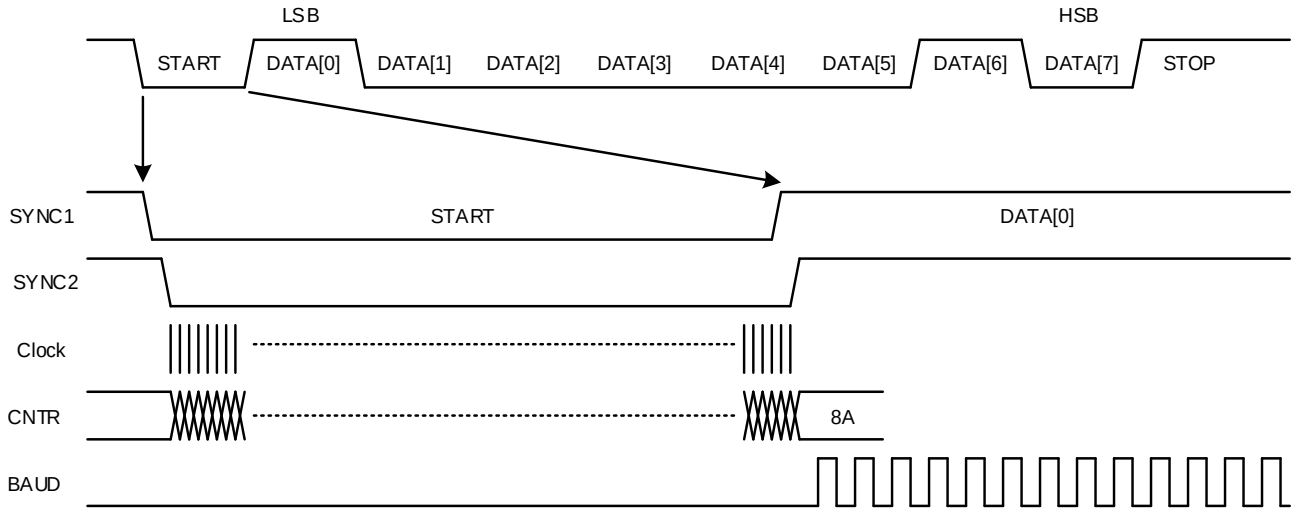
The auto-baud function can generate two interrupts.

- 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.

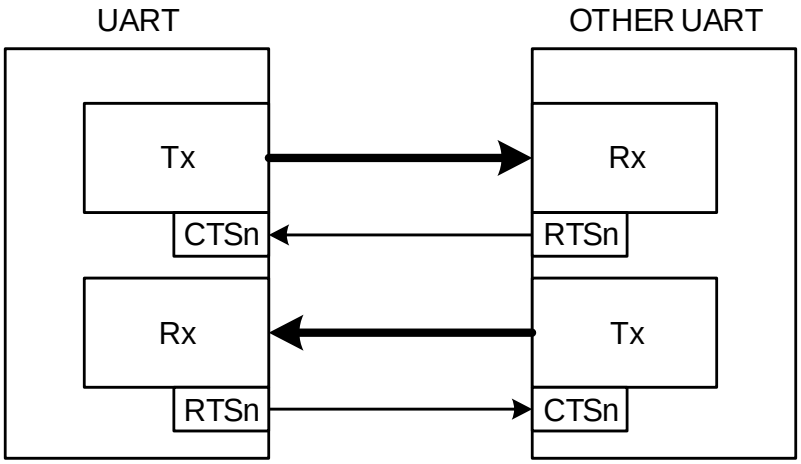
Figure 4.7-2: Only START bit to use for Auto-Baud-Rate



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



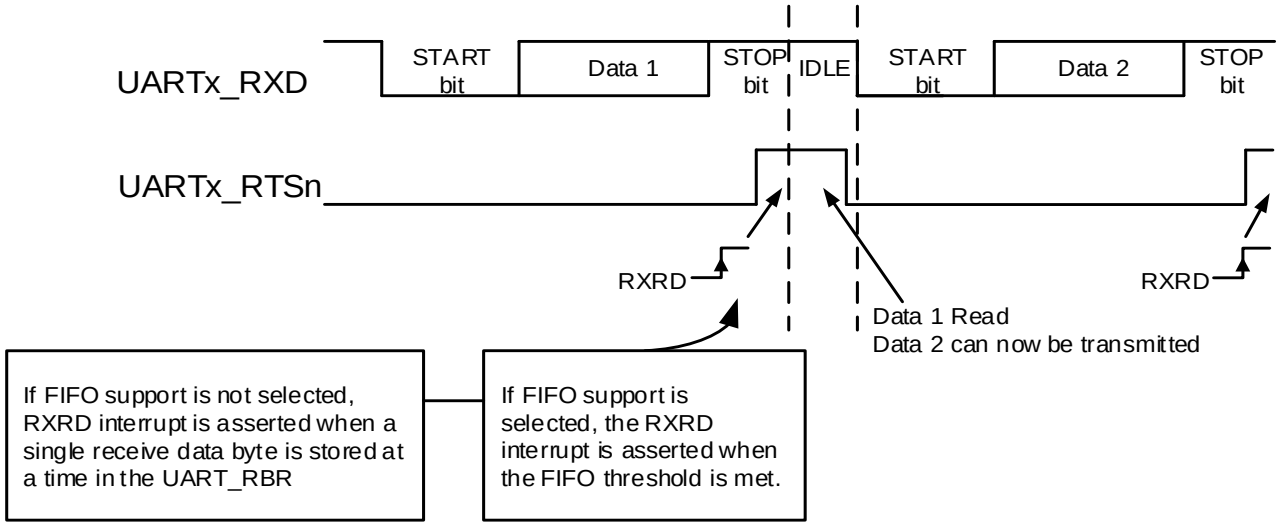
AUTO-RTS

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

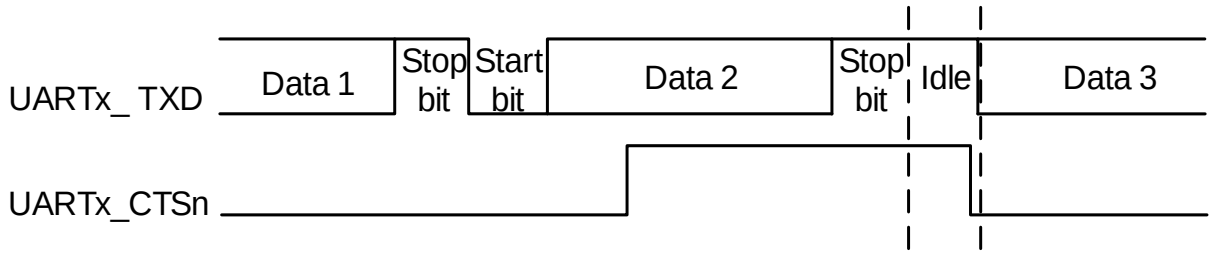


AUTO-CTS

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, $^1/_4$, $^1/_2$or "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

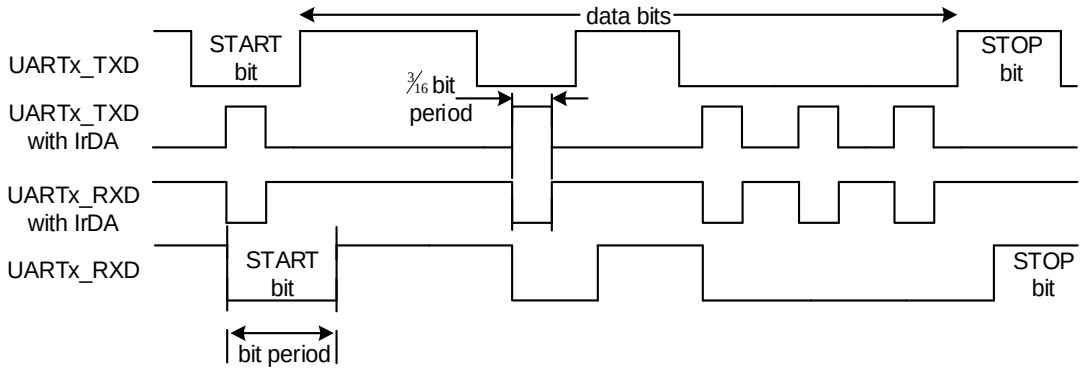
IrDA SIR mode supports bi-directional data communications with remote devices using infrared radiation as the transmission medium. Its data format is similar to the standard serial data format. Each data character is sent serially, beginning with a start bit, followed by 8 data bits, and ending with 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

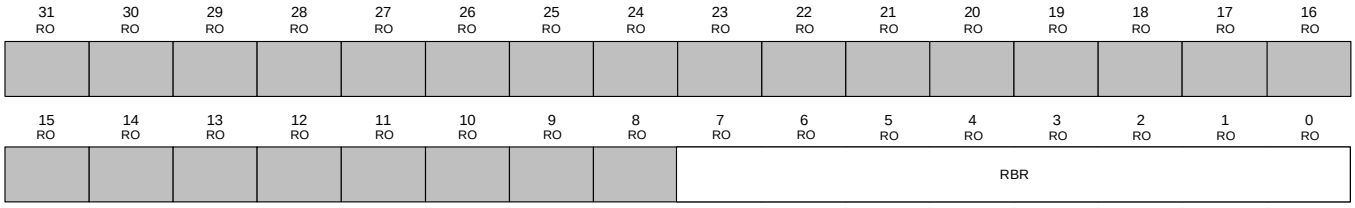
UART base address:

- UART 0: 0x4800_0000
- UART 1: 0x4800_0100

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	UART_RBR	RO	0x0000_0000	UART Receive Buffer Register	97
0x0000	UART_THR	WO	0x0000_0000	UART Transmit Holding Register	97
0x0000	UART_DLL	R/W	0x0000_0000	UART Divisor Latch (Low)	98
0x0004	UART_DLH	R/W	0x0000_0000	UART Divisor Latch (High)	98
0x0004	UART_FCR	R/W	0x0000_0000	UART FIFO Control Register	99
0x0008	UART_ACR	R/W	0x0000_0000	UART Auto Baud Rate Control Register	100
0x000C	UART_FDR	R/W	0x0000_0000	UART Fractional divider register	100
0x0010	UART_LCR	R/W	0x0000_0000	UART Line Control Register	101
0x0014	UART_MCR	R/W	0x0000_0000	UART Modem Control Register	102
0x0018	UART_LSR	RO	0x0000_0060	UART Line Status Register	103
0x001C	UART_MSR	RO	0x0000_0000	UART Modem Status Register	105
0x0020	UART_IER	WO	0x0000_0000	UART Interrupt Enable Register	106
0x0024	UART_IDR	WO	0x0000_0000	UART Interrupt Disable Register	107
0x0028	UART_IMR	RO	0x0000_0000	UART Interrupt Mask Register	108
0x002C	UART_RIS	RO	0x0000_0000	UART Raw Interrupt Status Register	109
0x0030	UART_ISC	R/W1C	0x0000_0000	UART Interrupt Status and Clear Register	110
0x0034	UART_485CTRL	R/W	0x0000_0000	UART RS485 Control Register	111
0x0038	UART_485ADX	R/W	0x0000_0000	UART RS485 Address Match Register	112
0x003C	UART_485DLY	R/W	0x0000_0000	UART RS485 Direction Delay Register	112
0x007C	UART_USR	RO	0x0000_0006	UART Status Register	113
0x0080	UART_TFL	RO	0x0000_0000	UART Transmit FIFO Level Register	114
0x0084	UART_RFL	RO	0x0000_0000	UART Receive FIFO Level Register	114

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.

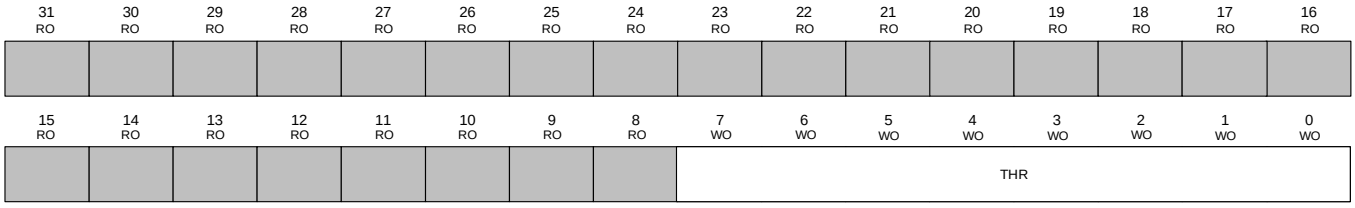


Offset: 0x0000

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.

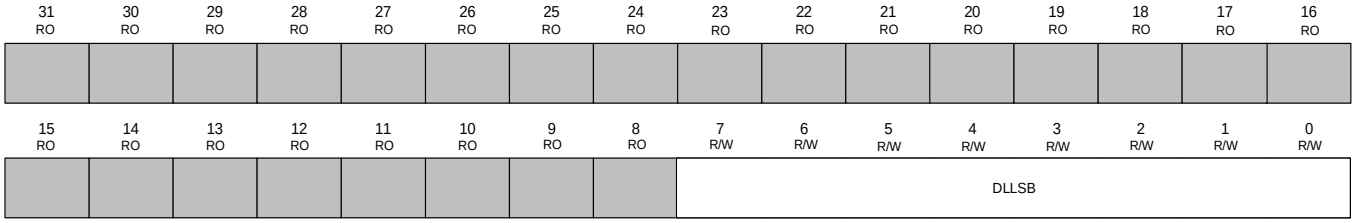


Offset: 0x0000

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.

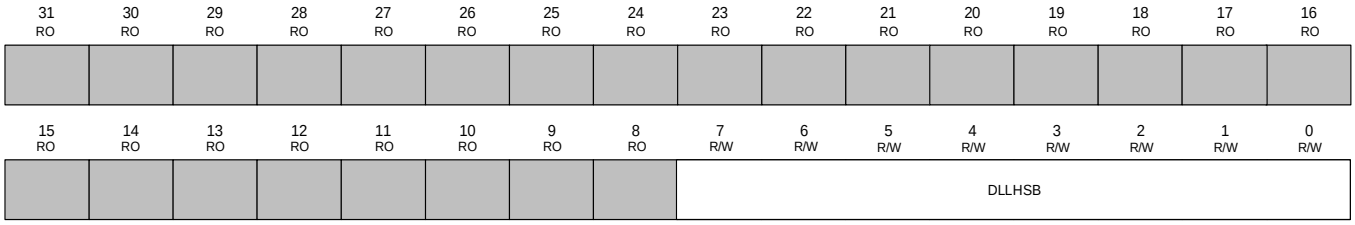


Offset: 0x0000

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

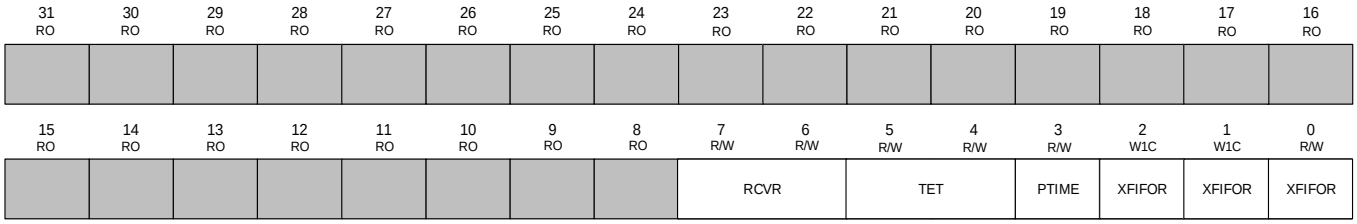


Offset: 0x0004

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.

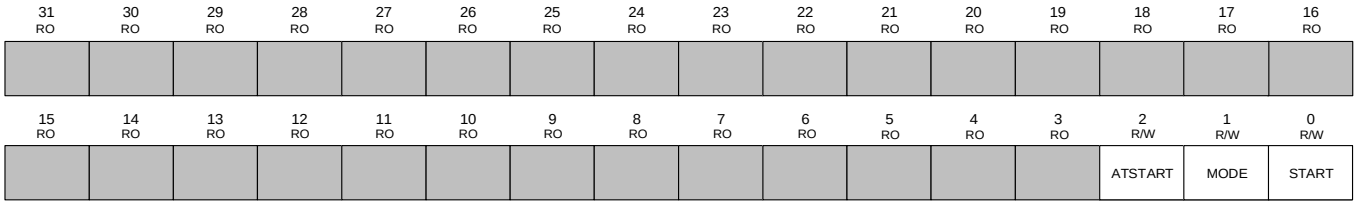


Offset: 0x0004

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.

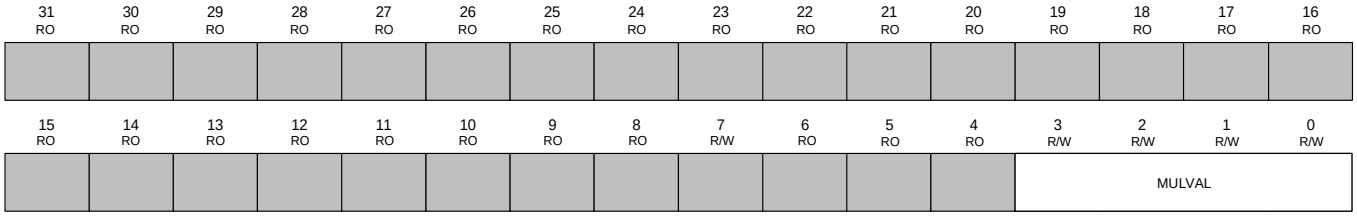


Offset: 0x0008

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.

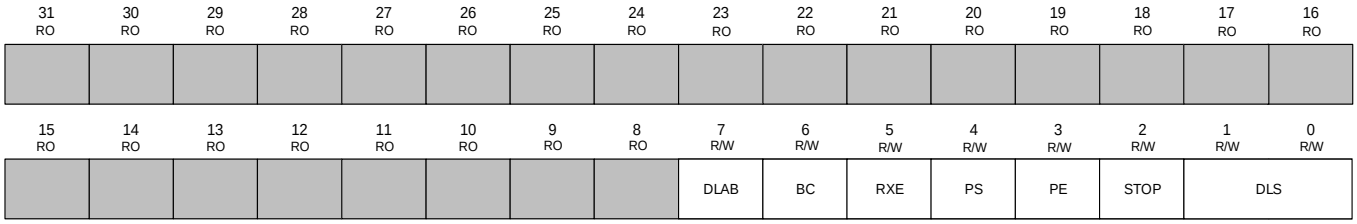


Offset: 0x000C

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.

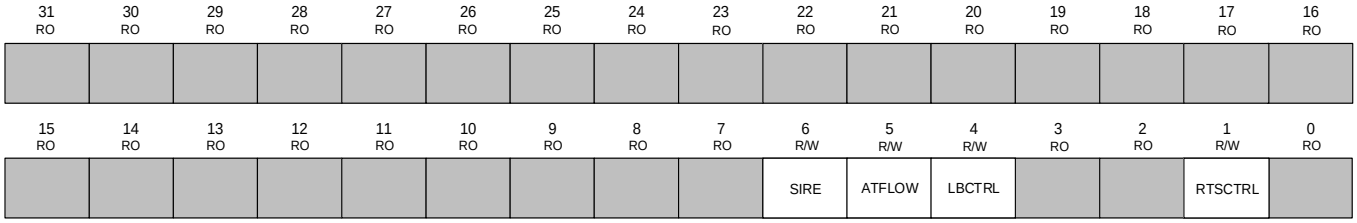


Offset: 0x0010

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.

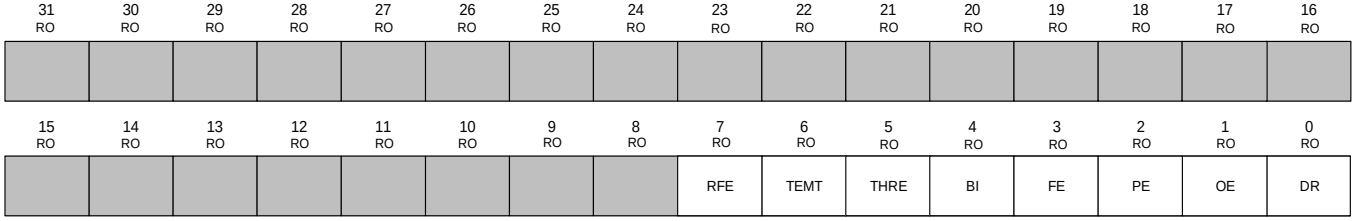


Offset: 0x0014

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.



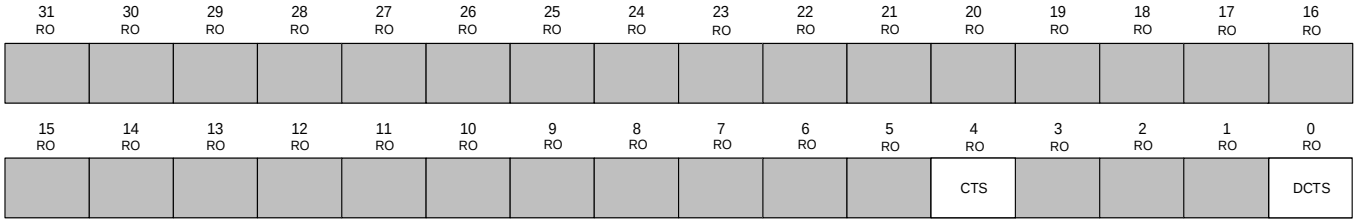
Offset: 0x0018

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

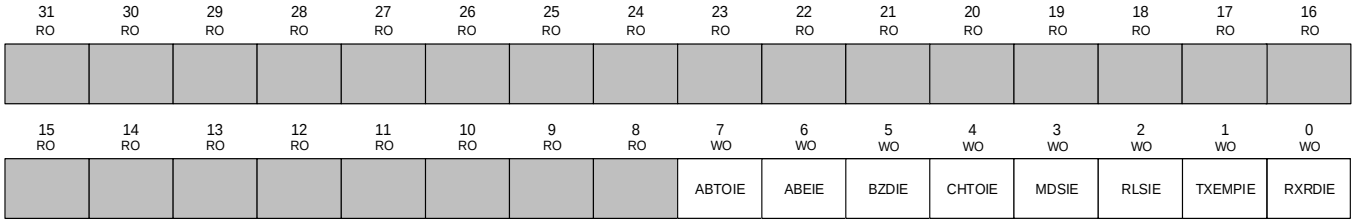
This is a read-only register that provides Modem input signals status information. Read this register clears the **UART_MSR**.*DCTS*. Note that modem signal does not have a direct impact on the UART operation, it only helps software to achieve Modem signal operation.



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

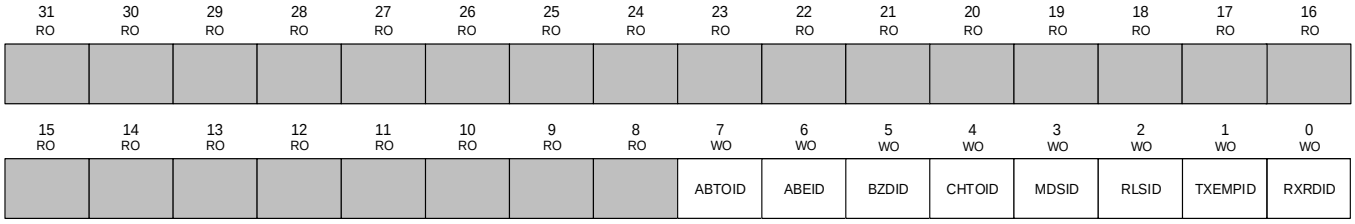
4.7.3.12 UART_IER - UART INTERRUPT ENABLE REGISTER



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.

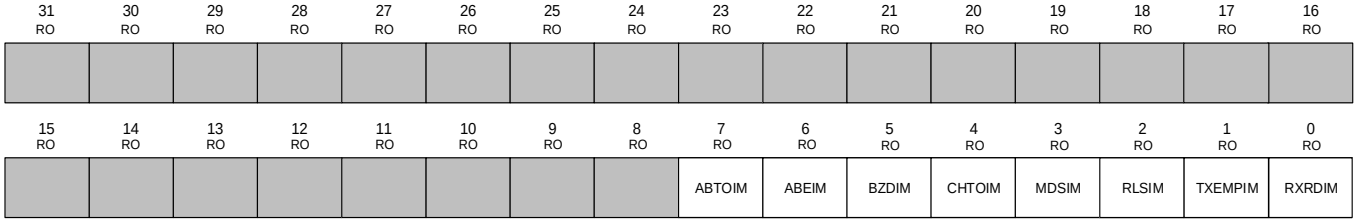
4.7.3.13 UART_IDR - UART INTERRUPT DISABLE REGISTER



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.

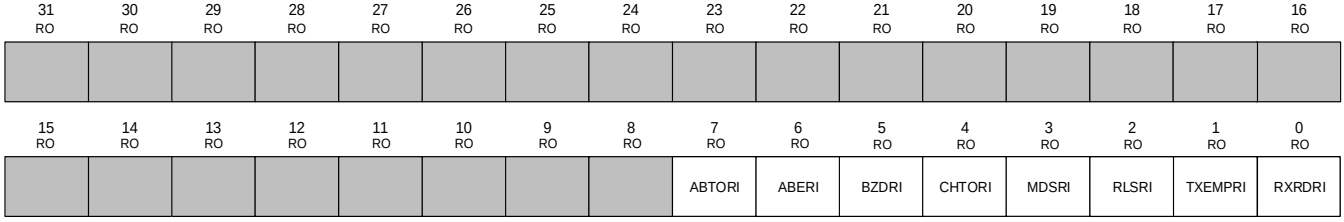
4.7.3.14 UART_IMR - UART INTERRUPT MASK STATUS REGISTER



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

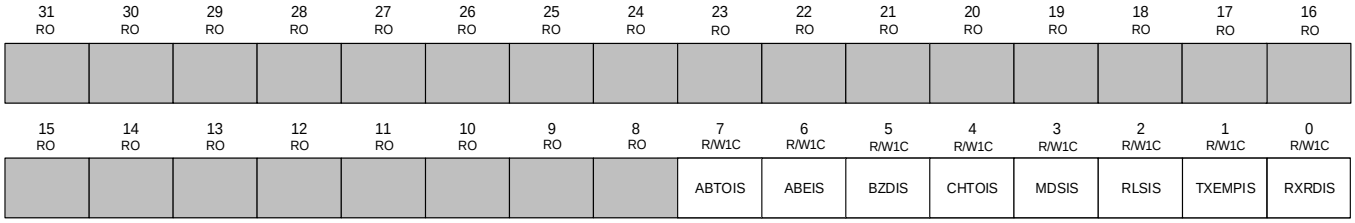


Offset: 0x002C

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

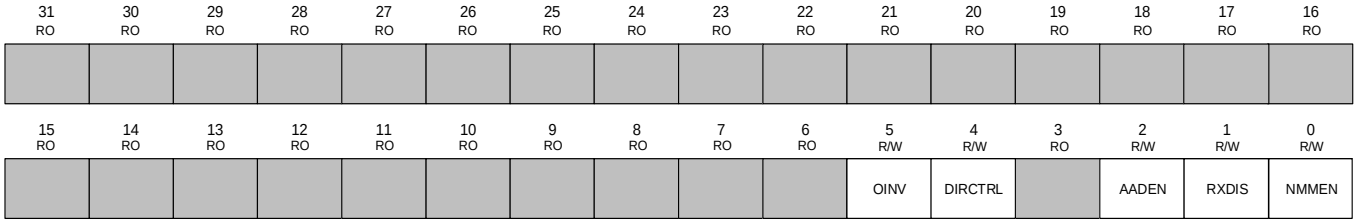
Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt status.



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.

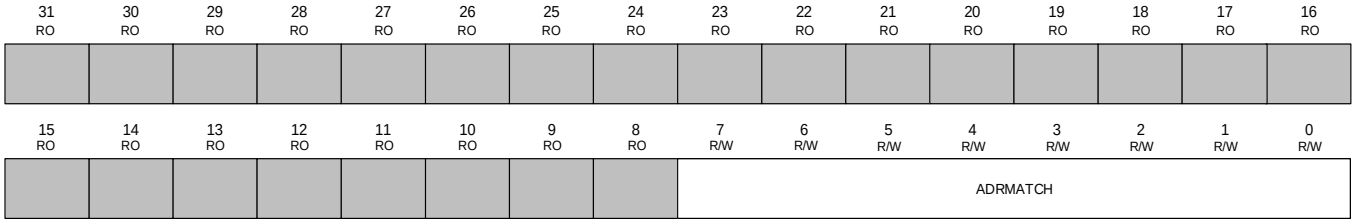
4.7.3.17 UART_485CTRL - UART RS485 CONTROL REGISTER



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

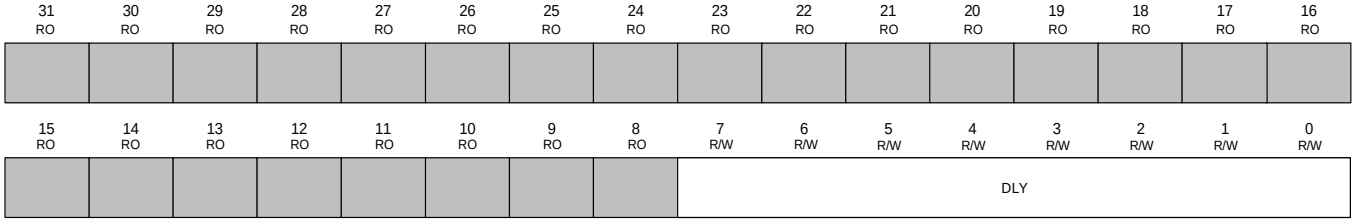


Offset: 0x0038

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

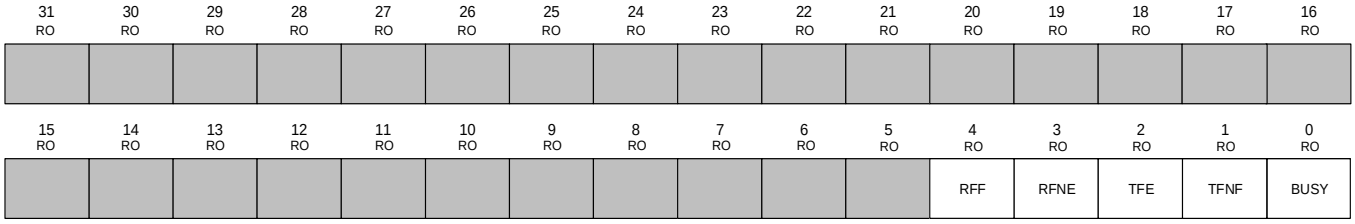
The user may program the 8-bit **UART_485DLY** register with a delay between the last stop bit leaving the TxFIFO and the de-assertion of RTSn (or CTSn). This delay time is in periods of the baud clock. Any delay time from 0 to 255 bit times may be programmed



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

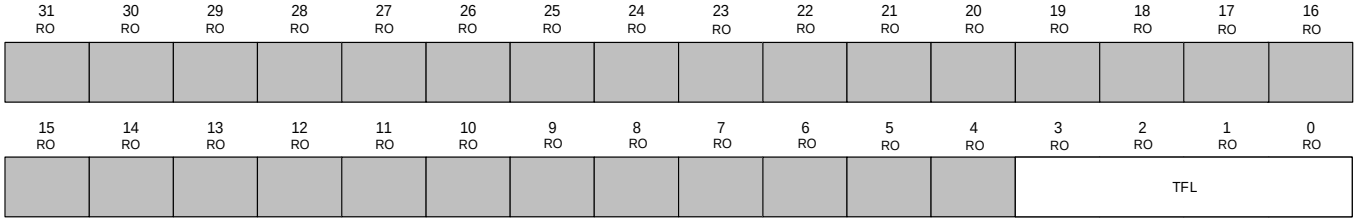


Offset: 0x007C

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.

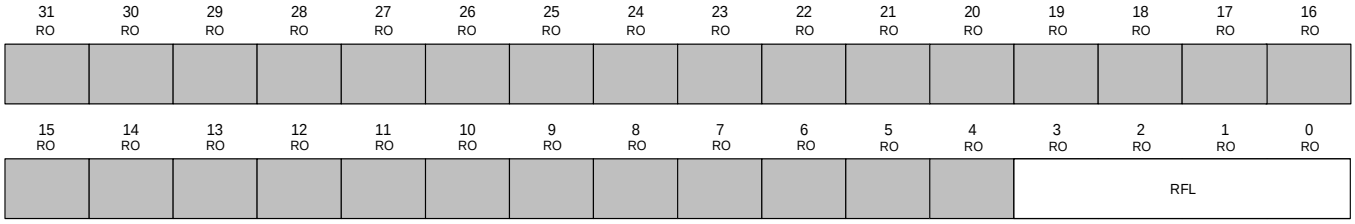


Offset: 0x0080

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.

4.8 PULSE WIDTH MODULATION (PWM)

Pulse Width Modulation (PWM) is a type of modulation that controls the width of the pulse according to the information of the modulating signal. By varying the pulse width or duty cycle the effective power applied to electrical devices can be controlled. PWM is commonly used in switching power supply and motor control applications.

PT32U301 PWM module consists of three PWM generator blocks and one control block. The control block determines the polarity of the PWM signals, and which signals are passed through to the pins.

Each PWM generator generates two PWM signals (pwmx_a' and pwmx_b', x=0, 1, 2) which can function independently with each other (using common timer frequency) or it can also be the complementary signals with programmable dead-band delay between rising and falling edges. The PWM output signals before being driven to the device pins are first managed by the output control module. The output control module selects which PWM output signals that are to be passed to the device pins (PWMx_A and PWMx_B x=0, 1, 2), and if signal inversion is to be applied.

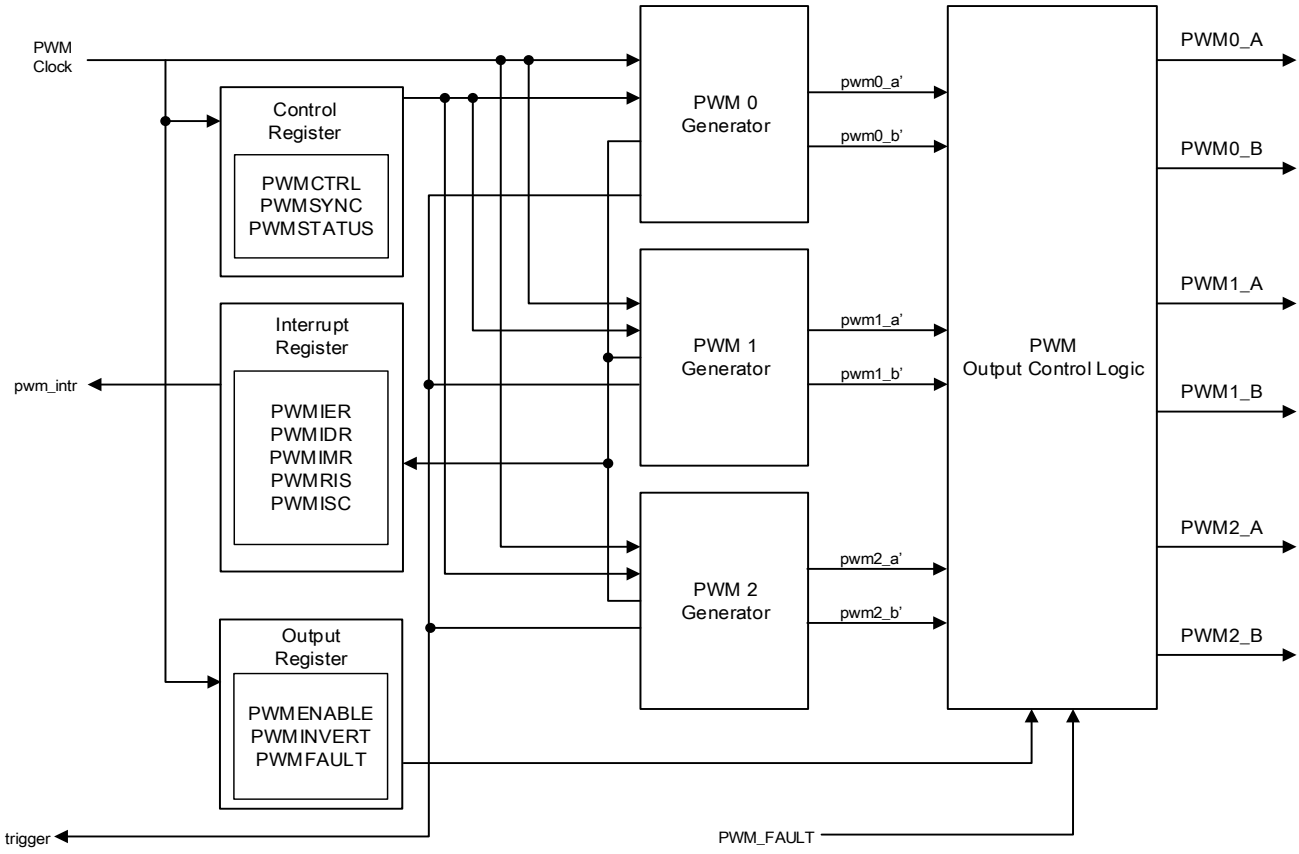
PWM module's operations are very flexible. It can produce a simple PWM signals required by charge pump or it can also produce complementary signals with dead-band delay such as commonly used in H-bridge driver circuits. The three PWM generators can produce 3-phase complementary PWM signals suitable for motor control.

Each PWM generator (PWM0, PWM1, and PWM2) has one 16-bit PWM counter, three comparators for PWM duty control, one 4-bit prescaler, one dead-band generator, an ADC trigger and one signal generator. Each PWM generator provides the following features:

- PWM Signal Generator
  - Output PWM signal is constructed based on actions taken as a result of the Timer and PWM comparator output signals
  - Produces two independent PWM signals
- One16-bit Counter
  - Runs in: Down counting, Up counting and Up/Down counting mode.
  - Output frequency controlled by a 16-bit load value
  - Load value updates can be synchronized
  - Produces output signals at zero and load value
- Three PWM comparators
  - Support synchronize update comparator value
  - Produce output signals on match
- Dead-band generator
  - Produce two PWM signals with programmable dead-band delays for driving a half-H bridge
  - Can be bypassed, leaving input signals unmodified
- PWM output can be used to trigger and start the ADC conversion.

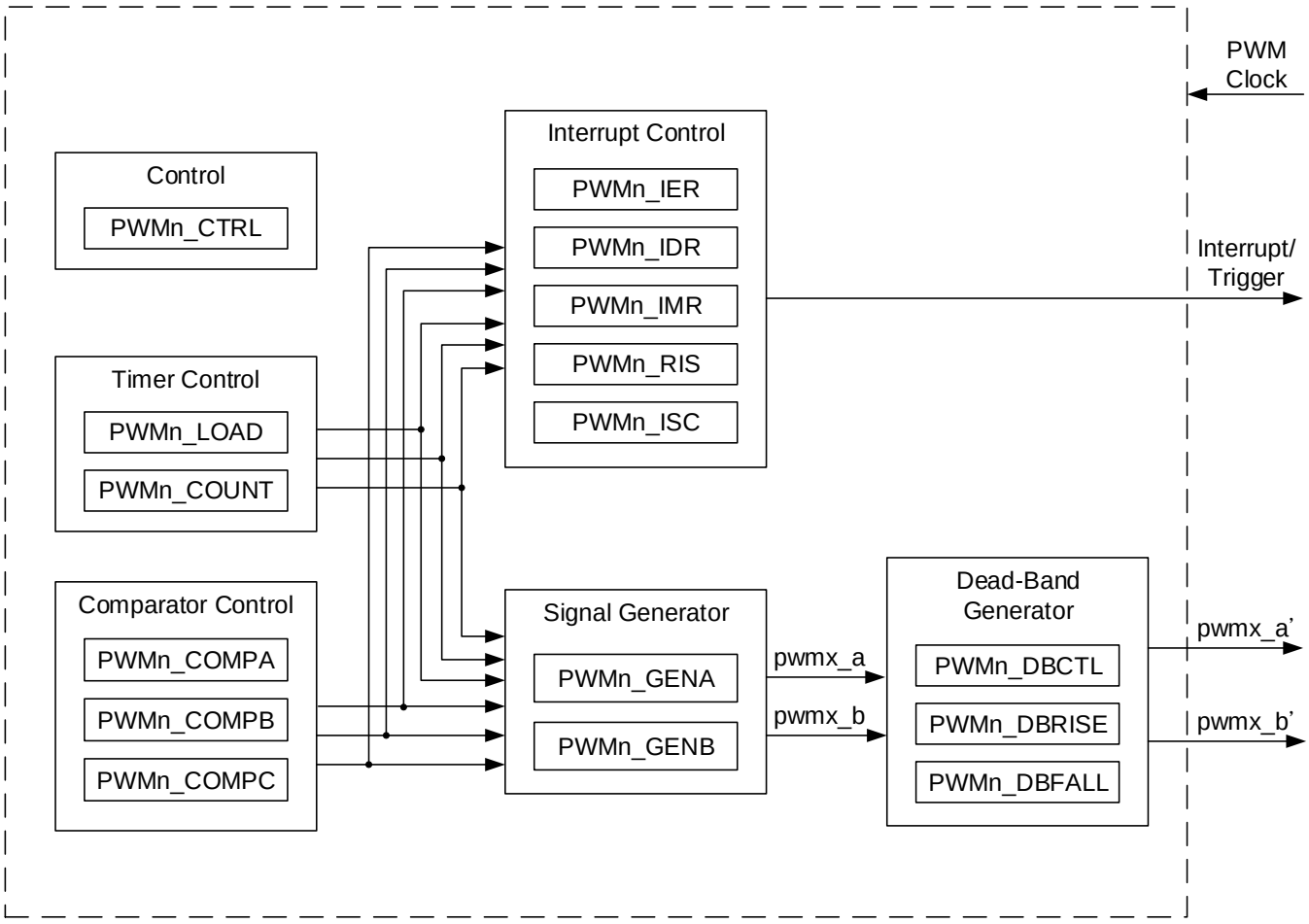
4.8.1 PWM BLOCK DIAGRAM

Figure 4.8-1: PWM Block Diagram



4.8.2 PWM SIGNAL GENERATOR BLOCK DIAGRAM

Figure 4.8-2: PWM Signal Generator Block Diagram



4.8.3 PWM FUNCTIONAL DESCRIPTION

4.8.3.1 PWM TIMER

In PT32U301, the PWM timer, configured by Counter Register (**PWMn_COUNT**), Load Value Register (**PWMn_LOAD**) and Comparator Register, supports three counting modes:

- Down Counting Mode

  Timer starts counting down from load value to zero, goes back to the load value, and continues counting down. This counting mode is generally used for right-aligned PWM.
- Up Counting Mode

  Timer starts counting from zero up to the load value then it back down to zero and continues counting up. This counting mode is generally used for left-aligned PWM.
- Up/Down Counting Mode

Timer counts from zero up to the load value and counts down from load value to zero, and so on. This counting mode is generally used for center-aligned PWM.

Right-aligned PWM is typically used in special cases requiring alignment opposite of left-aligned PWM. Left-aligned PWM is used for most general purpose PWM uses. Center-aligned PWM is generally used for AC motor control to maintain phase alignment.

PWM timer output three control signals that are used in the PWM signal generation process:

- Direction signal (labeled Direction) indicating if the timer is counting down (“0”) or counting up (“1”).
- A single-clock-cycle-width High when the counter is zero (labeled Zero).
- A single-clock-cycle-width High when the counter is equal to the load value (labeled Load).

Note: In the down counting mode, zero pulse is followed by load pulse. In the up counting mode, load pulse is followed by a zero pulse.

4.8.3.2 PWM COMPARATOR

Each PWM generator has three comparators for monitoring the counter value, when either match the counter, they output a single-clock-cycle-width High pulse, labeled “A”, “B”, “C” in this chapter

When in Up/Down Counting mode, these comparators match both when counting up and when counting down, and thus are qualified by the counter direction signal (Direction). These pulses are used in PWM signal generation process. If either comparator value is greater than the counter load value, then that comparator never outputs a High pulse. Following figures illustrate the different behavior of the counter and the relationship of these pulses under different counter mode. .

Figure 4.8-3: PWM Down Counting Mode

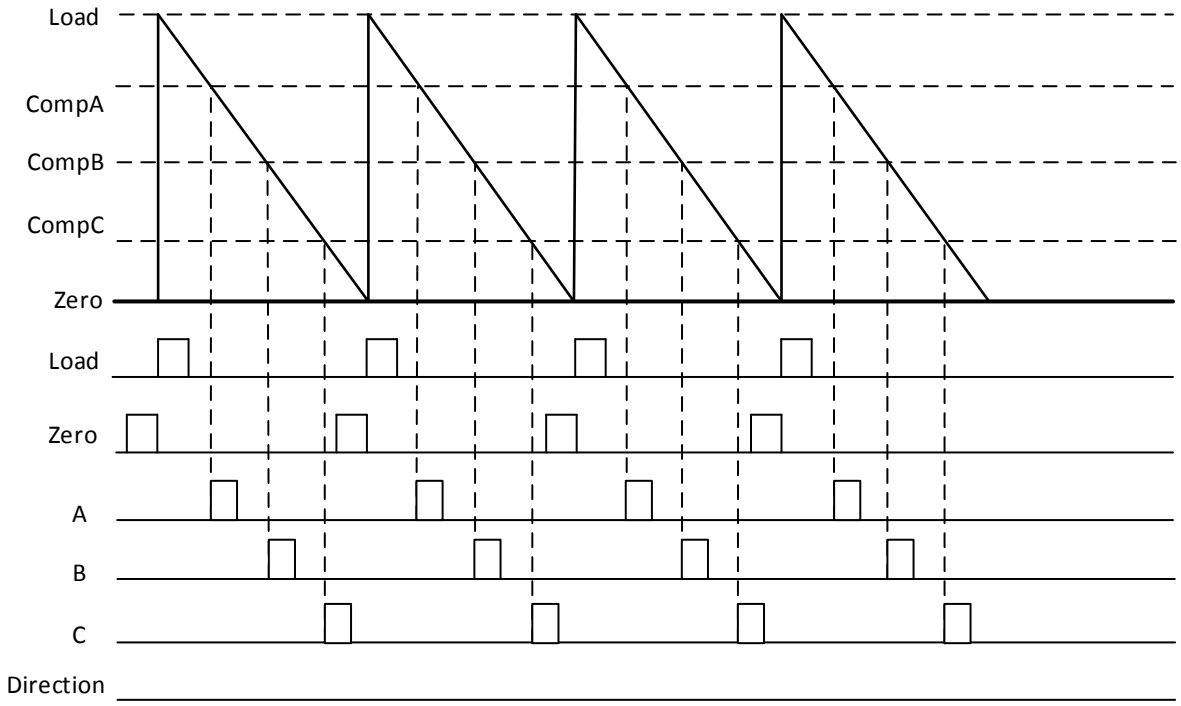


Figure 4.8-4: PWM Up Counting Mode

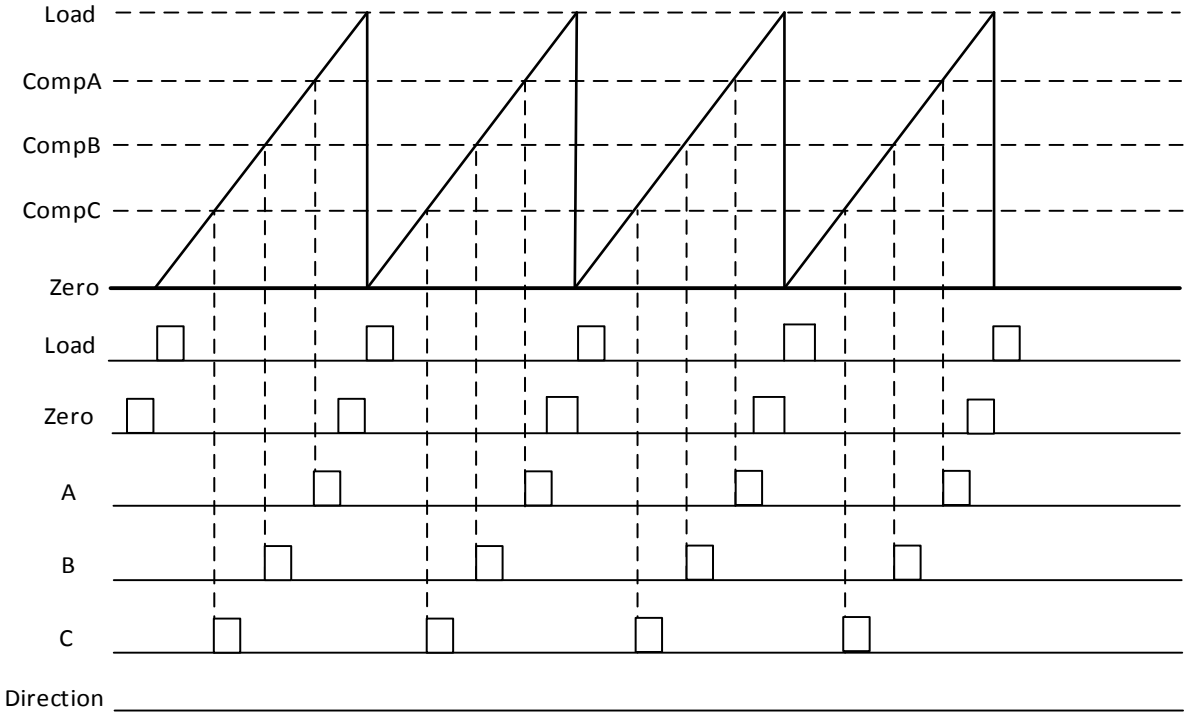
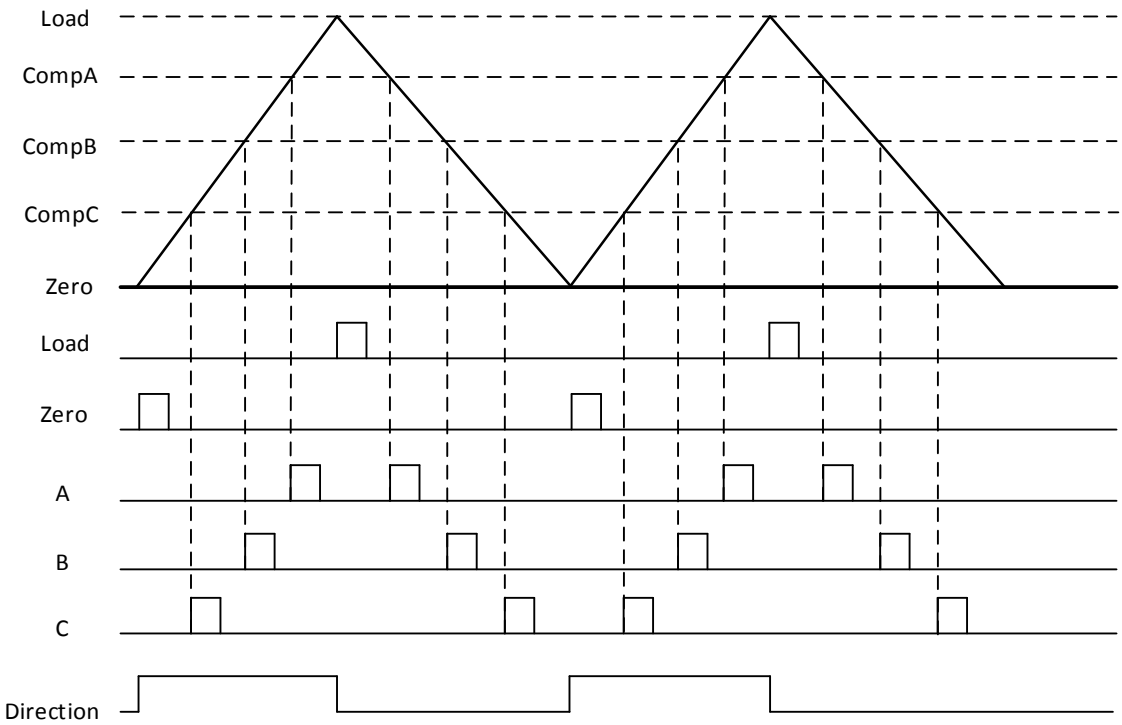


Figure 4.8-5: PWM Up/Down Counting Mode



4.8.3.3 PWM SIGNAL GENERATOR

The two internal PWM outputs (pwmx_a and pwmx_b) are defined independently of each affecting by following match events:

- Matching load value
- Matching zero
- Matching with comparator A value (Match A up/down; qualified by the Direction signal)
- Matching with comparator B value (Match B up/down; qualified by the Direction signal)
- Matching with comparator C value (Match C up/down; qualified by the Direction signal)

In the down-counting PWM output mode, the internal PWM output signals are affected by these events: (Load, Zero, Match A down, Match B down, Match C down).

In the up-counting PWM output mode, the internal PWM output signals are affected by these events: (Load, Zero, Match A up, Match B up, Match C up).

In the Up/Down counting PWM output mode, the internal PWM output signals are affected by these events: (Load, Zero, Match A down, Match B down, Match C down, Match A down, Match B down, Match C down).

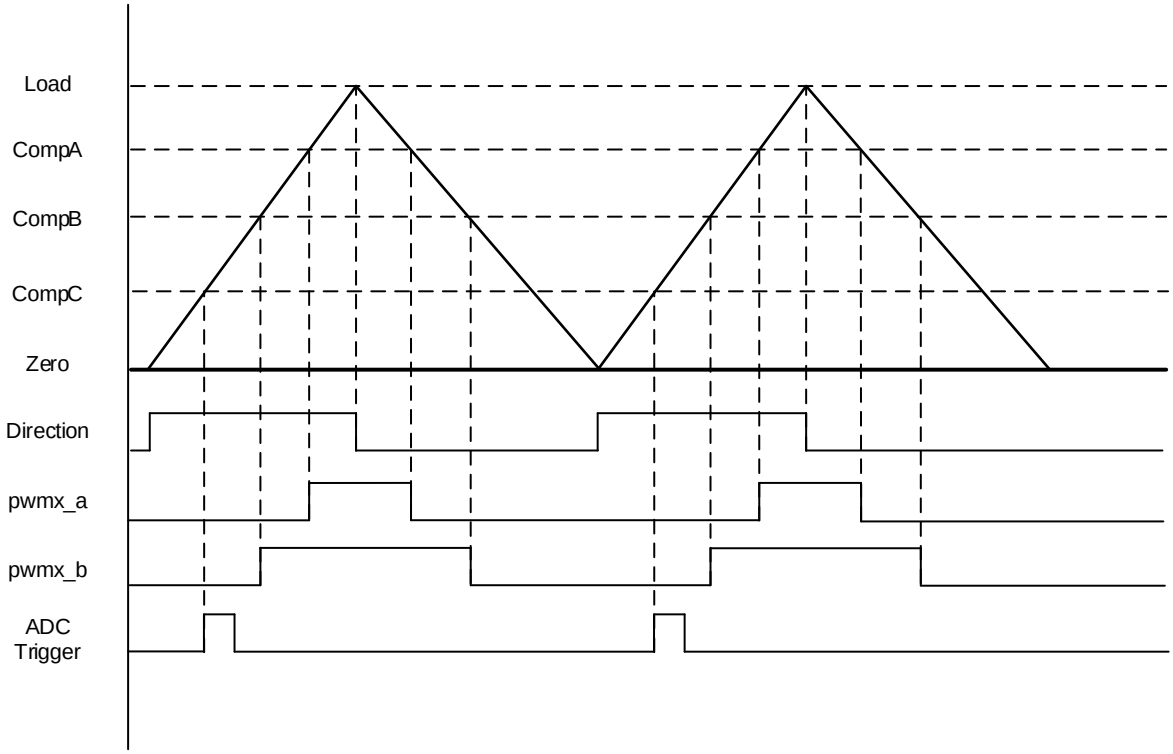
Match A or Match B or Match C event is ignored if it coincides with the zero or loading events. If Match A and Match B or Match C coincides, the first signal PWMx_A event is generated based only on Match A event, the second signal PWMx_B event is generated based only on Match B event.

The toggle action of each matching event on the PWM output signal is programmable:

- Ignore the event
- Reverse
- Drive low
- Drive high.

These actions can be used to generate a different position and duty cycle of the PWM signal, which signal may or may not overlap.

Figure 4.8-6: PWM Output Waveform Example in Count-Up/Down Mode



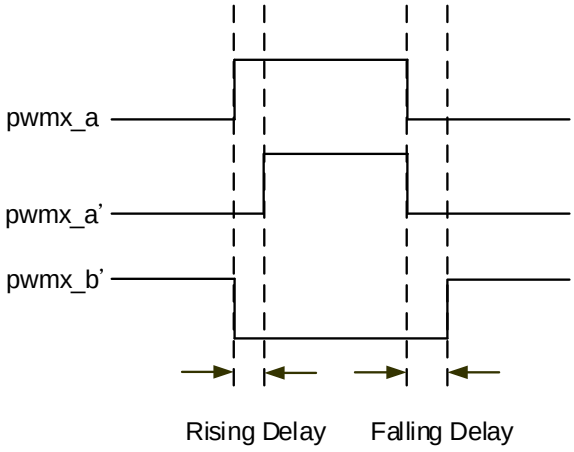
In this example, the pwmx_a is set to drive High on match A up, drive Low on match A down, and ignore other events. The pwmx_b is set to drive High on match B up, drive Low on match B down, and ignore other events. Changing the value of comparator A/B changes the duty cycle of the PWMA/PWMB signal. The Comparator C is set to send a pulse on match C up or down, however, the ADC Trigger pulse is only generated in up-counting timebase.

4.8.3.4 DEADBAND GENERATOR

Two PWM signals (pwmx_a and pwmx_b) are passed to the dead band generator. If the deadband generator is disabled, the PWM signals will pass through the module without modifying. If the deadband generator is enabled, the second PWM signal is discarded and uses the first PWM input signal to generate two PWM signals. The first PWM output signal is the same as the input signal but the rising edge is delayed by a programmable time. The second PWM output signal is the inversion of the input signal but the falling edge is delayed by a programmable time.

pwmx_a' and pwmx_b' form a non-overlapping active high signals where one is always High, except for a programmable amount of time at transitions where both are Low which is the dead-band time since both signals are inactive. When these signals are used for driving H-bridge configuration, these indicate both high side and low side NMOS transistors are off. Dead-band time is inserted between switching NMOS to prevent shoot through current. Following figure shows the effect of the dead band of the input PWM signal generator.

Figure 4.8-7: Effect of the dead band of the input PWM signal



4.8.3.5 INTERRUPT / ADC- TRIGGER SELECTOR

PWM generator also uses the same five (or eight) counter events to generate an interrupt or an ADC trigger signal. Any of these events or a set of these events can be selected as a source for an interrupt or ADC trigger signal. The interrupt or ADC trigger can be configured to happen at any point in the PWM cycle. Note that when dead-band is enabled, the interrupt and ADC trigger is based on the original (Raw) events in the PWM signal, delays in the PWM signals edges caused by the dead-band generator are not taken into account.

The interrupt or ADC trigger in PWM are controlled by a set of five registers.

- Interrupt Control (IER, IDR, IMR)

  PWM Interrupt enable register (PWM_IER) enables the interrupt request lines by writing a '1'. Similarly, **PWM Interrupt disable register (PWM_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 **PWM Interrupt Mask Register (PWM_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)

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

- Interrupt Clear (ISC)

  PWM Interrupt Status & Interrupt Clear Register (**PWM_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.

4.8.3.6 SYNCHRONIZATION METHOD

There is a global reset signal that can synchronously reset any or all of the counters in the PWM generators. If multiple PWM generators use the same counter load values then having a global reset will guarantee that the PWM generators have the same count values. The PWM generators must be configured first before executing a global reset. Thus, two or more PWM signals can be generated with a known relationship between edges since the counters always have the same values.

In the PWM generator, the comparator values can be updated in two methods. One is to update the comparator values when the counter reaches zero. This behavior is well defined that avoids too short or too long PWM output pulse.

Another method is to use a global synchronization update signal. When the counter reaches zero and the global synchronization update signal is enabled then the new value is used. This method allows multiple PWM generators to be updated simultaneously with a well-defined behavior. That means, everything runs from the old values until a point at which they all run from the new values.

4.8.3.7 FAULT STATE

There are two situation that affect the PWM module; one is a Fault input pin (PWM_FAULT), and the stalling of the controller generated by the debugger. There are two mechanisms to handle such fault conditions; force the output signals to go inactive and/or have the PWM timers stopped.

Each output signal has a corresponding fault bit control. If enabled and a fault input signal is present then the corresponding output signal will go inactive. Having the output signal go inactive during a fault condition prevents driving the external circuits in a dangerous manner. A fault condition can also generate a controller interrupt.

Each PWM generator can also be configured to stop counting whenever a stall condition is encountered. The counter can be made to stop once it reaches zero count. A stall condition does not generate an interrupt.

4.8.3.8 OUTPUT CONTROL MODULE

Each PWM generator produces two raw PWM signals and passes through the output control module for further conditioning before they are driven to the pins. The PWM Output Enable Register (**PWM_ENABLE**) is used to modify the final states of the selected PWM signals. This is very suitable specially for performing commutation of a brushless DC motor with a single write to a register. A final inversion can also be applied to any of the PWM signals using the PWM Output Polarity Control Register (**PWM_INVERT**) making them active “high” instead of active “low” and vice versa.

4.8.4 PWM REGISTER MAP

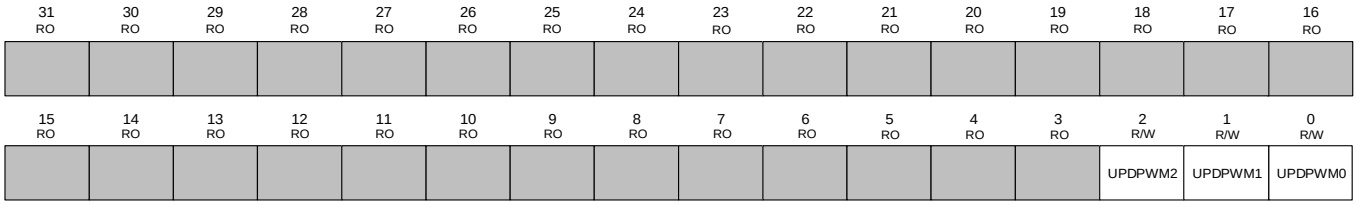
Base address: 0x4000_C000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	PWM_CTRL	R/W	0x0000_0000	PWM Main Control	126
0x0004	PWM_SYNC	R/W	0x0000_0000	PWM Timing synchronization	127
0x0008	PWM_ENABLE	R/W	0x0000_0000	PWM Output Enable	128
0x000C	PWM_INVERT	R/W	0x0000_0000	PWM Output Reverse	129
0x0010	PWM_FAULT	R/W	0x0000_0000	PWM Failure Control	130
0x0014	PWM_IER	WO	0x0000_0000	PWM Interrupt Enable	131
0x0018	PWM_IDR	WO	0x0000_0000	PWM Interrupt Disable	132
0x001C	PWM_IMR	RO	0x0000_0000	PWM Interrupt Mask	133
0x0020	PWM_RIS	RO	0x0000_0000	PWM Raw Interrupt Status	134
0x0024	PWM_ISC	R/W1C	0x0000_0000	PWM Interrupt Status and Clear	135
0x0028	PWM_STATUS	RO	0x0000_0000	PWM Fault Input Status	136
0x0040	PWM0_CTL	R/W	0x0000_0000	PWM0 Control	137
0x0044	PWM0_IER	WO	0x0000_0000	PWM0 Interrupt Enable	138
0x0048	PWM0_IDR	WO	0x0000_0000	PWM0 Interrupt Disable	140
0x004C	PWM0_IMR	RO	0x0000_0000	PWM0 Interrupt Mask	142
0x0050	PWM0_RIS	RO	0x0000_0000	PWM0 Raw Interrupt Status	144
0x0054	PWM0_ISC	R/W1C	0x0000_0000	PWM0 Interrupt Status and Clear	145
0x0058	PWM0_LOAD	R/W	0x0000_0000	PWM0 Load Value	146
0x005C	PWM0_COUNT	RO	0x0000_0000	PWM0 Counter	146
0x0060	PWM0_COMPA	R/W	0x0000_0000	PWM0 Comparator A	147
0x0064	PWM0_COMPB	R/W	0x0000_0000	PWM0 Comparator B	147
0x0068	PWM0_COMPC	R/W	0x0000_0000	PWM0 Comparator C	148
0x006C	PWM0_GENA	R/W	0x0000_0000	PWM0 Generator Control A	149
0x0070	PWM0_GENB	R/W	0x0000_0000	PWM0 Generator Control B	150
0x0074	PWM0_DBCTL	R/W	0x0000_0000	PWM0 Dead Band Control	151
0x0078	PWM0_DBRISE	R/W	0x0000_0000	PWM0 Dead Band Rise Delay	151
0x007C	PWM0_DBFALL	R/W	0x0000_0000	PWM0 Dead Band Fall Delay	153
0x0080	PWM1_CTL	R/W	0x0000_0000	PWM1 Control	137
0x0084	PWM1_IER	WO	0x0000_0000	PWM1 Interrupt Enable	138
0x0088	PWM1_IDR	WO	0x0000_0000	PWM1 Interrupt Disable	140
0x008C	PWM1_IMR	RO	0x0000_0000	PWM1 Interrupt Mask	142
0x0090	PWM1_RIS	RO	0x0000_0000	PWM1 Raw Interrupt Status	144
0x0094	PWM1_ISC	R/W1C	0x0000_0000	PWM1 Interrupt Status and Clear	145
0x0098	PWM1_LOAD	R/W	0x0000_0000	PWM1 Load Value	146
0x009C	PWM1_COUNT	RO	0x0000_0000	PWM1 Counter	146
0x00A0	PWM1_COMPA	R/W	0x0000_0000	PWM1 Comparator A	147
0x00A4	PWM1_COMPB	R/W	0x0000_0000	PWM1 Comparator B	147
0x00A8	PWM1_COMPC	R/W	0x0000_0000	PWM1 Comparator C	148
0x00AC	PWM1_GENA	R/W	0x0000_0000	PWM1Generator Control A	149
0x00B0	PWM1_GENB	R/W	0x0000_0000	PWM1 Generator Control B	150

Offset	Symbol	Type	Reset Value	Description	See page
0x00B4	PWM1_DBCTL	R/W	0x0000_0000	PWM1 Dead Band Control	151
0x00B8	PWM1_DBRISE	R/W	0x0000_0000	PWM1 Dead Band Rise Delay	151
0x00BC	PWM1_DBFALL	R/W	0x0000_0000	PWM1 Dead Band Fall Delay	153
0x00C0	PWM2_CTL	R/W	0x0000_0000	PWM2 Control	137
0x00C4	PWM2_IER	WO	0x0000_0000	PWM2 Interrupt Enable	138
0x00C8	PWM2_IDR	WO	0x0000_0000	PWM2 Interrupt Disable	140
0x00CC	PWM2_IMR	RO	0x0000_0000	PWM2 Interrupt Mask	142
0x00D0	PWM2_RIS	RO	0x0000_0000	PWM2 Raw Interrupt Status	144
0x00D4	PWM2_ISC	R/W1C	0x0000_0000	PWM2 Interrupt Status and Clear	145
0x00D8	PWM2_LOAD	R/W	0x0000_0000	PWM2 Load Value	146
0x00DC	PWM2_COUNT	RO	0x0000_0000	PWM2 Counter	146
0x00E0	PWM2_COMPA	R/W	0x0000_0000	PWM2 Comparator A	147
0x00E4	PWM2_COMPB	R/W	0x0000_0000	PWM2 Comparator B	147
0x00E8	PWM2_COMPC	R/W	0x0000_0000	PWM2 Comparator C	148
0x00EC	PWM2_GENA	R/W	0x0000_0000	PWM2Generator Control A	149
0x00F0	PWM2_GENB	R/W	0x0000_0000	PWM2 Generator Control B	150
0x00F4	PWM2_DBCTL	R/W	0x0000_0000	PWM2 Dead Band Control	151
0x00F8	PWM2_DBRISE	R/W	0x0000_0000	PWM2 Dead Band Rise Delay	151
0x00FC	PWM2_DBFALL	R/W	0x0000_0000	PWM2 Dead Band Fall Delay	153

4.8.4.1 PWM_CTRL – PWM MAIN CONTROL

The register for PWM generation module be the main control. Setting individual bit cause any queued update to a load or comparator register in the specific PWM generator x (x= 0, 1, 2) to be applied the next time the corresponding counter becomes zero. This bit automatically clears when the updates have completed.

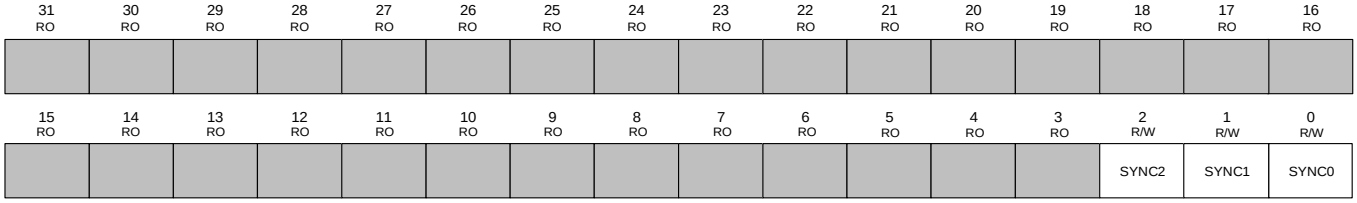


Offset: 0x0000

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	*UPDPWM2*	R/W	0	Update PWM Generator 2 0: No effect 1: Any queued update in PWM generator 2 is applied the next time the corresponding counter becomes zero.
1	*UPDPWM1*	R/W	0	Update PWM Generator 1 0: No effect 1: Any queued update in PWM generator 1 is applied the next time the corresponding counter becomes zero.
0	*UPDPWM0*	R/W	0	Update PWM Generator 0 0: No effect 1: Any queued update in PWM generator 0 is applied the next time the corresponding counter becomes zero.

4.8.4.2 PWM_SYNC - PWM TIMING SYNCHRONIZATION

This register provides a method to perform counter synchronization in the PWM generators. Writing '1' causes the specified counter to reset back to 0. Writing multiple bits allows multiple counter to be reset simultaneously. Once the reset has occurred, the bits are automatically cleared. Reading the bits back as zero indicates that synchronization reset has completed.

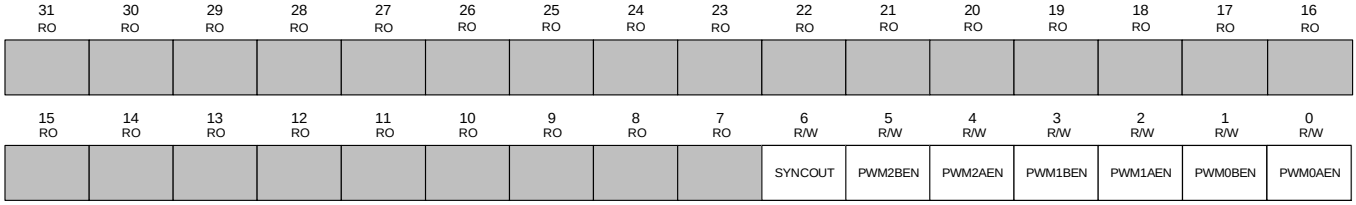


Offset: 0x0004

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	*SYNC2*	R/W	0	Reset Generator 2 Counter 0 : No effect 1: Reset PWM Generator 2 Counter
1	*SYNC1*	R/W	0	Reset Generator 1 Counter 0 : No effect 1: Reset PWM Generator 1 Counter
0	*SYNC0*	R/W	0	Reset Generator 0 Counter 0 : No effect 1: Reset PWM Generator 0 Counter

4.8.4.3 PWM_ENABLE - PWM OUTPUT ENABLE

This register provides a master control of which generated PWM signals are output to device pins. The PWM operation is still running normally even the output is disabled. Setting the bits to '1' to let the corresponding PWM signals pass through the output stage which is controlled by **PWM_INVERT** register. When the bits are cleared, the corresponding PWM signals are replaced with zero values and passed to the output stage.

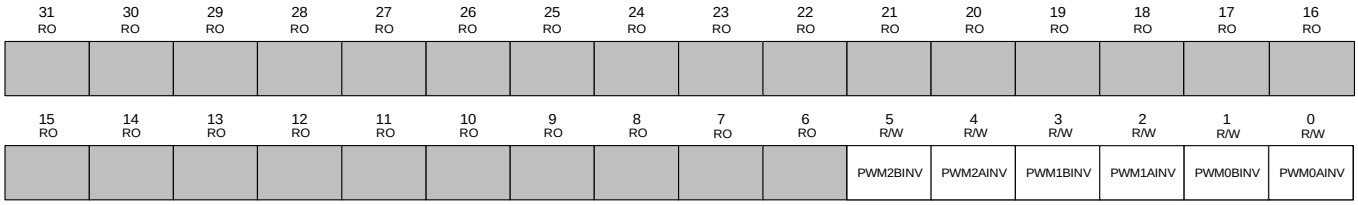


Offset: 0x0008

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	*SYNCOUT*	R/W	0	1: PWM output enable will change only after zero point
5	*PWM2BEN*	R/W	0	PWM2_B Output Enable 0: The PWM2_B signal has a zero value 1: Allows the generated PWM2_Bsignal to be passed to the device pin.
4	*PWM2AEN*	R/W	0	PWM2_A Output Enable 0: The PWM2_A signal has a zero value 1: Allows the generated PWM2_Asignal to be passed to the device pin.
3	*PWM1BEN*	R/W	0	PWM1_B Output Enable 0: The PWM1_B signal has a zero value 1: Allows the generated PWM1_Bsignal to be passed to the device pin.
2	*PWM1AEN*	R/W	0	PWM1_A Output Enable 0: The PWM1_A signal has a zero value 1: Allows the generated PWM1_Asignal to be passed to the device pin.
1	*PWM0BEN*	R/W	0	PWM0_B Output Enable 0: The PWM0_B signal has a zero value 1: Allows the generated PWM0_Bsignal to be passed to the device pin.
0	*PWM0AEN*	R/W	0	PWM0_A Output Enable 0: The PWM0_A signal has a zero value 1: Allows the generated PWM0_Asignal to be passed to the device pin.

4.8.4.4 PWM_INVERT - PWM OUTPUT POLARITY CONTROL

This register provides a master control of the polarity of the PWM signals before driving the device pins. The PWM signals generated by the PWM generator are active High which can optionally be made active Low via this register.

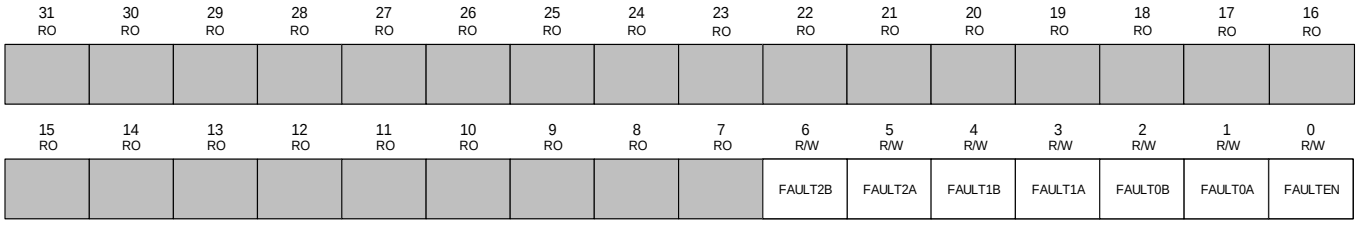


Offset: 0x000C

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	*PWM2BINV*	R/W	0	Invert PWM2_B Signal 0: No signal inversion 1: Inverts PWM2_B signal
4	*PWM2AINV*	R/W	0	Invert PWM2_A Signal 0: No signal inversion 1: Inverts PWM2_A signal
3	*PWM1BINV*	R/W	0	Invert PWM1_B Signal 0: No signal inversion 1: Inverts PWM1_B signal
2	*PWM1AINV*	R/W	0	Invert PWM1_A Signal 0: No signal inversion 1: Inverts PWM1_A signal.
1	*PWM0BINV*	R/W	0	Invert PWM0_B Signal 0: No signal inversion 1: Inverts PWM0_B signal
0	*PWM0AINV*	R/W	0	Invert PWM0_A Signal 0: No signal inversion 1: Inverts PWM0_A signal

4.8.4.5 PWM_FAULT - PWM FAILURE CONTROL

This register is used to control the behavior of the PWM output when a fault condition occurs. Fault input and debug events can be seen as a fault condition. Under fault conditions, each of the PWM signal can be pass through unmodified or drive low. For configured pass-through outputs debug event handler will be responsible whether to continue with the PWM signal generation. Fault condition occurs before the output inverter. So, PWM signals driven low due to fault condition will pass through the signal inversion before driving the pins.

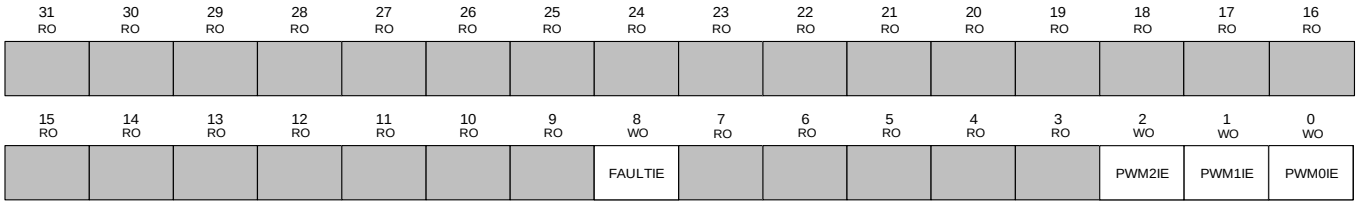


Offset: 0x0010

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	*FAULT2B*	R/W	0	PWM2_B Fault 0: The generated pwm2_b' signal is passed to the PWM2_B pin. 1: The PWM2_B output signal is driven Low on a fault condition
5	*FAULT2A*	R/W	0	PWM2_A Fault 0: The generated pwm2_a' signal is passed to the PWM2_A pin. 1: The PWM2_A output signal is driven Low on a fault condition
4	*FAULT1B*	R/W	0	PWM1_B Fault 0: The generated pwm1_b' signal is passed to the PWM1_B pin. 1: The PWM1_B output signal is driven Low on a fault condition
3	*FAULT1A*	R/W	0	PWM1_A Fault 0: The generated pwm1_a' signal is passed to the PWM1_A pin. 1: The PWM1_A output signal is driven Low on a fault condition
2	*FAULT0A*	R/W	0	PWM0_B Fault 0: The generated pwm0_b' signal is passed to the PWM0_B pin. 1: The PWM0_B output signal is driven Low on a fault condition
1	*FAULT0B*	R/W	0	PWM0_A Fault 0: The generated pwm0_a' signal is passed to the PWM0_A pin. 1: The PWM0_A output signal is driven Low on a fault condition
0	*FAULTEN*	R/W	0	PWM Fault software Enable 0: Disable fault condition control by software 1: Enable fault condition control

4.8.4.6 PWM_IER - PWM INTERRUPT ENABLE

This register controls the global interrupt generation capabilities of the PWM module. The events that can cause an interrupt are the fault input and the individual interrupts from the PWM generators. Use **PWMn_IER** to control individual PWM interrupt enable function

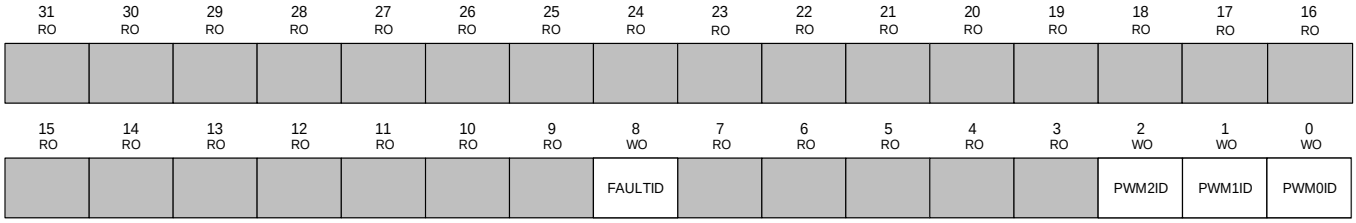


Offset: 0x0014

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	*FAULTIE*	WO	0	Fault Interrupt Enable 1: An interrupt occurs when the fault input is asserted.
7:3	*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	*PWM2IE*	WO	0	PWM2 Interrupt Enable 1: An interrupt occurs when the PWM generator 2 block asserts an interrupt.
1	*PWM1IE*	WO	0	PWM1 Interrupt Enable 1: An interrupt occurs when the PWM generator 1 block asserts an interrupt.
0	*PWM0IE*	WO	0	PWM0 Interrupt Enable 1: An interrupt occurs when the PWM generator 0 block asserts an interrupt.

4.8.4.7 PWM_IDR - PWM INTERRUPT DISABLE

This register controls the global interrupt generation capabilities of the PWM module. The events that can cause an interrupt are the fault input and the individual interrupts from the PWM generators. Use **PWMn_IDR** to control individual PWM interrupt disable function

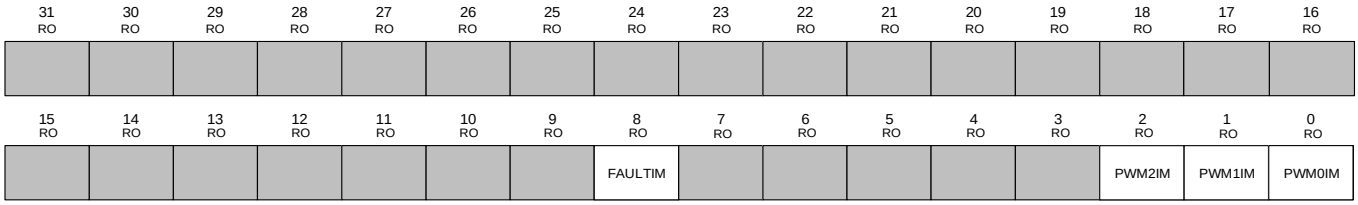


Offset: 0x0018

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	*FAULTID*	WO	0	Fault Interrupt Disable 1: An interrupt would not occurs when the fault input is asserted.
7:3	*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	*PWM2ID*	WO	0	PWM2 Interrupt Disable 1: An interrupt would not occurs when the PWM generator 2 block asserts an interrupt.
1	*PWM1ID*	WO	0	PWM1 Interrupt Disable 1: An interrupt would not occurs when the PWM generator 1 block asserts an interrupt.
0	*PWM0ID*	WO	0	PWM0 Interrupt Disable 1: An interrupt would not occurs when the PWM generator 0 block asserts an interrupt.

4.8.4.8 PWM_IMR - PWM INTERRUPT MASK

The register shows the global PWM module interrupt mask status. Use **PWMn_IMP** to read individual interrupt mask status.

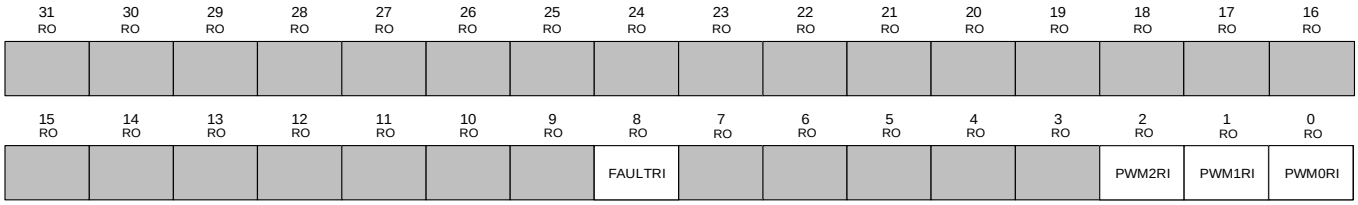


Offset: 0x001C

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	*FAULTIM*	RO	0	Fault Interrupt Mask status 0: Fault input interrupt will be masked. 1: Fault input interrupt is enabled.
7:3	*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	*PWM2IM*	RO	0	PWM2 Interrupt Mask status 0: PWM2 input interrupt will be masked. 1: PWM2 input interrupt is enabled.
1	*PWM1IM*	RO	0	PWM1 Interrupt Mask status 0: PWM1 input interrupt will be masked. 1: PWM1 input interrupt is enabled
0	*PWM0IM*	RO	0	PWM0 Interrupt Mask status 0: PWM0 input interrupt will be masked. 1: PWM0 input interrupt is enabled

4.8.4.9 PWM_RIS - PWM RAW INTERRUPT STATUS

This register provides the current set of interrupt sources that are asserted regardless of whether they cause an interrupt to be asserted by the controller. The fault interrupt is latched on detection so it must be cleared via PWM_ISC register.

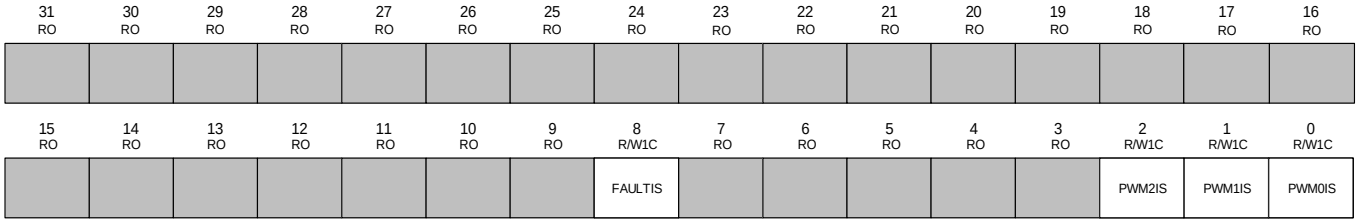


Offset: 0x0020

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	*FAULTRI*	RO	0	Fault Raw Interrupt status 0: No interrupt 1: Fault input is asserting.
7:3	*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	*PWM2RI*	RO	0	PWM2 Raw Interrupt status 0: No interrupt 1: PWM2 is asserting its interrupt
1	*PWM1RI*	RO	0	PWM1 Raw Interrupt status 0: No interrupt 1: PWM1 is asserting its interrupt
0	*PWM0RI*	RO	0	PWM0 Raw Interrupt status 0: No interrupt 1: PWM0 is asserting its interrupt

4.8.4.10 PWM_ISC - PWM INTERRUPT STATUS AND CLEAR REGISTER

This register provides the summary of the interrupt status of individual PWM generators. A bit set to '1' indicates an active interrupt. The corresponding PWM generator interrupt status register can be consulted to know the specific source of interrupt. For the fault interrupt, writing a '1' to that bit position clears the latched interrupt status. A write of '1' to individual bit clears the respective interrupt status.

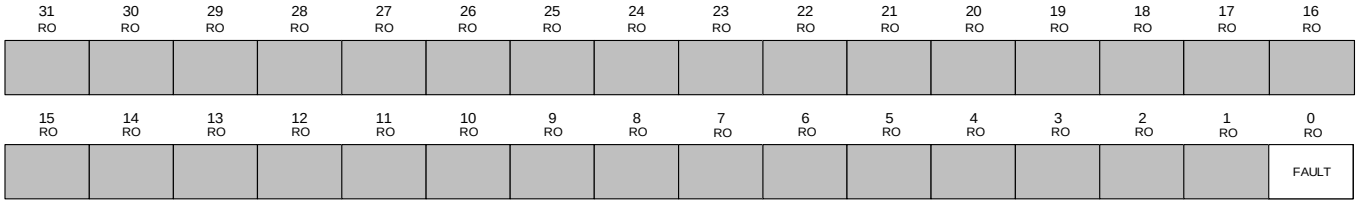


Offset: 0x0024

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	*FAULTIS*	R/W1C	0	Fault Interrupt Status and clear 0: Fault input has not asserted its interrupt or the interrupt is masked. 1: Fault input interrupt has been signaled.
7:3	*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	*PWM2IS*	R/W1C	0	PWM2 Interrupt Status and clear 0: PWM Generator 2 has not asserted its interrupt or the interrupt is masked. 1: PWM Generator 2 interrupt has been signaled.
1	*PWM1IS*	R/W1C	0	PWM1 Interrupt Status and clear 0: PWM Generator 1 has not asserted its interrupt or the interrupt is masked. 1: PWM Generator 1 interrupt has been signaled.
0	*PWM0IS*	R/W1C	0	PWM0 Interrupt Status and clear 0: PWM Generator 0 has not asserted its interrupt or the interrupt is masked. 1: PWM Generator 0 interrupt has been signaled.

4.8.4.11 PWM_STATUS - PWM FAULT INPUT STATUS

This register is used to display the status of the FAULT input signal.

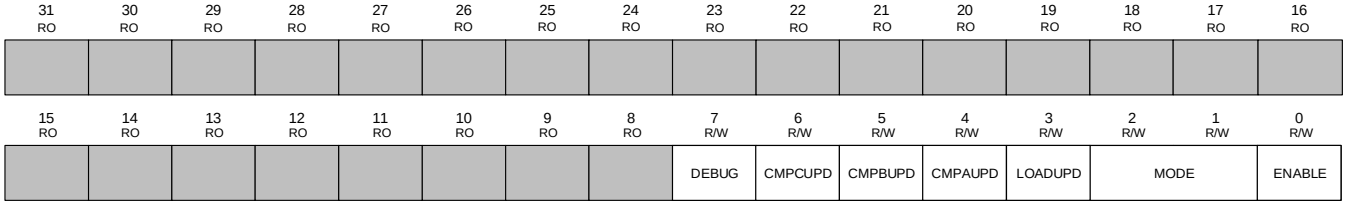


Offset: 0x0028

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	*FAULT*	RO	0	Fault Interrupt Status 0: The fault condition is not asserted 1: In Read operation, indicates the fault input is asserted. In Write operation, triggers FAULT.

4.8.4.12 PWMN_CTL - PWM0/1/2 CONTROL

These registers configure the PWM signal generation. The register update mode, debug mode, counting mode and PWM block enable are all controlled via these registers. The PWM generators produce PWM signals which can be either two independent PWM signals (with common counter) or a pair of PWM signals with dead-band delay inserted.



Offset:

PWM0_CTL: 0x0040
PWM1_CTL: 0x0080
PWM2_CTL: 0x00C0

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	*DEBUG*	R/W	0	Debug mode The behavior of the counter in Debug mode. 0: The counter always runs. 1: Counter stops running when it next reaches 0, and continues running again when no longer in Debug mode.
6	*CMPCUPD*	R/W	0	Comparator C Update mode The Update mode for the comparator A register. 0: Updates to the register are reflected to the comparator the next time the counter is 0. 1: Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the PWM Main Control (PWM_CTRL).
5	*CMPBUPD*	R/W	0	Comparator B Update mode Same description as in CMPCUPD but only for the comparator B register.
4	*CMPAUPD*	R/W	0	Comparator A Update mode Same description as in CMPCUPD but only for the comparator A register.
3	*LOADUPD*	R/W	0	Load register Update mode. The Update mode for the load register. 0: Updates to the register are reflected to the counter the next time the counter is 0. 1: Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the PWM Main Control(PWM_CTRL)
2:1	*MODE*	R/W	0	Counter Mode 00: Down Counting Mode 01: Up Counting Mode 10: Up/Down Counting Mode 11: Reserved
0	*ENABLE*	R/W	0	PWM block Enable Master enable for the PWM generation block. 0: Module is disabled and not clocked. 1; Module is enabled and produces PWM signals.

4.8.4.13 PWMN_IER - PWM0/1/2 INTERRUPT ENABLE

These registers control the interrupt and ADC trigger capabilities of PWM generators. Following lists the events that can cause an interrupt or an ADC trigger: Any combination of these events can generate either an interrupt or an ADC trigger.

The counter being equal to the load register.

The counter being equal to zero.

The counter being equal to comparator A register while counting up

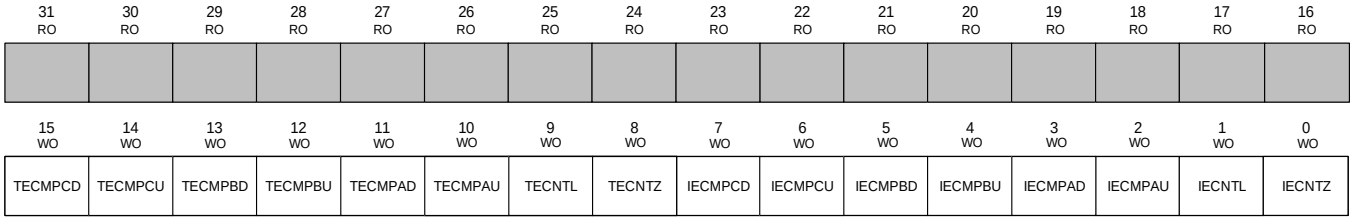
The counter being equal to comparator A register while counting down

The counter being equal to comparator B register while counting up

The counter being equal to comparator B register while counting down

The counter being equal to comparator C register while counting up

The counter being equal to comparator C register while counting down



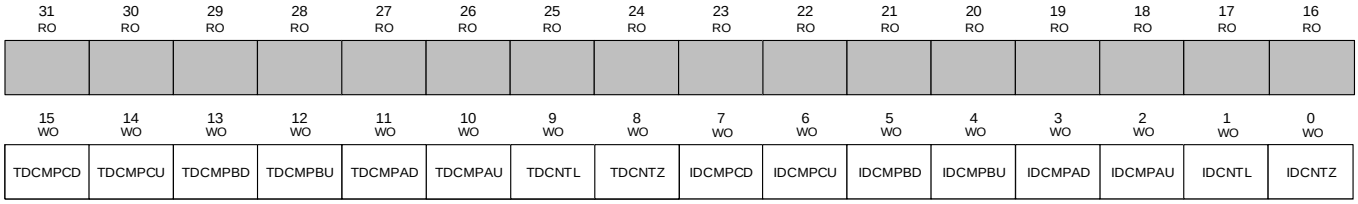
Offset:
PWM0_IER: 0x0044
PWM1_IER: 0x0084
PWM2_IER: 0x00C4

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	*TECMPCD*	WO	0	Trigger Enable when Counter = Comparator C Down 0: No ADC trigger 1: ADC trigger is generated when the counter matches the value in PWMn_COMPC in counting down mode.
14	*TECMPCU*	WO	0	Trigger Enable when Counter = Comparator C Up 0: No ADC trigger 1: ADC trigger is generated when the counter matches the value in PWMn_COMPC in counting up mode.
13	*TECMPBD*	WO	0	Trigger Enable when Counter = Comparator B Down 0: No ADC trigger 1: ADC trigger is generated when the counter matches the value in PWMn_COMPB in counting down mode.
12	*TECMPBU*	WO	0	Trigger Enable when Counter = Comparator B Up 0: No ADC trigger 1: ADC trigger is generated when the counter matches the value in PWMn_COMPB in counting up mode.
11	*TECMPAD*	WO	0	Trigger Enable when Counter = Comparator A Down 0: No ADC trigger 1: ADC trigger is generated when the counter matches the value in PWMn_COMPA in counting down mode.
10	*TECMPAU*	WO	0	Trigger Enable when Counter = Comparator A Up 0: No ADC trigger 1: ADC trigger is generated when the counter matches the value in PWMn_COMPA in counting up mode.

Bits	Name	Type	Reset	Description
9	*TECNTL*	WO	0	Trigger Enable when Counter = Load 0: No ADC trigger is output 1: An ADC trigger pulse is output when the counter matches the PWMn_LOAD register.
8	*TECNTZ*	WO	0	Trigger Enable when Counter = Zero 0: No ADC trigger is output 1: An ADC trigger pulse is output when the counter is 0.
7	*IECMPCD*	WO	0	Interrupt Enable for Counter = Comparator C Down 0: No interrupt 1: A raw interrupt occurs when the counter matches the value in the PWMn_COMPC register value while counting down.
6	*IECMPCU*	WO	0	Interrupt Enable for Counter = Comparator C Up 0: No interrupt 1: A raw interrupt occurs when the counter matches the value in the PWMn_COMPC register value while counting up.
5	*IECMPBD*	WO	0	Interrupt Enable for Counter = Comparator B Down 0: No interrupt 1: A raw interrupt occurs when the counter matches the value in the PWMn_COMPB register value while counting down.
4	*IECMPBU*	WO	0	Interrupt Enable for Counter = Comparator B Up 0: No interrupt 1: A raw interrupt occurs when the counter matches the value in the PWMn_COMPB register value while counting up.
3	*IECMPAD*	WO	0	Interrupt Enable for Counter = Comparator A Down 0: No interrupt 1: A raw interrupt occurs when the counter matches the value in the PWMn_COMPA register value while counting down.
2	*IECMPAU*	WO	0	Interrupt Enable for Counter = Comparator A Up 0: No interrupt 1: A raw interrupt occurs when the counter matches the value in the PWMn_COMPA register value while counting up.
1	*IECNTL*	WO	0	Interrupt Enable for Counter = Load 0: No interrupt 1: A raw interrupt occurs when the counter matches the values in the PWMn_LOAD register value.
0	*IECNTZ*	WO	0	Interrupt Enable for Counter = Zero 0: No interrupt 1: A raw interrupt occurs when the counter is zero.

4.8.4.14 PWMN_IDR - PWM0/1/2 INTERRUPT DISABLE

These registers control the ability to disable the PWM generator interrupt and ADC trigger signals.



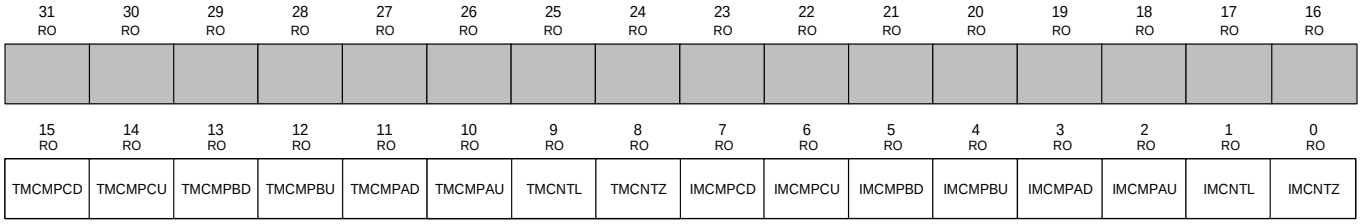
Offset:
PWM0_IDR: 0x0048
PWM1_IDR: 0x0088
PWM2_IDR: 0x00C8

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	*TDCMPCD*	WO	0	Trigger Disable for Counter = Comparator C Down 0: No ADC trigger 1: Disable ADC trigger pulse output when the counter matches the value in PWMn_COMPC in counting down mode.
14	*TDCMPCU*	WO	0	Trigger Disable for Counter = Comparator C Up 0: No ADC trigger 1: Disable ADC trigger pulse output when the counter matches the value in PWMn_COMPC in counting up mode.
13	*TDCMPBD*	WO	0	Trigger Disable for Counter = Comparator B Down 0: No ADC trigger 1: Disable ADC trigger pulse output when the counter matches the value in PWMn_COMPB in counting down mode.
12	*TDCMPBU*	WO	0	Trigger Disable for Counter = Comparator B Up 0: No ADC trigger 1: Disable ADC trigger pulse output when the counter matches the value in PWMn_COMPB in counting up mode.
11	*TDCMPAD*	WO	0	Trigger Disable for Counter = Comparator A Down 0: No ADC trigger 1: Disable ADC trigger pulse output when the counter matches the value in PWMn_COMPA in counting down mode.
10	*TDCMPAU*	WO	0	Trigger Disable for Counter = Comparator A Up 0: No ADC trigger 1: Disable ADC trigger pulse output when the counter matches the value in PWMn_COMPA in counting up mode.
9	*TDCNTL*	WO	0	Trigger Disable for Counter = Load 0: No ADC trigger is output 1: Disable ADC trigger pulse output when the counter matches the PWMn_LOAD register.
8	*TDCNTZ*	WO	0	Trigger Disable for Counter = Zero 0: No ADC trigger is output 1: Disable ADC trigger pulse output when the counter is 0.
7	*IDCMPCD*	WO	0	Interrupt Disable for Counter = Comparator C Down 0: No interrupt

Bits	Name	Type	Reset	Description
				1: A raw interrupt is disabled when the counter matches the value in the PWMn_COMPC register value while counting down.
6	*IDCMPCU*	WO	0	Interrupt Disable for Counter = Comparator C Up 0: No interrupt 1: A raw interrupt is disabled when the counter matches the value in the PWMn_COMPC register value while counting up.
5	*IDCMPBD*	WO	0	Interrupt Disable for Counter = Comparator B Down 0: No interrupt 1: A raw interrupt is disabled when the counter matches the value in the PWMn_COMPB register value while counting down.
4	*IDCMPBU*	WO	0	Interrupt Disable for Counter = Comparator B Up 0: No interrupt 1: A raw interrupt is disabled when the counter matches the value in the PWMn_COMPB register value while counting up.
3	*IDCMPAD*	WO	0	Interrupt Disable for Counter = Comparator A Down 0: No interrupt 1: A raw interrupt is disabled when the counter matches the value in the PWMn_COMPA register value while counting down.
2	*IDCMPAU*	WO	0	Interrupt Disable for Counter = Comparator A Up 0: No interrupt 1: A raw interrupt is disabled when the counter matches the value in the PWMn_COMPA register value while counting up.
1	*IDCNTL*	WO	0	Interrupt Disable for Counter = Load 0: No interrupt 1: A raw interrupt is disabled when the counter matches the values in the PWMn_LOAD register value.
0	*IDCNTZ*	WO	0	Interrupt Disable for Counter = Zero 0: No interrupt 1: A raw interrupt is disabled when the counter is zero.

4.8.4.15 PWMN_IMR - PWM0/1/2 INTERRUPT MASK

The register shows the states of the interrupt mask and ADC trigger signal.



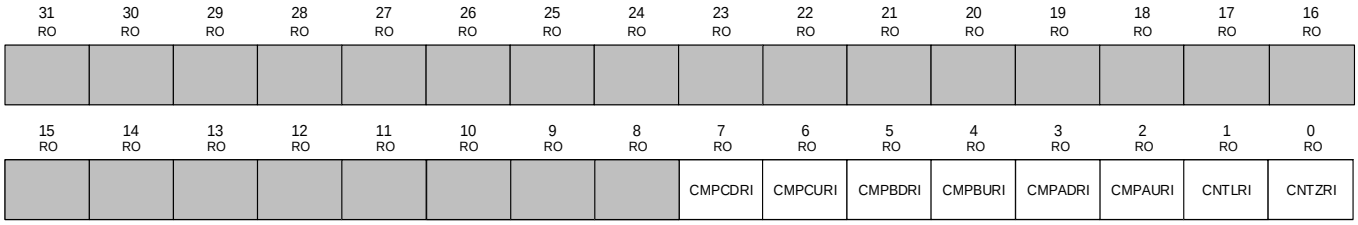
Offset:
PWM0_IMR: 0x004C
PWM1_IMR: 0x008C
PWM2_IMR: 0x00CC

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	*TMCMPCD*	RO	0	Trigger Mask Counter = Comparator C Down 0: ADC trigger pulse output interrupt will be masked when the counter matches the value in PWMn_COMPC in counting down mode. 1: ADC trigger pulse output interrupt is enabled
14	*TMCMPCU*	RO	0	Trigger Mask for Counter = Comparator C Up 0: ADC trigger pulse output interrupt will be masked when the counter matches the value in PWMn_COMPC in counting up mode. 1: ADC trigger pulse output interrupt is enabled
13	*TMCMPBD*	RO	0	Trigger Mask for Counter = Comparator B Down 0: ADC trigger pulse output interrupt will be masked when the counter matches the value in PWMn_COMPB in counting down mode. 1: ADC trigger pulse output interrupt is enabled
12	*TMCMPBU*	RO	0	Trigger Mask for Counter = Comparator B Up 0: ADC trigger pulse output interrupt will be masked when the counter matches the value in PWMn_COMPB in counting up mode. 1: ADC trigger pulse output interrupt is enabled
11	*TMCMPAD*	RO	0	Trigger Mask for Counter = Comparator A Down 0: ADC trigger pulse output interrupt will be masked when the counter matches the value in PWMn_COMPA in counting down mode. 1: ADC trigger pulse output interrupt is enabled
10	*TMCMPAU*	RO	0	Trigger Mask for Counter = Comparator A Up 0: ADC trigger pulse output interrupt will be masked when the counter matches the value in PWMn_COMPA in counting up mode. 1: ADC trigger pulse output interrupt is enabled
9	*TMCNTL*	RO	0	Trigger Mask for Counter = Load 0: ADC trigger pulse output interrupt will be masked when the counter matches the PWMn_LOAD register. 1: ADC trigger pulse output interrupt is enabled
8	*TMCNTZ*	RO	0	Trigger Mask for Counter = Zero 0: ADC trigger pulse output interrupt will be masked when the counter is 0. 1: ADC trigger pulse output interrupt is enabled

Bits	Name	Type	Reset	Description
7	*IMCMPCD*	RO	0	Interrupt Mask for Counter = Comparator C Down 0: A raw interrupt will be masked when the counter matches the value in the PWMn_COMPC register value while counting down. 1: A raw interrupt is enabled.
6	*IMCMPCU*	RO	0	Interrupt Mask for Counter = Comparator C Up 0: A raw interrupt will be masked when the counter matches the value in the PWMn_COMPC register value while counting up. 1: A raw interrupt is enabled.
5	*IMCMPBD*	RO	0	Interrupt Mask for Counter = Comparator B Down 0: A raw interrupt will be masked when the counter matches the value in the PWMn_COMPB register value while counting down. 1: A raw interrupt is enabled.
4	*IMCMPBU*	RO	0	Interrupt Mask for Counter = Comparator B Up 0: A raw interrupt will be masked when the counter matches the value in the PWMn_COMPB register value while counting up. 1: A raw interrupt is enabled.
3	*IMCMPAD*	RO	0	Interrupt Mask for Counter = Comparator A Down 0: A raw interrupt will be masked when the counter matches the value in the PWMn_COMPA register value while counting down. 1: A raw interrupt is enabled.
2	*IMCMPAU*	RO	0	Interrupt Mask for Counter = Comparator A Up 0: A raw interrupt will be masked when the counter matches the value in the PWMn_COMPA register value while counting up. 1: A raw interrupt is enabled.
1	*IMCNTL*	RO	0	Interrupt Mask for Counter = Load 0: A raw interrupt will be masked when the counter matches the values in the PWMn_LOAD register value. 1: A raw interrupt is enabled.
0	*IMCNTZ*	RO	0	Interrupt Mask for Counter = Zero 0: A raw interrupt will be masked when the counter is zero. 1: A raw interrupt is enabled.

4.8.4.16 PWMN_RIS - PWM0/1/2 RAW INTERRUPT STATUS

These registers provide the current set of interrupt sources that are asserted regardless of whether they will cause an interrupt to be asserted to the controller.

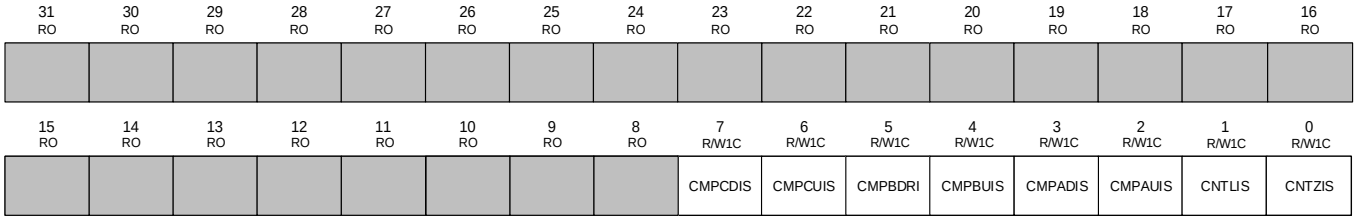


Offset:
PWM0_RIS: 0x0050
PWM1_RIS: 0x0090
PWM2_RIS: 0x00D0

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	*CMPCDRI*	RO	0	Comparator C Down Raw Interrupt status 0: No Interrupt 1: The counter has matched the value in PWMn_COMPC while counting down.
6	*CMPCURI*	RO	0	Comparator C Up Raw Interrupt status 0: No Interrupt 1: The counter has matched the value in PWMn_COMPC while counting up.
5	*CMPBDRI*	RO	0	Comparator B Down Raw Interrupt status 0: No Interrupt 1: The counter has matched the value in PWMn_COMPB while counting down.
4	*CMPBURI*	RO	0	Comparator B Up Raw Interrupt status 0: No Interrupt 1: The counter has matched the value in PWMn_COMPB while counting up.
3	*CMPADRI*	RO	0	Comparator A Down Raw Interrupt status 0: No Interrupt 1: The counter has matched the value in PWMn_COMPA while counting down.
2	*CMPAURI*	RO	0	Comparator A Up Raw Interrupt status 0: No Interrupt 1: The counter has matched the value in PWMn_COMPA while counting up.
1	*CNTLRI*	RO	0	Counter = Load Raw Interrupt status 0: No Interrupt 1: The counter has matched the value in PWMn_LOAD.
0	*CNTZRI*	RO	0	Counter = Zero Raw Interrupt status 0: No Interrupt 1: The counter has matched zero.

4.8.4.17 PWMN_ISC - PWM0/1/2 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:

PWM0_ISC: 0x0054

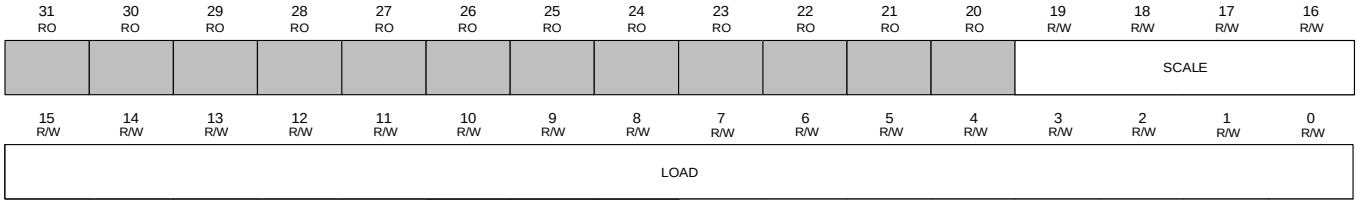
PWM1_ISC: 0x0094

PWM2_ISC: 0x00D4

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	*CMPCDIS*	R/W1C	0	Comparator C Down Interrupt Status and clear 0: No interrupt or the interrupt is masked. 1: Comparator C Down Interrupt has been signaled.
6	*CMPCUIS*	R/W1C	0	Comparator C Up Interrupt Status and clear 0: No interrupt or the interrupt is masked. 1: Comparator C Up Interrupt has been signaled.
5	*CMPBDIS*	R/W1C	0	Comparator B Down Interrupt Status and clear 0: No interrupt or the interrupt is masked. 1: Comparator B Down Interrupt has been signaled.
4	*CMPBUIS*	R/W1C	0	Comparator B Up Interrupt Status and clear 0: No interrupt or the interrupt is masked. 1: Comparator B Up Interrupt has been signaled.
3	*CMPADIS*	R/W1C	0	Comparator A Down Interrupt Status and clear 0: No interrupt or the interrupt is masked. 1: Comparator A Down Interrupt has been signaled.
2	*CMPAUIS*	R/W1C	0	Comparator A Up Interrupt Status and clear 0: No interrupt or the interrupt is masked. 1: Comparator A Up Interrupt has been signaled.
1	*CNTLIS*	R/W1C	0	Counter = Load Interrupt Status and clear 0: No Interrupt or the interrupt has been masked. 1: The counter has matched the load value and the interrupt has been signaled
0	*CNTZIS*	R/W1C	0	Counter = Zero Interrupt Status and clear 0: No Interrupt or the interrupt has been masked. 1: The counter has matched zero and the interrupt has been signaled.

4.8.4.18 PWMN_LOAD - PWM0/1/2 LOAD VALUE

These registers contain the load value for PWM counter. Depending on the counter mode, either this value is loaded into the counter when it reaches zero or use this value as limit value for up counting after which the counter is cleared to zero or count down towards zero. If the load update mode is immediate then this value is used the next time the counter reaches zero. Otherwise, if the load update mode is synchronized then the next time the counter reaches zero and a synchronization signal is active then this value will be used.

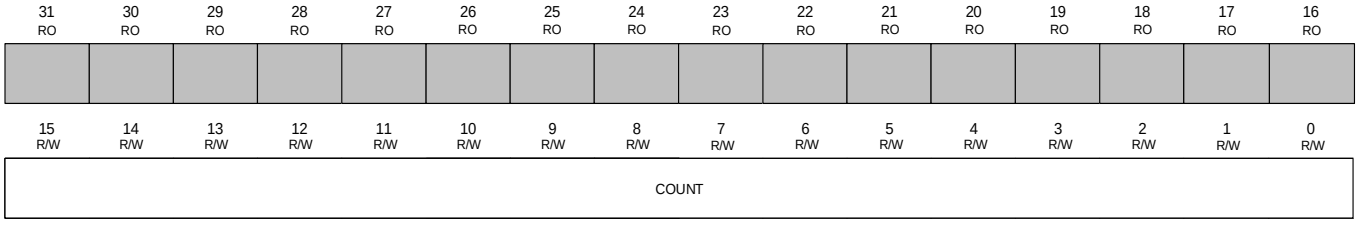


Offset:
PWM0_LOAD: 0x0058
PWM1_LOAD: 0x0098
PWM2_LOAD: 0x00D8

Bits	Name	Type	Reset	Description
31:20	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	SCALE	R/W	0x0	Counter Scale Control from 0 to 15
15:0	LOAD	R/W	0x0	Counter Load Value.

4.8.4.19 PWMN_COUNT - PWM0/1/2 COUNTER

These registers contain the current value of the PWM counter. When the count value equal of the load register value, it will generate a pulse; use to drive in generation of PWM signal (via **PWMn_GENA** / **PWMn_GENB** register) or drive an interrupt of ADC trigger (through **PWMn_INTEN** register). A pulse is generated with the same capabilities when this value is zero.

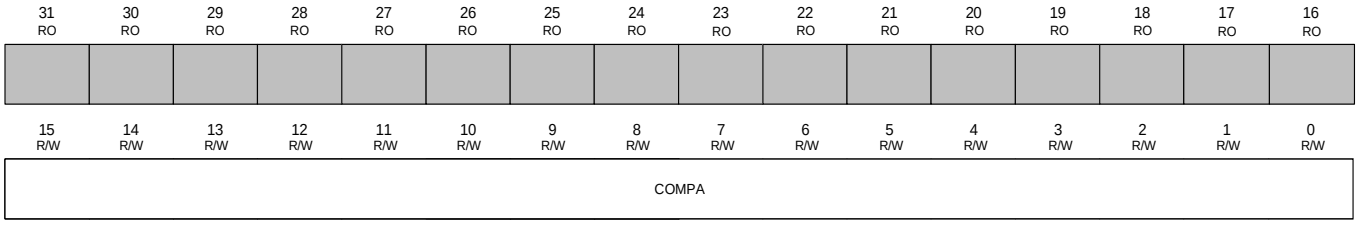


Offset:
PWM0_COUNT: 0x005C
PWM1_COUNT: 0x009C
PWM2_COUNT: 0x00DC

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	*COUNT*	R/W	0x0	Counter Value The current value of the counter.

4.8.4.20 PWMN_COMPA - PWM0/1/2 COMPARATOR A

These registers contain the values to be compared with the counter. When this value matches the counter, a pulse is output which can drive; a) the generation of PWM signal, b) interrupt signal, c) ADC trigger signal. If the value is greater than the **PWMn_LOAD** register then no pulse is ever output. If the load update mode is immediate then this value is used the next time the counter reaches zero. Otherwise, if the load update mode is synchronized then the next time the counter reaches zero and a synchronization signal is active then this value will be used.

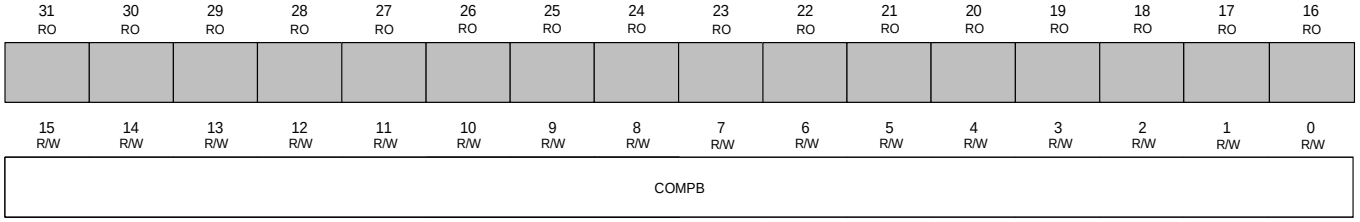


Offset:
PWM0_COMPA: 0x0060
PWM1_COMPA: 0x00A0
PWM2_COMPA: 0x00E0

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	*COMPA*	R/W	0x0	Comparator A value The value to be compared against the counter.

4.8.4.21 PWMN_COMPB - PWM0/1/2 COMPARATOR B

This register perform the same function as in PWMn_COMPA but only for Comparator B.

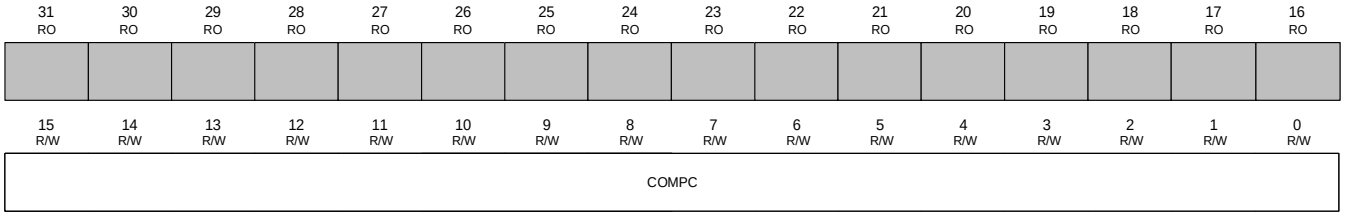


Offset:
PWM0_COMPB: 0x0064
PWM1_COMPB: 0x00A4
PWM2_COMPB: 0x00E4

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	*COMPB*	R/W	0x0	Comparator B value The value to be compared against the counter.

4.8.4.22 PWMN_COMPC - PWM0/1/2 COMPARATOR C

This register perform the same function as in PWMn_COMPA but only for Comparator B.

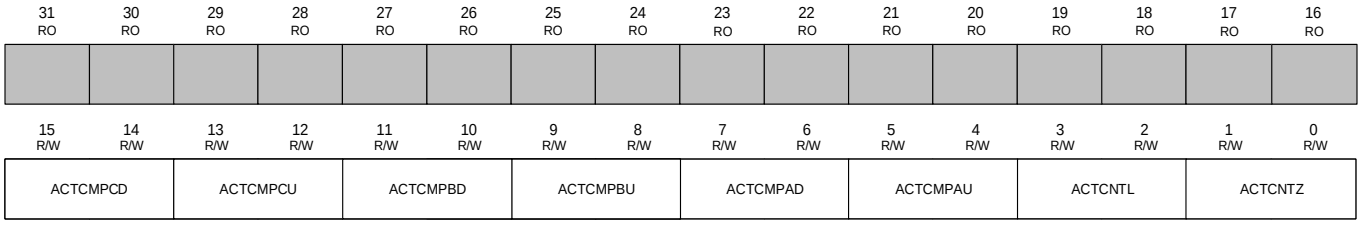


Offset:
PWM0_COMPC: 0x0068
PWM1_COMPC: 0x00A8
PWM2_COMPC: 0x00E8

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	*COMPC*	R/W	0x0	Comparator C value The value to be compared against the counter.

4.8.4.23 PWMN_GENA - PWM 0/1/2 GENERATOR CONTROL A

These registers control the PWMn_A signal generation based on the load and zero output pulse events from the counter, as well as the compare A and compare B pulse events from the comparators. For counters in down counting mode or up counting mode only four events occur. For counters in up/down counting mode all six events occur. These events allow greater flexibility in the generation of PWM signal. If a zero or load event coincides with a compare A or compare B event, the zero or load action is taken ignoring the others. If a compare A event coincides with a compare B event, the compare A action is taken ignoring compare B action.

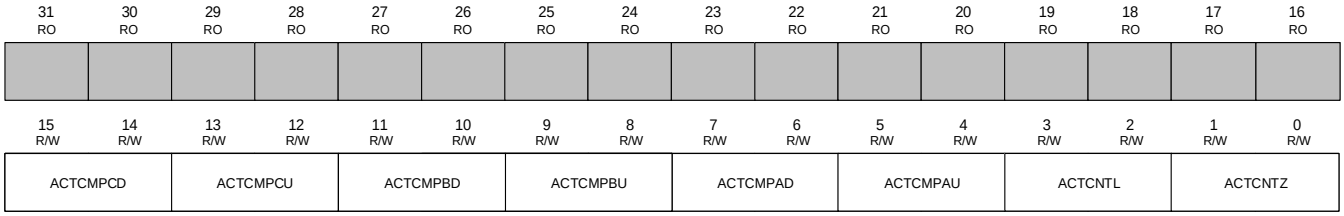


Offset:
PWM0_GENA: 0x006C
PWM1_GENA: 0x00AC
PWM2_GENA: 0x00EC

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.
15:14	*ACTCMPCD*	R/W	0x0	Action for comparator C Down The action to be taken when the counter matches comparator C while counting down. Same action table as in *ACTCNTZ*.
13:12	*ACTCMPCU*	R/W	0x0	Action for comparator C Up The action to be taken when the counter matches comparator C while counting up. Same action table as in *ACTCNTZ*.
11:10	*ACTCMPBD*	R/W	0x0	Action for comparator B Down The action to be taken when the counter matches comparator B while counting down. Same action table as in *ACTCNTZ*.
9:8	*ACTCMPBU*	R/W	0x0	Action for comparator B Up The action to be taken when the counter matches comparator B while counting up. Same action table as in *ACTCNTZ*.
7:6	*ACTCMPAD*	R/W	0x0	Action for comparator A Down The action to be taken when the counter matches comparator A while counting down. Same action table as in *ACTCNTZ*.
5:4	*ACTCMPAU*	R/W	0x0	Action for comparator A Up The action to be taken when the counter matches comparator A while counting up. Same action table as in *ACTCNTZ*.
3:2	*ACTCNTL*	R/W	0x0	Action for Counter = Load The action to be taken when the counter matches the load value. Same action table as in *ACTCNTZ*.
1:0	*ACTCNTZ*	R/W	0x0	Action for Counter = Zero The action to be taken when the counter is zero. Action Table: 00: Do Nothing 01: Invert the output signal 10: Set the output signal to 0. 11: Set the output signal to 1.

4.8.4.24 PWMN_GENB - PWM0/1/2 GENERATOR CONTROL B

This register perform the same function as in PWMn_GENA but only for Generator B.

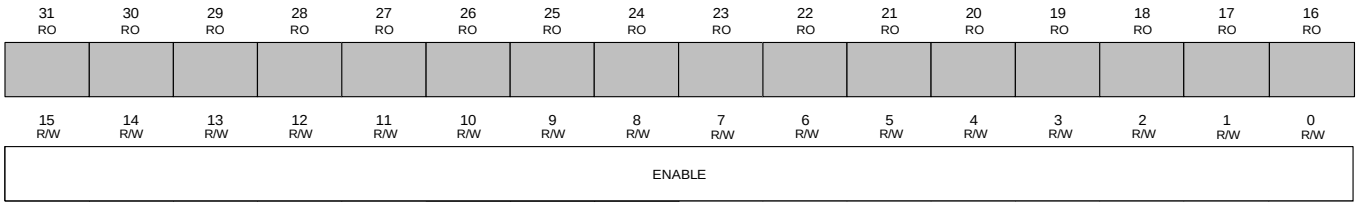


Offset:
PWM0_GENB: 0x0070
PWM1_GENB: 0x00B0
PWM2_GENB: 0x00F0

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.
15:14	ACTCMPCD	R/W	0x0	Action for comparator C Down The action to be taken when the counter matches comparator C while counting down. Same action table as in *ACTCNTZ*.
13:12	ACTCMPCU	R/W	0x0	Action for comparator C Up The action to be taken when the counter matches comparator C while counting up. Same action table as in *ACTCNTZ*.
11:10	ACTCMPBD	R/W	0x0	Action for comparator B Down The action to be taken when the counter matches comparator B while counting down. Same action table as in *ACTCNTZ*.
9:8	ACTCMPBU	R/W	0x0	Action for comparator B Up The action to be taken when the counter matches comparator B while counting up. Same action table as in *ACTCNTZ*.
7:6	ACTCMPAD	R/W	0x0	Action for comparator A Down The action to be taken when the counter matches comparator A while counting down. Same action table as in *ACTCNTZ*.
5:4	ACTCMPAU	R/W	0x0	Action for comparator A Up The action to be taken when the counter matches comparator A while counting up. Same action table as in *ACTCNTZ*.
3:2	ACTCNTL	R/W	0x0	Action for Counter = Load The action to be taken when the counter matches the load value. Same action table as in *ACTCNTZ*.
1:0	ACTCNTZ	R/W	0x0	Action for Counter = Zero The action to be taken when the counter is zero. Action Table: 00: Do Nothing 01: Invert the output signal 10: Set the output signal to 0. 11: Set the output signal to 1.

4.8.4.25 PWMN_DBCTL - PWM0/1/2 DEAD BAND CONTROL

This register controls the dead-band generator which produces PWMn_A and PWMn_B signals based on the pwmn_a and pwmn_b signals. When disabled, the pwm_a' is equal to pwm0_a and pwm0_b' is equal to pwm0_b. When enabled and inverting the resulting waveform, the pwm0_b signal is ignored. The pwm0_a' signal is generated by delaying the rising edge of the pwm_a by the value in the **PWMn_DBRISE** register and the pwm0_b' signal is generated by delaying the falling edge of the pwm0_a signal by the value in the **PWMn_DBFALL** register. The output control block outputs the pwm0_a' signal on the PWM0_A signal and the pwm0_b' signal on the PWM0_B signal and other PWM output signal are produced based on the same manner.

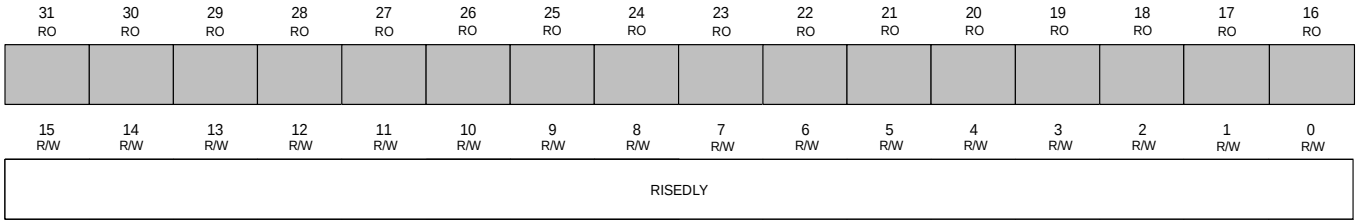


Offset:
PWM0_DBCTL: 0x0074
PWM1_DBCTL: 0x00B4
PWM2_DBCTL: 0x00F4

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	*ENABLE*	R/W	0x0	Dead-Band Generator Enable 0: Passes the PWM signal directly. 1: Dead-band generator inserts dead bands into the output signals.

4.8.4.26 PWMN_DBRISE - PWM0/1/2 DEAD BAND RISE DELAY

This register contains the number of clock ticks to delay the rising edge of the pwmn_a signal when generating the pwmn_a. This register is ignored if the dead-band generator is disabled. If the value of this register is higher than the width of a High pulse on the input PWM signal, the rising-edge delay consumes the entire High time of the signal resulting in no High time on the output. It is important to ensure that the input High time is always longer than the rising-edge delay.

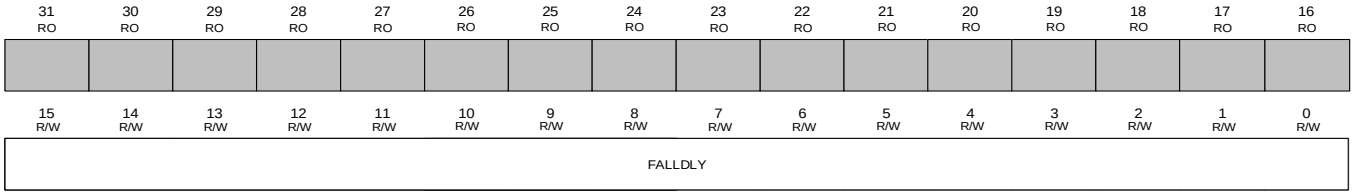


Offset:
PWM0_DBRISE: 0x0078
PWM1_DBRISE: 0x00B8
PWM2_DBRISE: 0x00F8

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	*RISEDLY*	R/W	0x0	Dead-Band Rise Delay The number of clock ticks to delay the rising edge.

4.8.4.27 PWMN_DBFALL - PWM0/1/2 DEAD BAND FALL DELAY

This register contains the number of clock ticks to delay the falling edge of the pwmn_a signal when generating the pwmn_b'. This register is ignored if the dead-band generator is disabled. If the value of this register is higher than the width of a Low pulse on the input PWM signal, the falling-edge delay consumes the entire Low time of the signal resulting in no Low time on the output. It is important to ensure that the input Low time is always longer than the falling-edge delay.



Offset:
PWM0_DBFALL: 0x007C
PWM1_DBFALL: 0x00BC
PWM2_DBFALL: 0x00FC

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	*FALLDLY*	R/W	0x0	Dead-Band Fall Delay The number of clock ticks to delay the falling edge.

4.9 ANALOG TO DIGITAL CONVERTER (ADC)

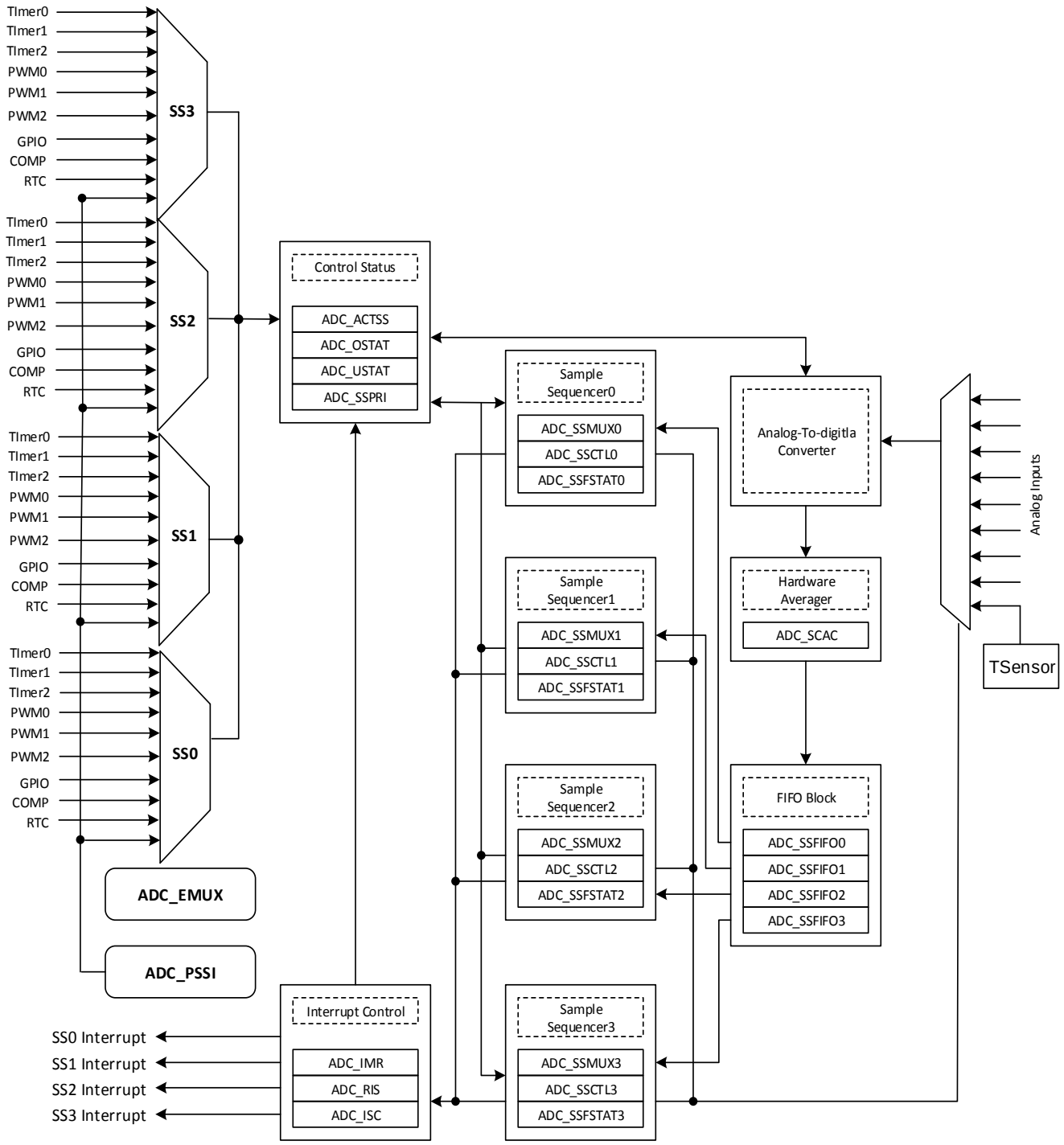
An analog-to-digital converter (ADC) is a peripheral that converts a continuous analog voltage to a discrete digital number. The PT32U301 contains one 12-bit successive approximation analog-to-digital converter (SAR A/D converter) with 8 external input channels, plus an internal temperature sensor. Each 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
  - 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

4.9.1 BLOCK DIAGRAM

Figure 4.9-1: ADC Block Diagram



4.9.2 FUNCTIONAL DESCRIPTION

The PT32U301 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.

4.9.2.1 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 4.9-1: 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.**

4.9.2.2 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)**.

4.9.2.3 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.

4.9.2.4 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, PWM and 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.

4.9.2.5 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 PT32U301, 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.

4.9.2.6 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.

4.9.3 INITIALIZATION AND CONFIGURATION

4.9.3.1 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 to 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

4.9.3.2 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.

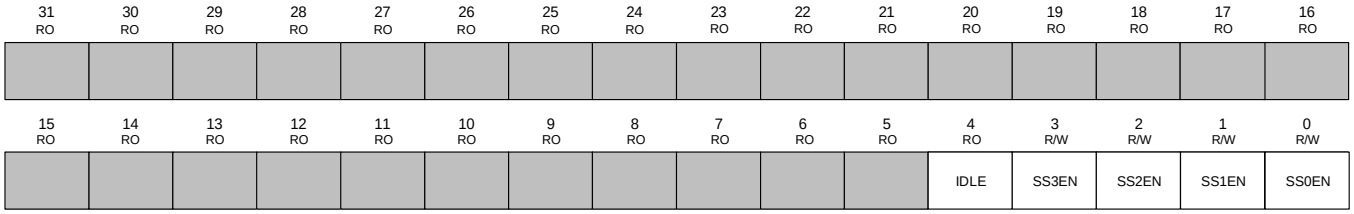
4.9.4 ADC REGISTER MAP

Base Address: 0x4802_0000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	ADC_ACTSS	R/W	0x0000_0010	ADC Active Sample Sequencer	160
0x0004	ADC_IER	WO	0x0000_0000	ADC Interrupt Enable	161
0x0008	ADC_IDR	WO	0x0000_0000	ADC Interrupt Disable	161
0x000C	ADC_IMR	RO	0x0000_0000	ADC Interrupt Mask Status	162
0x0010	ADC_RIS	RO	0x0000_0000	ADC Raw Interrupt Status	163
0x0014	ADC_ISC	R/W1C	0x0000_0000	ADC Interrupt Status and Clear	164
0x0018	ADC_OSTAT	R/W1C	0x0000_0000	ADC Overflow Status	165
0x001C	ADC_EMUX	R/W	0x0000_0000	ADC Event Multiplexer Select	166
0x0020	ADC_USTAT	R/W1C	0x0000_0000	ADC Underflow Status	167
0x0028	ADC_SSPRI	R/W	0x0000_3210	ADC Sample Sequencer Priority	168
0x002C	ADC_INI	R/W	0x0000_0000	ADC Initial Control	169
0x0030	ADC_PSSI	R/W	0x0000_0000	ADC Processor Sample Sequence Initiate	170
0x0034	ADC_AVGC	R/W	0x0000_0000	ADC Average Control	171
0x0038	ADC_GAINC	R/W	0x0000_0000	ADC PGA Gain Control	173
0x003C	ADC_PGAC	R/W	0x0000_0000	ADC PGA Mode Control	175
0x0040	ADC_TMPC	R/W	0x0000_0000	ADC Temperature Control	176
0x0044	ADC_FRF	W1C	0x0000_0000	ADC FIFO Refresh	176
0x0048	ADC_WAIT	R/W	0x0000_0000	ADC Wait Counter Register	177
0x0050	ADC_SSMUX0	R/W	0x0000_0000	ADC Sample Sequence Input Multiplexer Select 0	177
0x0054	ADC_SSCTL0	R/W	0x2000_0000	ADC Sample Sequence Control 0	179
0x0058	ADC_SSFIFO0	RO	0x0000_0000	ADC Sample Sequence Result FIFO 0	181
0x005C	ADC_SSFSTAT0	RO	0x0000_0100	ADC Sample Sequence FIFO 0 Status	182
0x0070	ADC_SSMUX1	R/W	0x0000_0000	ADC Sample Sequence Input Multiplexer Select 1	183
0x0074	ADC_SSCTL1	R/W	0x0000_2000	ADC Sample Sequence Control 1	184
0x0078	ADC_SSFIFO1	RO	0x0000_0000	ADC Sample Sequence Result FIFO 1	181
0x007C	ADC_SSFSTAT1	RO	0x0000_0100	ADC Sample Sequence FIFO 1 Status	182
0x0090	ADC_SSMUX2	R/W	0x0000_0000	ADC Sample Sequence Input Multiplexer Select 2	186
0x0094	ADC_SSCTL2	R/W	0x0000_2000	ADC Sample Sequence Control 2	184
0x0098	ADC_SSFIFO2	RO	0x0000_0000	ADC Sample Sequence Result FIFO 2	181
0x009C	ADC_SSFSTAT2	RO	0x0000_0100	ADC Sample Sequence FIFO 2 Status	182
0x00B0	ADC_SSMUX3	R/W	0x0000_0000	ADC Sample Sequence Input Multiplexer Select 3	186
0x00B4	ADC_SSCTL3	R/W	0x0000_0002	ADC Sample Sequence Control 3	186
0x00B8	ADC_SSFIFO3	RO	0x0000_0000	ADC Sample Sequence Result FIFO 3	181
0x00BC	ADC_SSFSTAT3	RO	0x0000_0100	ADC Sample Sequence FIFO 3 Status	182

4.9.4.1 ADC_ACTSS - ADC ACTIVE SAMPLE SEQUENCER

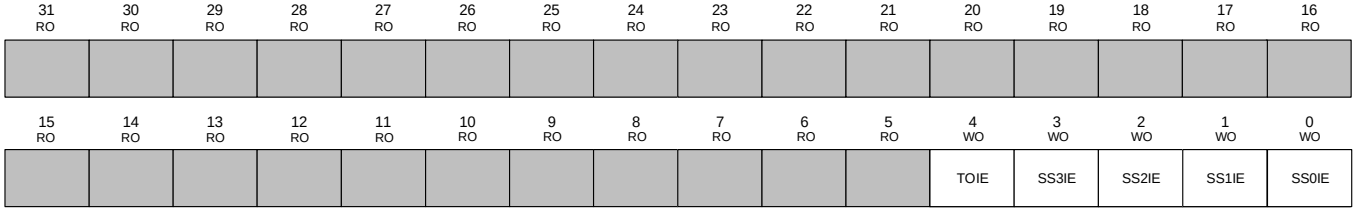
This register controls the activation of the sample sequencers. Each sample sequencer can be enabled or disabled independently.



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0	Software should not 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

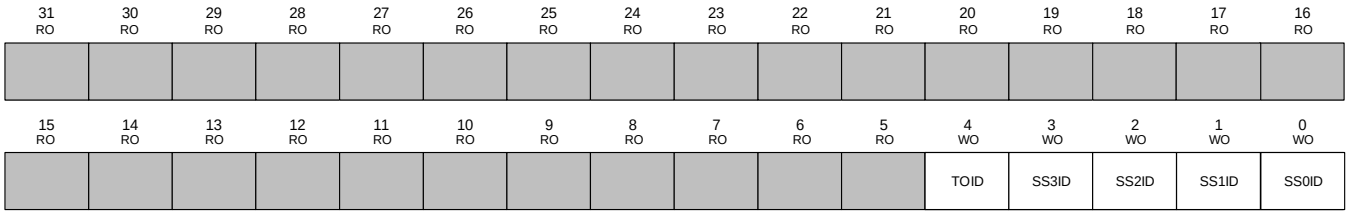
4.9.4.2 ADC_IER - ADC INTERRUPT ENABLE



Offset: 0x0004

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	*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

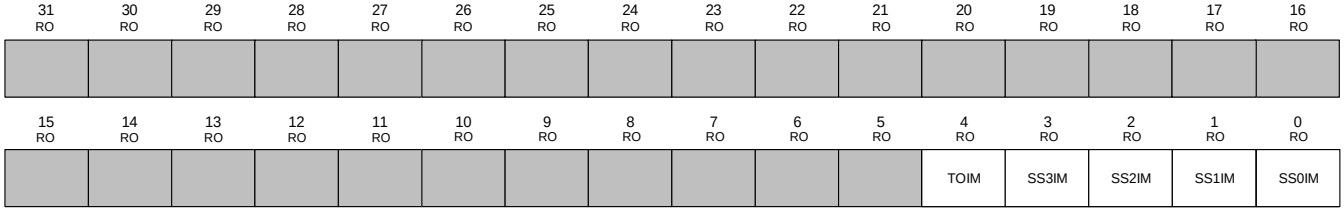
4.9.4.3 ADC_IDR - ADC INTERRUPT DISABLE



Offset: 0x0008

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	*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

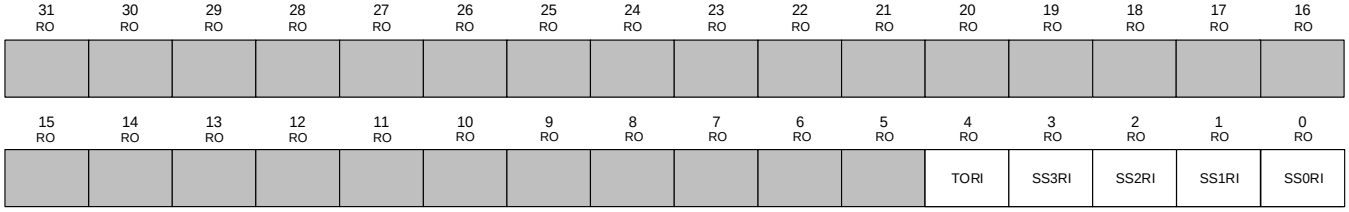
4.9.4.4 ADC_IMR - ADC INTERRUPT MASK STATUS



Offset: 0x000C

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	*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.

4.9.4.5 ADC_RIS - ADC RAW INTERRUPT STATUS

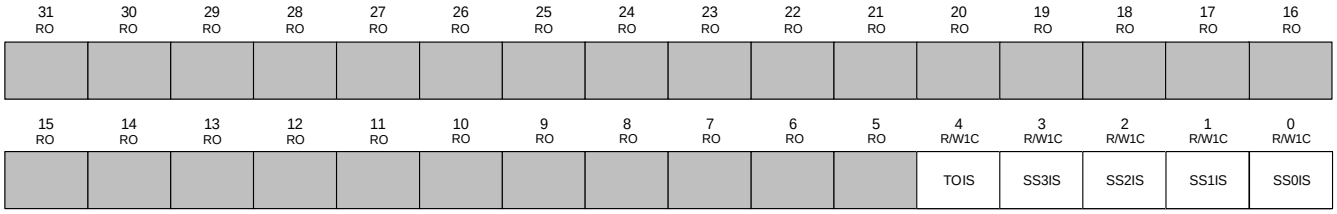


Offset: 0x0010

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.

4.9.4.6 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.

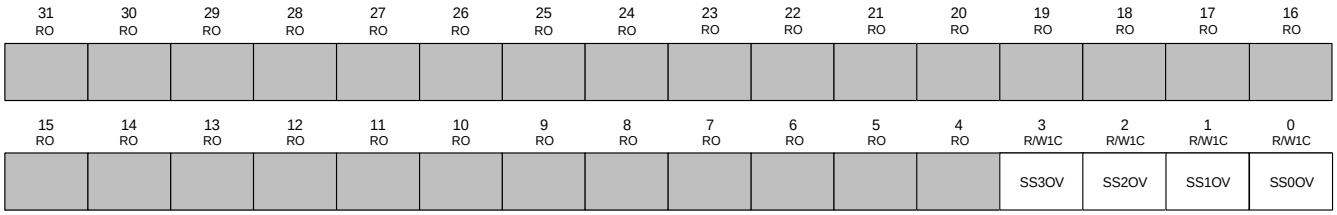


Offset: 0x0014

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0	Software should not 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.

4.9.4.7 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.

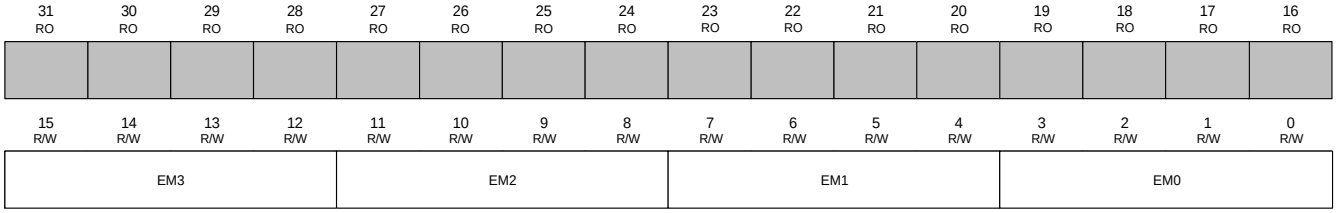


Offset: 0x0018

Bits	Name	Type	Reset	Description
31:4	reserved	RO	0x0	Software should not 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.

4.9.4.8 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.

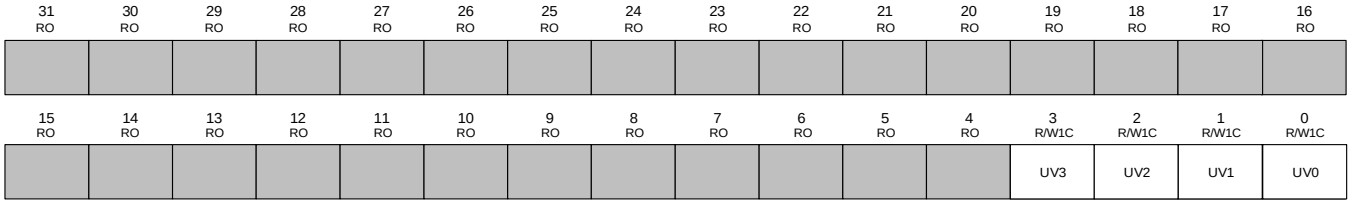


Offset: 0x001C

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: reserved 0x8: PWM0 0x9: PWM1 0xA: PWM2 0xB: reserved 0xC: Analog Comparator 0 0xD: Analog Comparator 1 0xE: Analog Comparator 2 0xF: Analog Comparator 3

4.9.4.9 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.

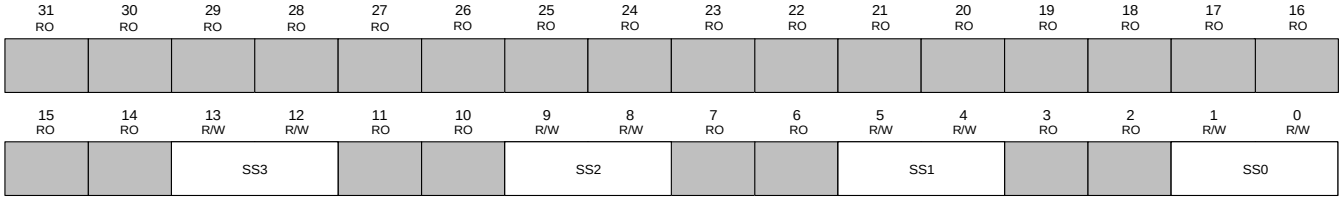


Offset: 0x0020

Bits	Name	Type	Reset	Description
31:4	*reserved*	RO	0x0	Software should not 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 overflowed. 1: SS2 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
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

4.9.4.10 ADC_SSPRI - ADC SAMPLE SEQUENCER PRIORITY

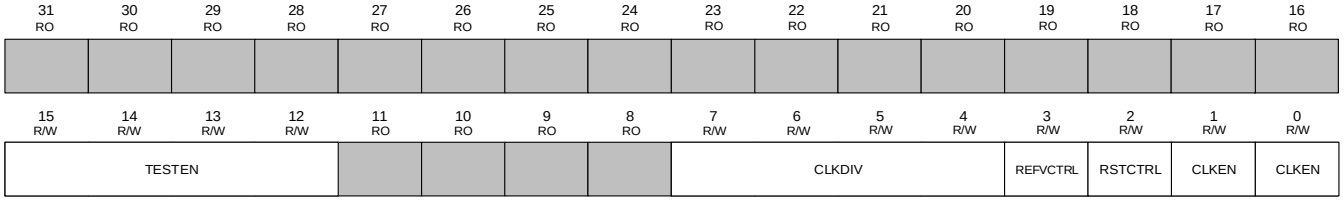
This register sets the priority for each of the sample sequencers. Out of reset, Sequencer 0 has the highest priority, and Sequencer 3 has the lowest priority. When reconfiguring sequence priorities, each sequence must have a unique priority for the ADC to operate properly.



Offset: 0x0028

Bits	Name	Type	Reset	Description
31:14	*reserved*	RO	0x0	Software should not 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

4.9.4.11 ADC_INI - ADC INITIAL CONTROL

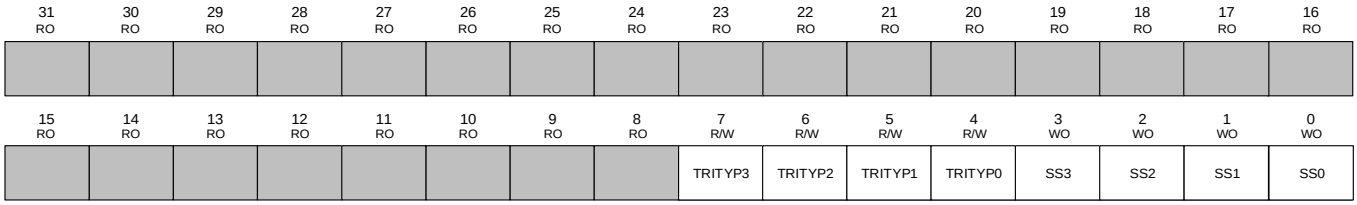


Offset: 0x002C

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	*TESTEN*	R/W	0x0	ADC macro Test mode Enable 0XED: 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

4.9.4.12 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.

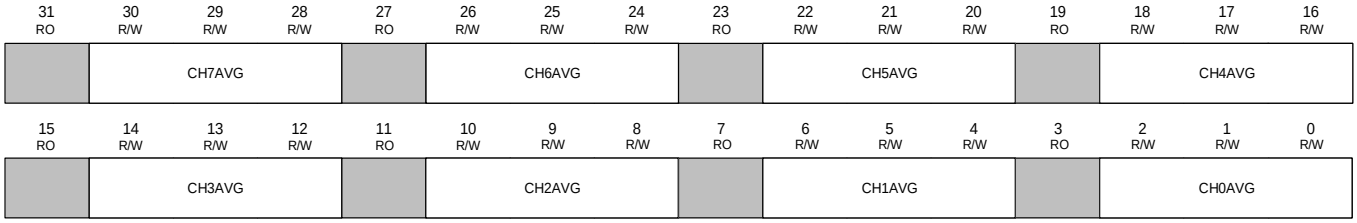


Offset: 0x0030

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not 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**.

4.9.4.13 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	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	*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

4.9.4.14 ADC_GAINC - 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 differnet PGA mode which is controller by **ADC_PGAC** register.

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

Figure 4.9-2: PGA Block Diagram

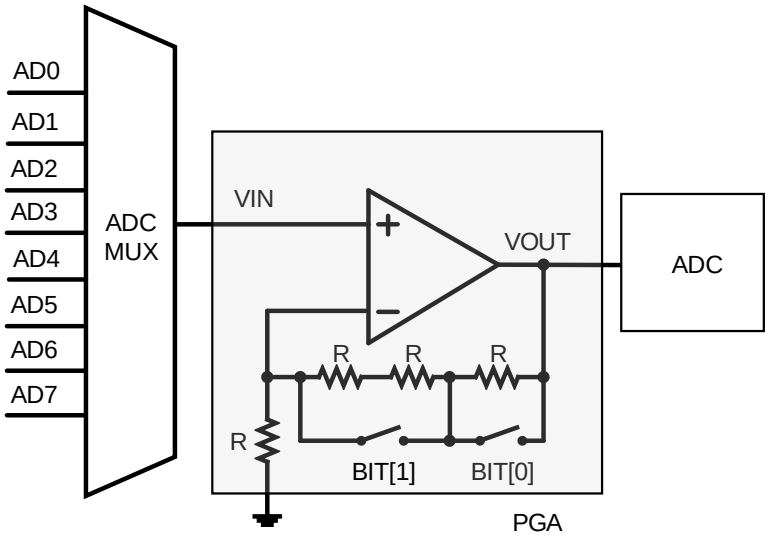
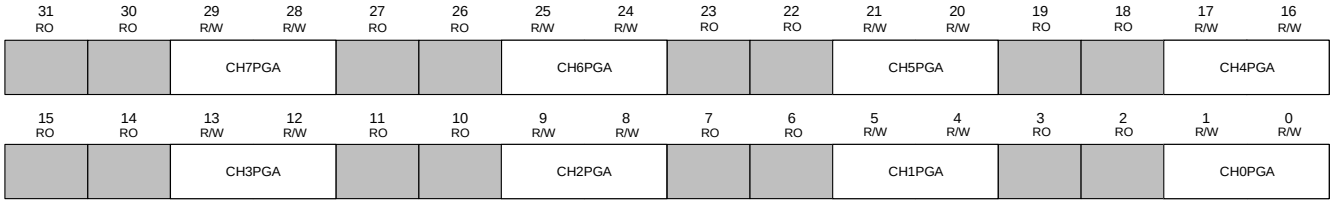


Table 4.9-2: Gain Amplification under different channel value

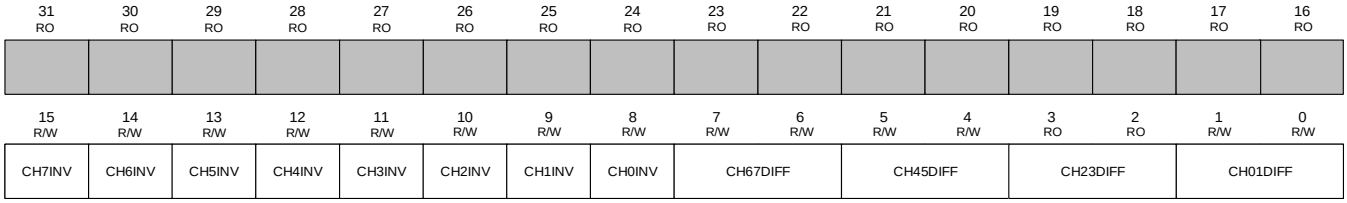
Channel Value (BIT[0],BIT[1])	Gain amplification
00	x1 (bypass)
01	x2
10	x3
11	x5



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

4.9.4.15 ADC_PGAC - ADC PGA MODE CONTROL

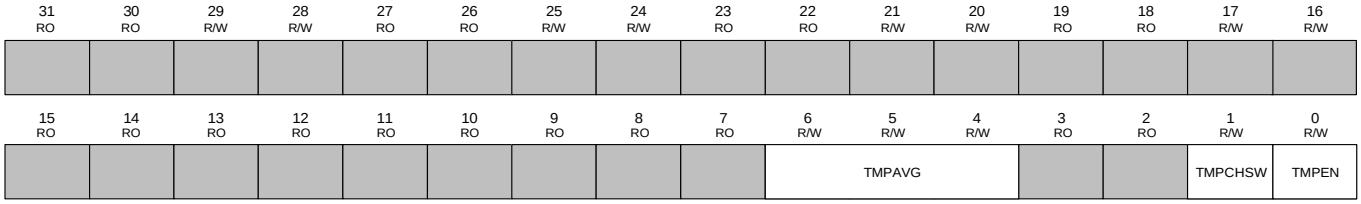


Offset: 0x003C

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	*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.
7:6	*CH67DIFF*	R/W	0x0	Channel 7 and 6 Differential Control 0: No differential control between channel 7 and 6 1: Differential control is activated.
5:4	*CH45DIFF*	R/W	0x0	Channel 5 and 4 Differential Control 0: No differential control between channel 5 and 4 1: Differential control is activated.
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.

4.9.4.16 ADC_TMC - ADC TEMPERATURE CONTROL

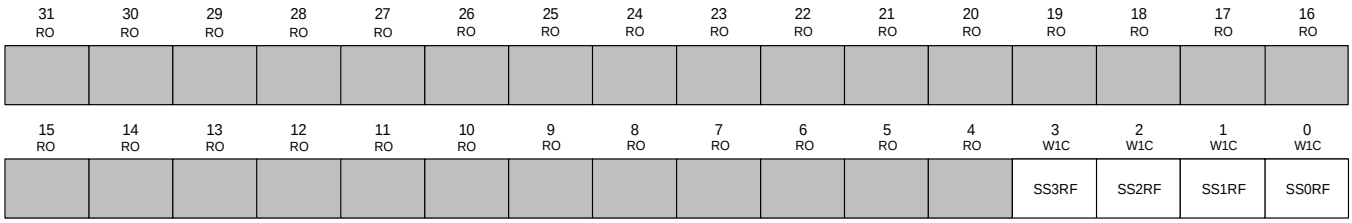
This register controls ADC temperature detection control.



Offset: 0x0040

Bits	Name	Type	Reset	Description
31:7	*reserved*	RO	0x0	Software should not 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	0x0	Software should not 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.

4.9.4.17 ADC_FRF - ADC FIFO REFRESH.

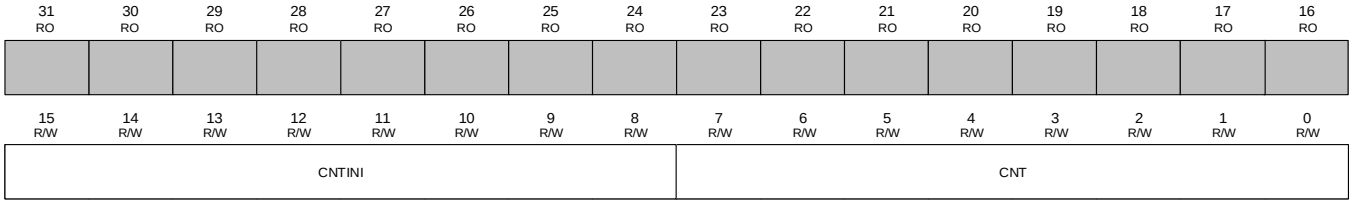


Offset: 0x0044

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	*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

4.9.4.18 ADC_WAIT - ADC WAIT COUNTER

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

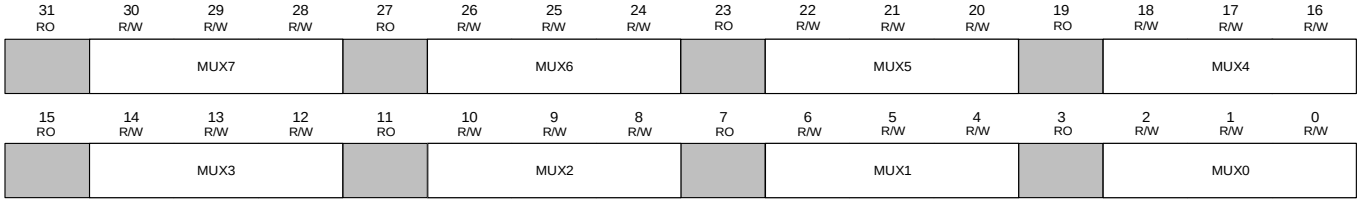


Offset: 0x0044

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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.

4.9.4.19 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.



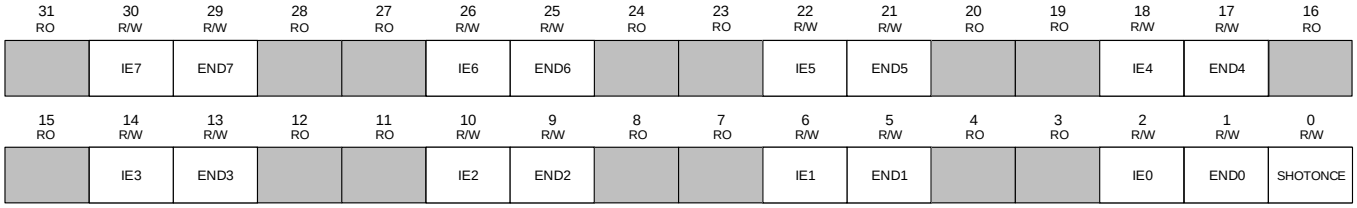
Offset: 0x0050

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	8^{TH} 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	7^{TH} 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	6^{TH} 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.

Bits	Name	Type	Reset	Description
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	5^{TH} 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	4^{TH} 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.
10:8	*MUX2*	R/W	0x0	3^{TH} 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	2^{TH} 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	1^{TH} 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.

4.9.4.20 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.



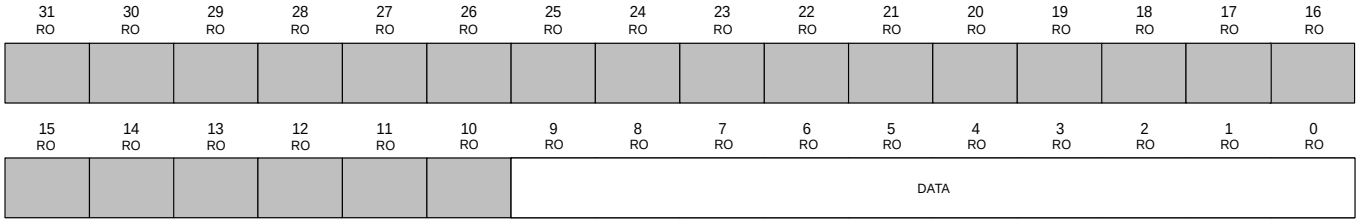
Offset: 0x0054

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.

Bits	Name	Type	Reset	Description
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. .
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.

4.9.4.21 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.

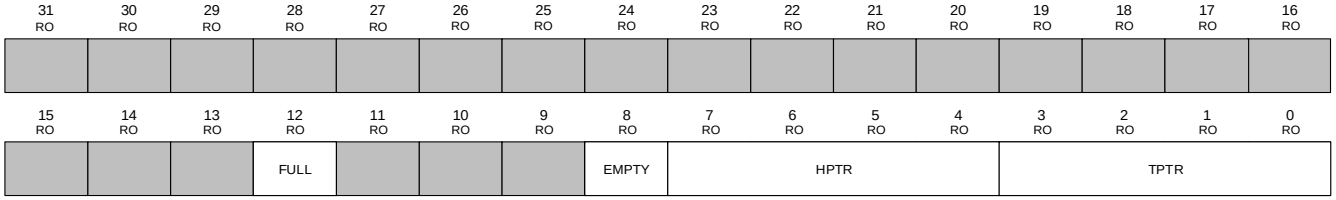


Offset:
ADC_SSFIFO0: 0x0058
ADC_SSFIFO1: 0x0078
ADC_SSFIFO2: 0x0098
ADC_SSFIFO3: 0x00B8

Bits	Name	Type	Reset	Description
31:10	*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.
11:0	*DATA*	RO	0x0	Conversion Result Data

4.9.4.22 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.

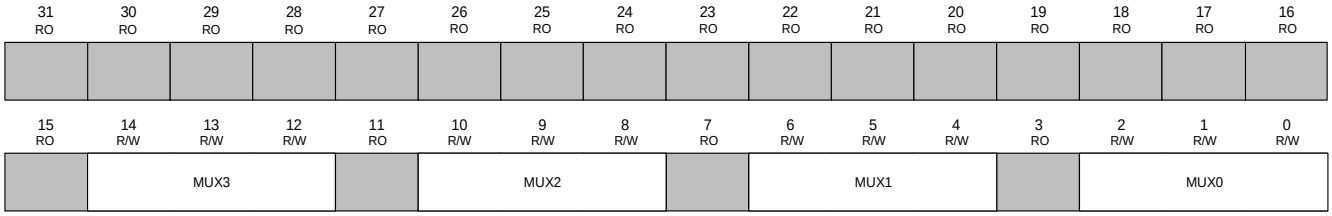


Offset:
ADC_SSFIFO0: 0x005C
ADC_SSFIFO1: 0x007C
ADC_SSFIFO2: 0x009C
ADC_SSFIFO3: 0x00BC

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

4.9.4.23 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.

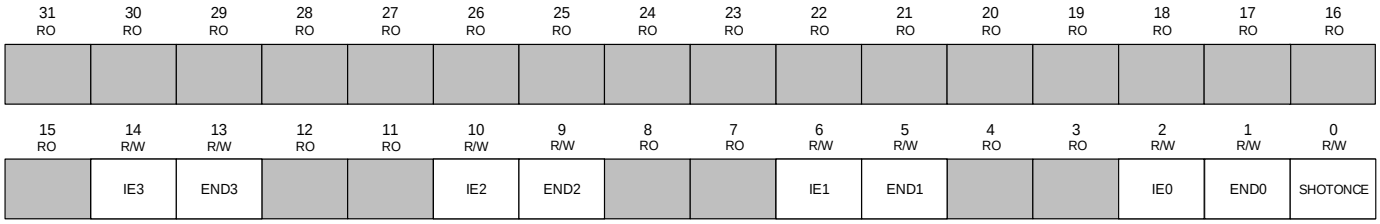


Offset:
ADC_SSMUX1: 0x0070
ADC_SSMUX2: 0x0090

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.

4.9.4.24 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.



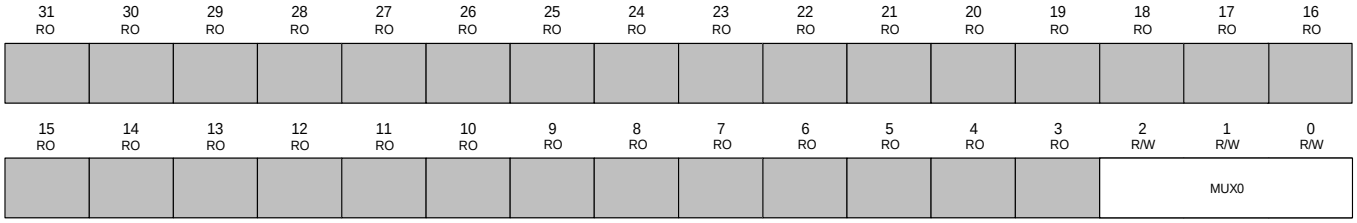
Offset:
ADC_SSCTL1: 0x0074
ADC_SSCTL2: 0x0094

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.

4.9.4.25 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.

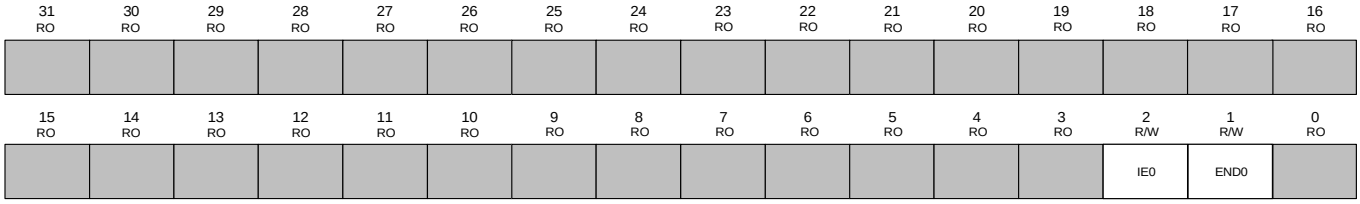


Offset: 0x00B0

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.

4.9.4.26 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.

4.10 GENERAL PURPOSE TIMERS (GPT)

The PT32U301 contains three 16/32-bit General-Purpose Timer Module (GPTM0, GPTM1, and GPTM2) blocks. Each 16/32-bit GPTM block provides two 16- bit timer/counters (referred to as Timer-A and Timer-B) that can be configured to operate independently as timers or event counters, or concatenated to operate as one 32-bit timer.

In addition, timers can be used to trigger analog-to-digital (ADC) conversions. The ADC trigger signals from all of the general-purpose timers logically ORed together before reaching the ADC module, so only one timer should be used to trigger ADC events.

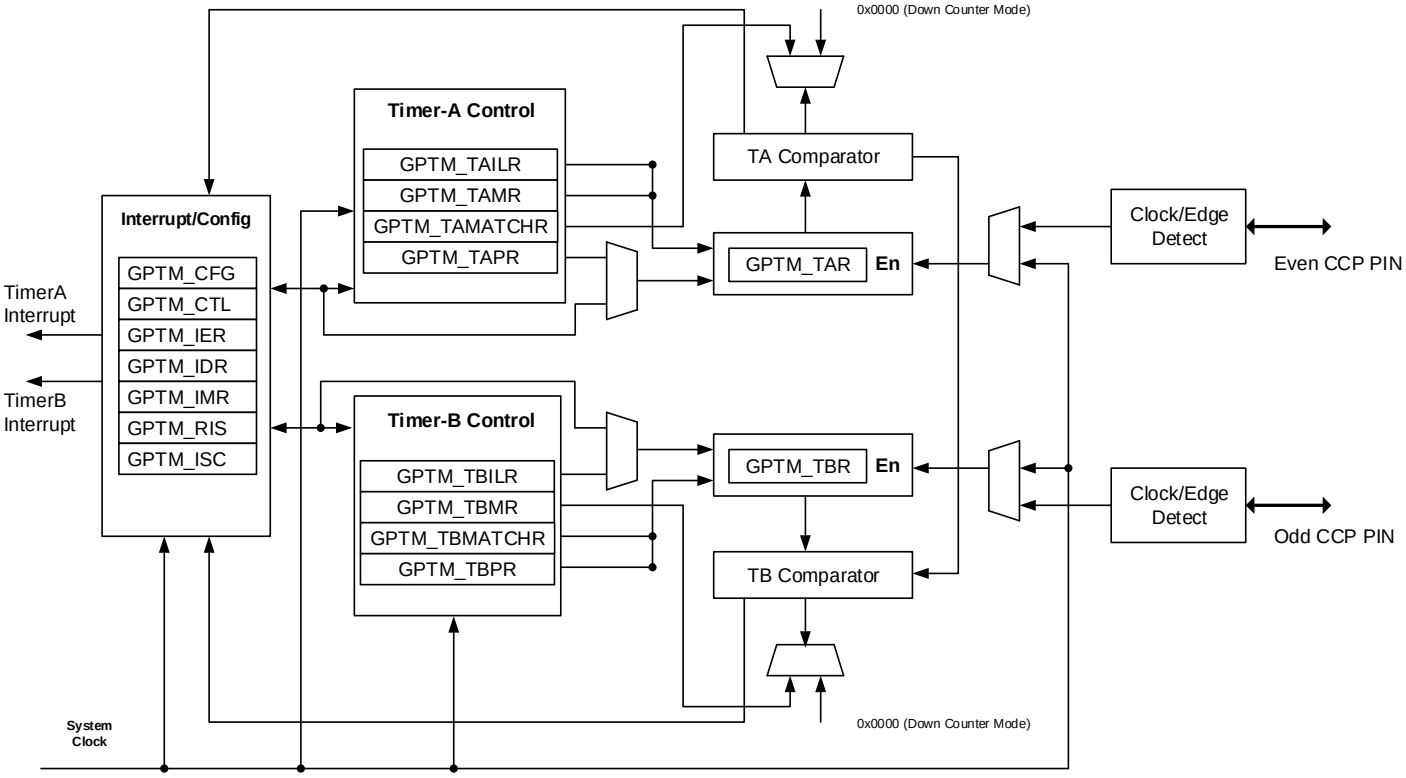
This GPTM is one timing resource available on the PT32U301. Other timer resources include the System Timer (SysTick).

The General-Purpose Timer Modules (GPTM) can be configured to operate independently providing following features:

- 16/32-bit Timer modes
  - 16 bit general-purpose timer function with a 16-bit prescaler.
  - 16-bit input PWM edge detection mode
  - 16-bit input edge count capture or input edge time capture with a 16-bit prescaler.
  - 16-bit PWM mode with a 16-bit prescaler and software-programmable output inversion of the PWM signal
  - 16- or 32- bit programmable one-shot timer.
  - 16- or 32- bit programmable periodic timer.
- ADC event trigger
- User-enabled stalling when the controller asserts CPU Halt flag during debug.

4.10.1 BLOCK DIAGRAM

Figure 4.10-1: GPTM Module Block Diagram



4.10.2 FUNCTIONAL DESCRIPTION

The main components of each GPTM block are two free-running 16-bit up/down counters (referred to as Timer-A and Timer-B), two 16-bit match registers, two prescaler match registers, and two 16-bit load/initialization registers and their associated control functions. The exact functionality of each GPTM is controlled by software and configured through the register interface. Note that the prescaler can only be used when Timer-A and Timer-B are used individually.

4.10.2.1 GPTM RESET CONDITIONS

When the GPTM module resets, the module is in an inactive state and all control registers are cleared and in its default state. Counter A and Counter B along with their corresponding load registers are initialized to 0xFFFF_FFFF. Prescaler counter is initialized to 0xFFFF.

4.10.2.2 GPTM OPERATING MODES

The available modes for each GPTM block are shown in the following table.

Table 4.10-1: Available Operation Mode

Mode	Timer Use	Counter Size
One-shot	Individual	16-bit
	Concatenated	32-bit
Periodic	Individual	16-bit
	Concatenated	32-bit
Edge Count	Individual	16-bit
Edge Time	Individual	16-bit
PWM Edge	Individual	16-bit
PWM	Individual	16-bit

Note:

1. The prescaler is only available when the timers are used individually
2. All timer counts down.

This section describes the operation of the various timer modes. When using Timer A and Timer B in concatenated mode, only the Timer A control and status bits must be used; there is no need to use Timer B control and status bit. By writing the *CFG* bit in the GPTM Configuration register (**GPTM_CFG**) to configure the GPTM for different operating modes. Timer-A and Timer-B have identical modes, so a single description is given using an “n” to reference both.

One-Shot/Periodic Timer Mode

The selection of one-shot or periodic mode is determined by value of *MODE* field of the GPTM Timer n Mode register (**GPTM_TnMR**).

The optional prescaler is loaded into the GPTM Timer n Prescale **(GPTM_TnPR)** register. In 32-bit GPTM, an optional 16-bit prescaler can effectively extends the counting range of the timer to 48 bits. In 16-bit GPTM an optional 16-bit prescaler can effectively extends the counting range of the timer to 32 bits.

When software writes the *TnEN* bit in the GPTM Control register (**GPTM_CTL**), the timer begins counting down from its preloaded value and when it reaches 0x0, the timer reloaded its start value from *TnIRL*/*TnIRH* from GPTM Timer n Interval Load register (**GPTM_TnILR**) register on the next cycle. If configured to be a one-shot timer, the timer stops counting and clears the **GPTM_CTL**.*TnEN* bit. If configured as a periodic timer, the timer starts counting again on the next cycle.

In addition to reloading the count value, the GPTM can generate interrupts when it reached 0x0. The GPTM sets the *TnTOR*I bit in the GPTM Raw Interrupt Status **(GPTM_RIS)** register, and holds it until it is cleared by writing the GPTM Interrupt Status and Clear **(GPTM_ISC)** register.

If software updates the **GPTM_TnILR** or the **GPTM_TnPR** register while the counter is running, the counter loads the new value on the next clock cycle and continues counting from the new value. If the *TnSTALL* bit in the **GPTM_CTL** register is set, the timer freezes counting while the processor is halted by the debugger. The timer resumes counting when the processor resumes execution.

The following tables shows a variety of configurations for a 16-bit or 32-bit free running timer while using the prescaler. All values assume a 16-MHz clock with Tc=62.5 ns (clock period).

Table 4.10-2: 16 bit Timer with Prescaler Configuration

Prescale	Clock	Max Time	Units
00000000	1	4.096	ms
00000001	2	8.192	ms
00000010	3	12.288	ms
--	--	--	--
11111101	254	1040.384	ms
11111110	255	1044.48	ms
11111111	256	1048.576	ms

Table 4.10-3: 32 bit Timer with Prescaler Configuration

Prescale	Clock	Max Time	Units
00000000	1	268.435	s
00000001	2	536.871	s
00000010	3	805.306	s
--	--	--	--
11111101	254	0.682	10^5 s
11111110	255	0.685	10^5 s
11111111	256	0.687	10^5 s

Input Edge Count Mode

Note: *For rising-edge detection, the input signal must be HIGH for at least two system clock periods following the rising edge. Similarly, for falling-edge detection, the input signal must be LOW for at least two system clock periods following the falling edge. Based on this criteria, the maximum input frequency for edge detection is 1/4 of the system frequency.*

Note: *The prescaler is not available in 16-Bit Input Edge Count Mode.*

In Edge Count Mode, the timer is configured as a down-counter and is capable of capturing three types of events: rising edge, falling edge, or both. To place the timer in Edge Count Mode, the **GPTM_TnMR**.*MODE* bit must be set to 0x3 and **GPTM_TnMR**.*CM* must be cleared. The type of edge that the timer counts is determined by the *TnEVENT* fields of the **GPTM_CTL** register.

During initialization, the GPTM Timer Match **(GPTM_TnMATCHR)** register is configured so that the difference between the value in the **GPTM_TnILR** register and the **GPTM_TnMATCHR** register equals the number of edge events that must be counted.

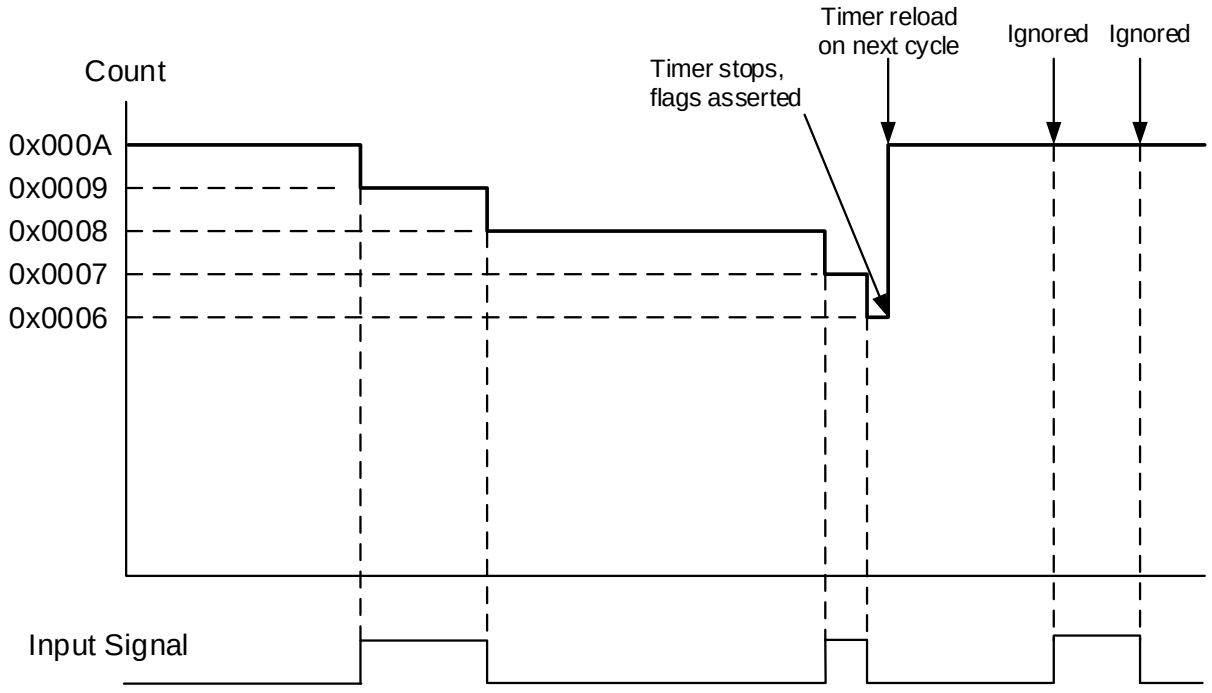
When software writes the *TnEN* bit in the GPTM Control **(GPTM_CTL)** register, the timer is enabled for event capture. Each input event on the CCP pin decrements the counter by 1 until the event count matches **GPTM_TnMATCHR**. When the counts match, the GPTM asserts the *CnMRI* bit in the **GPTM_RIS** register (and set the *CnMISC* bit, if the interrupt is not masked).

The counter is then reloaded using the value in **GPTM_TnILR**, and stopped since the GPTM automatically clears the *TnEN* bit in the **GPTM_CTL** register. Once the event count has been reached, all further events are ignored until *TnEN* bit is re-enabled by software.

The following figure shows how input edge count mode works. In this case, the timer start value is set to **GPTM_TnILR** = 0x000A and the match value is set to **GPTM_TnMATCHR** = 0x0006 so that three edge events are counted. The counter is configured to detect both edges of the input signal.

Note: *that the last two edges are not counted since the timer automatically clears the **TnEN** bit after the current count matches the value in the **GPTM_TnMATCHR** register.*

Figure 4.10-2: 16-Bit Input Edge Count Mode Example



Input Edge Time Mode

Note: *For rising-edge detection, the input signal must be HIGH for at least two system clock periods following the rising edge. Similarly, for falling edge detection, the input signal must be Low for at least two system clock periods following the falling edge. Based on this criteria, the maximum input frequency for edge detection is 1/4 of the system frequency.*

Note: *The prescaler is not available in 16-Bit Input Edge Time Mode.*

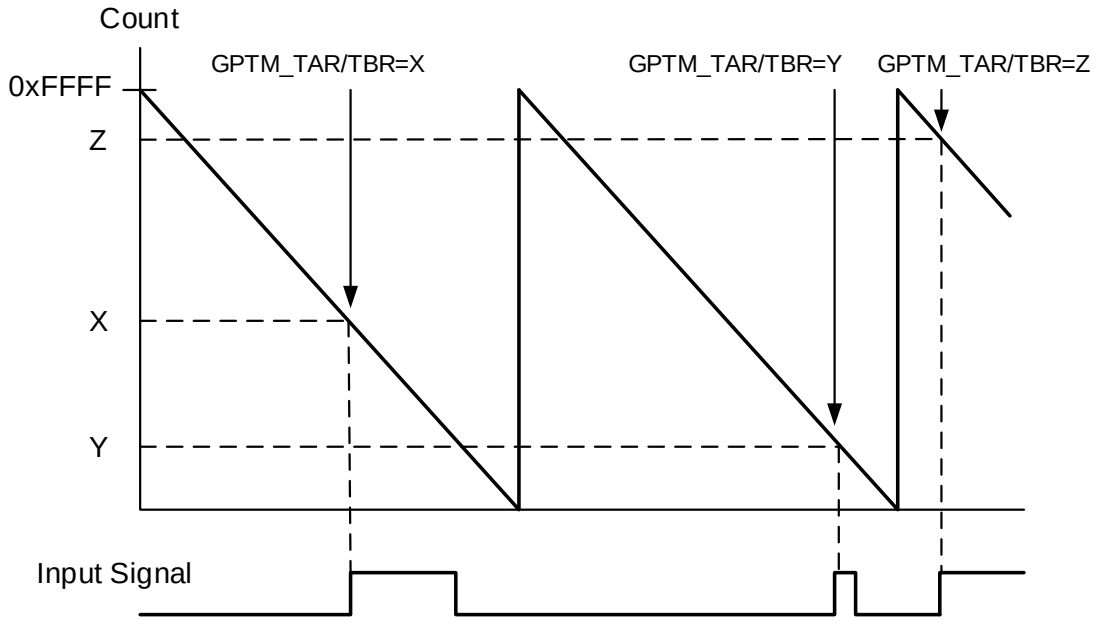
In Edge Time mode, the timer is configured as a free-running down-counter initialized to the value loaded in the **GPTM_TnILR** register. The timer is capable of capturing three types of events: rising edge, falling edge, or both. The timer is placed into Edge Time mode by setting the **GPTM_TnMR.***MODE* bit, and the type of event that the timer captures is determined by the **GPTM_CTL**.*TnEVENT* fields.

When software writes the **GPTM_CTL**.*TnEN* bit, the timer is enabled for event capture. When the selected input event is detected, the current timer counter value is captured in the **GPTM_TnR** register and is available to be read by the controller. The GPTM then asserts the *CnERI* bit (and the *CnEISC* bit, if the interrupt is not masked).

After an event has been captured, the timer does not stop counting. It continues to count until the *TnEN* bit is cleared. When the timer reaches the 0x0000 state, it is reloaded with the value from the **GPTM_TnILR** register.

Following figure shows how input edge timing mode works. In the diagram, it is assumed that the start value of the timer is the default value of 0xFFFF, and the timer is configured to capture rising edge events. Each time a rising edge event is detected, the current count value is loaded into the **GPTM_TnR** register, and is held there until another rising edge is detected (at which point the new count value is loaded into **GPTM_TnR**).

Figure 4.10-3: 16-Bit Input Edge Time Mode Example



Input PWM Edge Detect Mode

Note: *For rising-edge detection, the input signal must be high for at least two system clock periods following the rising edge. Similarly, for falling edge detection, the input signal must be Low for at least two system clock periods following the falling edge. Based on this criteria, the maximum input frequency for edge detection is 1/4 of the system frequency.*

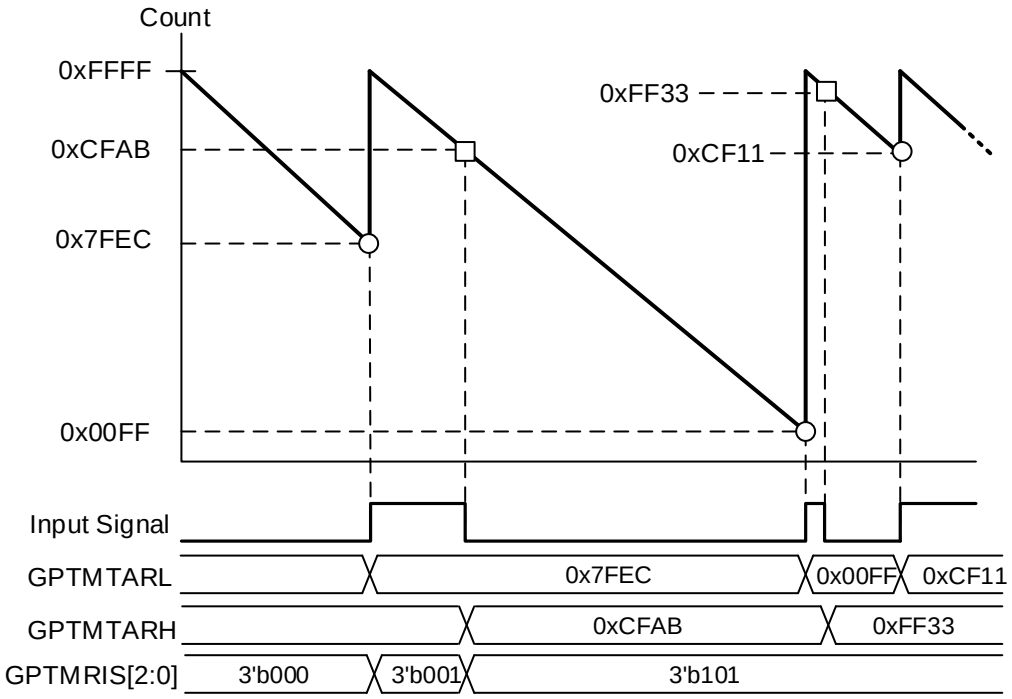
Note: *The prescaler is not available in 16-Bit Input PWM Edge Detect Mode.*

In PWM edge detect mode, the timer is configured as a free-running down-counter initialized to the value loaded in the **GPTM_TnILR** register. The timer capture mode must be set in both edge mode. The timer is placed into PWM edge detect mode by setting the **GPTM_TnMR**.*MODE* bit, and the type of event that the timer captures is determined by writing 0x3 to **GPTM_CTL.***TnEVENT* fields.

When software writes the *TnEN* bit in the **GPTM_CTL** register, the timer is enabled for event capture. When the rising edge is detected, the current timer counter value is captured in the **GPTM_TnR** register's low 15 bits and is available to be read by the controller. And when the falling edge is detected, the current timer counter value is captured in the **GPTM_TnR** register's high 15 bits and is available to be read by the controller. The GPTM then asserts the *CnERI* bit (and the *CnEISC* bit, if the interrupt is not masked) If detect the rising edge, the *TnTORI* will be set. After a rising edge was be detected or the timer reaches the 0, it is reloaded with the value from the **GPTM_TnILR** register. When detect the falling edge, the timer does not stop counting until it detect the rising edge or the *TnEN* bit is cleared.

Figure shows how input PWM edge detect mode works. In the diagram, it is assumed that the start value of the timer is the default value of 0xFFFF, and the timer is configured to capture both edge events.

Figure 4.10-4: 16-Bit Input Edge Detect Mode Example



PWM Mode

Note: *The prescaler is not available in 16-Bit PWM mode. The GPTM supports a simple PWM generation mode.*

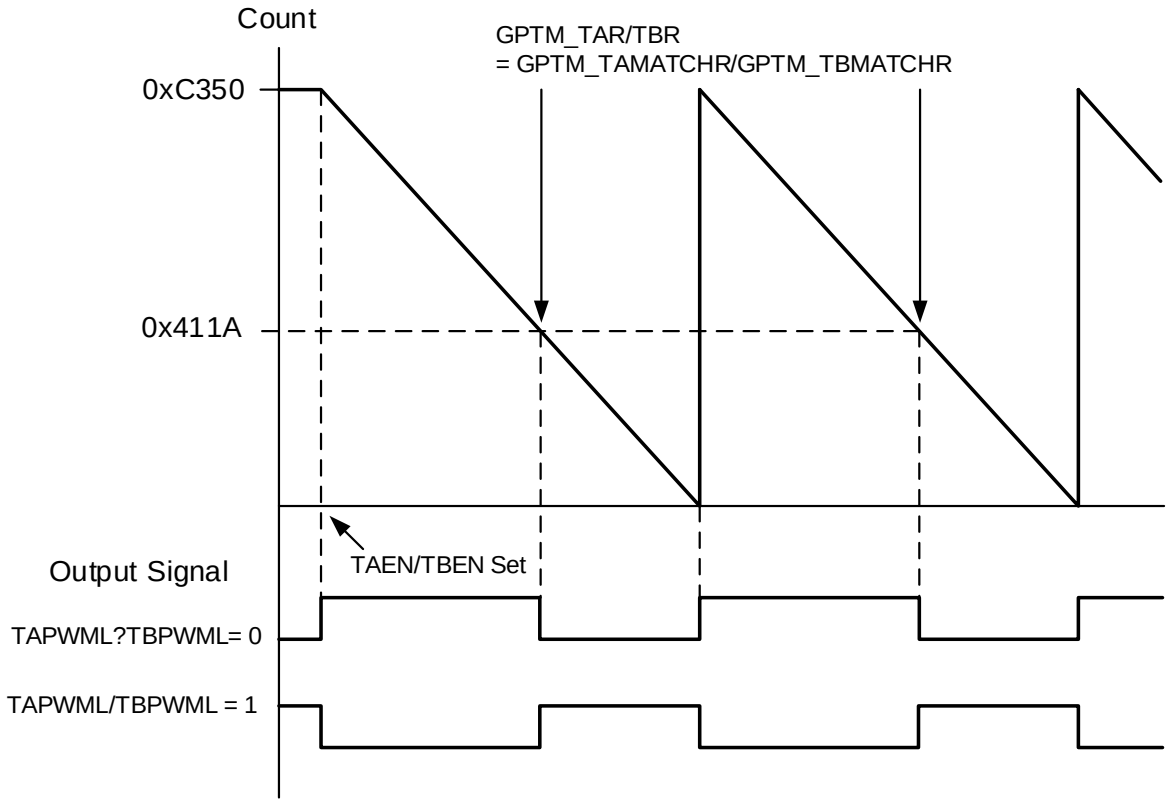
In PWM mode, the timer is configured as a down-counter with a start value (and thus period) defined by **GPTM_TnILR**. In this mode, the PWM frequency and period are synchronous events and therefore guaranteed to be glitch free. PWM mode is enabled with the **GPTM_TnMR** register by setting the *AMS* bit to 0x1, the *CM* bit to 0x0, and the *MODE* field to 0x0 or 0x2.

When software writes the *TnEN* bit in the **GPTM_CTL** register, the counter begins counting down until it reaches the 0x0000 state. On the next counter cycle, the counter reloads its start value from **GPTM_TnILR** and continues counting until disabled by software clearing the *TnEN* bit in the **GPTM_CTL** register. No interrupts or status bits are asserted in PWM mode.

The output PWM signal asserts when the counter is at the value of the **GPTM_TnILR** register (its start state), and is de-asserted when the counter value equals the value in the GPTM Match Register **(GPTM_TnMATCHR)**. Software has the capability of inverting the output PWM signal by setting the *TnPWML* bit in the **GPTM_CTL** register.

Figure shows how to generate an output PWM with a 1-ms period and a 66% duty cycle assuming a 50-MHz input clock and GPTM Match *TnPWML* =0 (duty cycle would be 33% for the *TnPWML* = 1 configuration). For this example, the start value is G**PTM_TnIRL**= 0xC350 and the match value is **GPTM_TnMATCHR** = 0x411A.

Figure 4.10-5: 16-Bit PWM Detect Mode Example



Interrupt Control

Interrupt generation at capture event, capture match and timer time-out. The interrupt in gptimer are controlled by a set of five registers.

- Interrupt Control (IER, IDR, IMR)
  GPTM Interrupt enable register (GPTM_IER) enables the interrupt request lines by writing a '1'. Similarly, **GPTM Interrupt disable register (GPTM_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 **GPTM Interrupt Mask Register (GPTM_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)
  GPTM Raw Interrupt Status (GPTM_RIS) is a read-only register to read all interrupt status of the module.
- Interrupt Clear (ISC)
  GPTM Interrupt Status & Interrupt Clear Register (GPTM_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.

4.10.3 GPTM INITIALIZATION AND CONFIGURATION

This section shows module initialization and configuration examples for each of the supported timer modes.

4.10.3.1 32-BIT ONE-SHOT/PERIODIC TIMER MODE

The GPTM is configured for 32-bit One-Shot and Periodic modes by the following sequence:

1. Ensure the timer is disabled (the *TnEN* bit in the **GPTM_CTL** register is cleared) before making any changes.
2. Write the GPTM Configuration Register (**GPTM_CFG**) with a value of 0x0.
3. Set the *MODE* field in the GPTM Timer-A/B Mode Register (**GPTM_TnMR**):
   a. Write a value of 0x1 for One-Shot mode.
   b. Write a value of 0x2 for Periodic mode.
4. If a prescaler is to be used, write the prescale value to the GPTM Timer Prescale Register **(GPTM_TnPR)**.
5. Load the start value into the GPTM Timer-A/B Interval Load Register (**GPTM_TnILR**).
6. If interrupts are required, set the *TnTOIE* bit in the GPTM Interrupt Enable Register (**GPTM_IER**).
7. Set the *TnEN* bit in the **GPTM_CTL** register to enable the timer and start counting.
8. Poll the *TnTORIS* bit in the **GPTM_RIS** register or wait for the interrupt to be generated (if enabled).In both cases, the status flags are cleared by writing a 1 to the *TnTOISC* bit of the GPTM Interrupt Status and Clear Register **(GPTM_ISC)**.

In One-Shot mode, the timer stops counting after step 8. To re-enable the timer, repeat the sequence. A timer configured in Periodic mode does not stop counting after it times out.

4.10.3.2 16-BIT ONE-SHOT/PERIODIC TIMER MODE

A timer is configured for 16-bit One-Shot and Periodic modes by the following sequence:

1. Ensure the timer is disabled (the *TnEN* bit is cleared) before making any changes.
2. Write the GPTM Configuration Register (**GPTM_CFG**) with a value of follows:
   a. Write a value of 0x4 for One-Shot mode.
b. Write a value of 0x5 for Periodic mode.
3. Set the *MODE* field in the GPTM Timer Mode (**GPTM_TnMR**) register:
   a. Write a value of 0x1 for One-Shot mode.
   b. Write a value of 0x2 for Periodic mode.
4. If a prescaler is to be used, write the prescale value to the GPTM Timer Prescale Register **(GPTM_TnPR)**.
5. Load the start value into the GPTM Timer Interval Load Register (**GPTM_TnILR**).
6. If interrupts are required, set the *TnTOIE* bit in the GPTM Interrupt Enable Register (**GPTM_IER**).
7. Set the *TnEN* bit in the GPTM Control Register **(GPTM_CTL)** to enable the timer and start counting.
8. Poll the *TnTORI* bit in the **GPTM_RIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the *TnTOISC* bit of the GPTM Interrupt Status and Clear Register **(GPTM_ISC)**.

In One-Shot mode, the timer stops counting after step 8. To re-enable the timer, repeat the sequence. A timer configured in Periodic mode does not stop counting after it times out.

4.10.3.3 16-BIT INPUT EDGE COUNT MODE

A timer is configured to Input Edge Count Mode by the following sequence:

1. Ensure the timer is disabled (the *TnEN* bit is cleared) before making any changes.
2. Write the GPTM Configuration **(GPTM_CFG)** register with a value of 0x6.
3. In the GPTM Timer Mode **(GPTM_TnMR)** register, write the *CM* field to 0x0 and the *MODE* field to 0x3.
4. Configure the type of event(s) that the timer captures by writing the *TnEVENT* field of the GPTM Control **(GPTM_CTL)** register.
   a. Write a value of 0x0 for positive edge.
   b. Write a value of 0x1 for negative edge.
   c. Write a value of 0x3 for both edge.
5. Load the timer start value into the GPTM Timer Interval Load (**GPTM_TnILR**) register.
6. Load the desired event count into the GPTM Timer Match **(GPTM_TnMATCHR)** register.
7. If debounce are required, set GPTM Timer Debounce **(GPTM_DBC)** register
8. If interrupts are required, set the *CnMIE* bit in the GPTM Interrupt Enable **(GPTM_IER)** register.
9. Set the *TnEN* bit in the **GPTM_CTL** register to enable the timer and begin waiting for edge events.
10. Poll the *CnMRI* bit in the **GPTM_RIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the *CnMISC* bit of the GPTM Interrupt Status and Clear **(GPTM_ISC)** register.

In Input Edge Count Mode, the timer stops after the desired number of edge events has been detected. To re-enable the timer, ensure that the TnENbit is cleared and repeat step 4, through step 10.

4.10.3.4 16-BIT INPUT EDGE TIMING MODE

A timer is configured to Input Edge Timing mode by the following sequence:

1. Ensure the timer is disabled (the *TnEN* bit is cleared) before making any changes.
2. Write the GPTM Configuration **(GPTM_CFG)** register with a value of 0x6.
3. In the GPTM Timer Mode **(GPTM_TnMR)** register, write the *CM* field to 0x1 and the *MODE* field to 0x3.
4. Configure the type of event that the timer captures by writing the *TnEVENT* field of the GPTM Control **(GPTM_CTL)** register.
   a. Write a value of 0x0 for positive edge.
   b. Write a value of 0x1 for negative edge.
   c. Write a value of 0x3 for both edge.
5. Load the timer start value into the GPTM Timer Interval Load (**GPTM_TnILR**) register.
6. If interrupts are required, set the *CnEIE* bit in the GPTM Interrupt Enable **(GPTM_IER)** register.
7. If debounce are required, set GPTM Timer Debounce **(GPTM_DBC)** register.
8. Set the *TnEN* bit in the GPTM Control **(GPTM_CTL)** register to enable the timer and start counting.
9. Poll the *CnERI* bit in the **GPTM_RIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the *CnEISC* bit of the GPTM Interrupt Status and Clear **(GPTM_ISC)** register. The time at which the event happened can be obtained by reading the GPTM Timer Mode **(GPTM_TnR)** register.

In Input Edge Timing mode, the timer continues running after an edge event has been detected, but the timer interval can be changed at any time by writing the **GPTM_TnILR** register. The change takes effect at the next cycle after the write.

4.10.3.5 16-BIT INPUT PWM EDGE DETECT MODE

A timer is configured to Input PWM Edge Detect mode by the following sequence:

1. Ensure the timer is disabled (the *TnEN* bit is cleared) before making any changes.
2. Write the GPTM Configuration **(GPTM_CFG)** register with a value of 0x6.
3. In the GPTM Timer Mode **(GPTM_TnMR)** register, write the *CM* field to 0x1 and the *MODE* field to 0x3.
4. Configure the *TnEVENT* field of the GPTM Control **(GPTM_CTL)** register to 0x3, it means both edge.
5 .Load the timer start value into the GPTM Timer Interval Load (**GPTM_TnILR**) register.
6. If debounce are required, set GPTM Timer Debounce **(GPTM_DBC)** register.
7. If interrupts are required, set the *CnEIE* bit and *TnTOIE* bit in the GPTM Interrupt Enable **(GPTM_IER)** register.
8. Set the *TnEN* bit in the GPTM Control **(GPTM_CTL)** register to enable the timer and begin waiting for input.
9. Poll the *CnERI* bit in the **GPTM_RIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the *CnEISC* bit of the GPTM Interrupt Status and Clear **(GPTM_ISC)** register. The time at which the event happened can be obtained by reading the GPTM Timer Mode **(GPTM_TnR)** register. When the *TnTOIE* was be set, its means the **GPTM_TnR** low 15 bits have a capture value for to calculate the PWM Duty Width (**GPTM_TnILR** - **GPTM_TnR** low 15 bits); And when the *CnEIE* was be set, its means the **GPTM_TnR** high 15 bits have a capture value for to calculate the PWM Positive width (**GPTM_TnILR** - **GPTM_TnR** high 15 bits).

4.10.3.6 16-BIT PWM MODE

A timer is configured to PWM mode using the following sequence:

1. Ensure the timer is disabled (the *TnEN* bit is cleared) before making any changes.
2. Write the GPTM Configuration **(GPTM_CFG)** register with a value of 0x7.
3. In the GPTM Timer Mode **(GPTM_TnMR)** register, set the *AMS* bit to 0x1, the *CM* bit to 0x0, and the *MODE* as follows:
   a. Write a value of 0x0 for triangle PWM mode.
   b. Write a value of 0x2 for PWM mode.
4. Configure the output state of the PWM signal (whether or not it is inverted) in the TnPWML field of the GPTM Control **(GPTM_CTL)** register.
5. Load the timer start value into the GPTM Timer Interval Load (**GPTM_TnILR**) register.
6. Load the GPTM Timer Match (**GPTM_TnMATCHR**) register with the desired value.
7. Set the *TnEN* bit in the GPTM Control **(GPTM_CTL)** register to enable the timer and begin generation of the output PWM signal.

In PWM Timing mode, the timer continues running after the PWM signal has been generated. The PWM period can be adjusted at any time by writing the **GPTM_TnILR** and **GPTM_TnMATCHR** register, and the change takes effect at the next cycle after the write.

4.10.4 GENERAL-PURPOSE TIMER REGISTER MAP

GPTM base address:

- GPTM0: 0x4000_0000
- GPTM1: 0x4000_0100
- GPTM2: 0x4000_0200

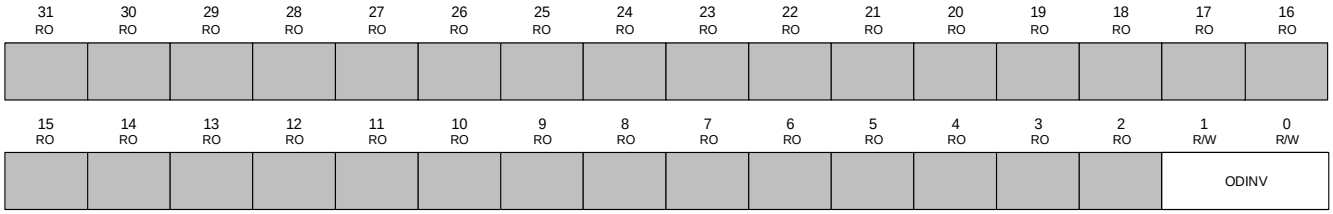
Offset	Symbol	Type	Reset Value	Description	See page
0x0000	GPTM_CFG	R/W	0x0000_0000	GPTM Configuration	200
0x0004	GPTM_TAMR	R/W	0x0000_0000	GPTM Timer-A Mode	201
0x0008	GPTM_TBMR	R/W	0x0000_0000	GPTM Timer-B Mode	202
0x000C	GPTM_CTL	R/W	0x0000_0000	GPTM Control	203
0x0010	GPTM_IER	WO	0x0000_0000	GPTM Interrupt Enable	204
0x0014	GPTM_IDR	WO	0x0000_0000	GPTM Interrupt Disable	205
0x0018	GPTM_IMR	RO	0x0000_0000	GPTM Interrupt Mask Status	206
0x001C	GPTM_RIS	RO	0x0000_0000	GPTM Raw Interrupt Status	207
0x0020	GPTM_ISC	R/W1C	0x0000_0000	GPTM Interrupt Status and Clear	208
0x0028	GPTM_TAILR	R/W	0xFFFF_FFFF	GPTM Timer-A Interval Load	209
0x002C	GPTM_TBILR	R/W	0xFFFF_FFFF	GPTM Timer-B Interval Load	209
0x0030	GPTM_TAMATCHR	R/W	0x0000_FFFF	GPTM Timer-A Match	210
0x0034	GPTM_TBMATCHR	R/W	0x0000_FFFF	GPTM Timer-B Match	210
0x0038	GPTM_TAPR	R/W	0x0000_FFFF	GPTM Timer-A Prescale	211
0x003C	GPTM_TBPR	R/W	0x0000_FFFF	GPTM Timer-B Prescale	211
0x0040	GPTM_DBC	R/W	0x0000_0000	GPTM Debounce	212
0x0048	GPTM_TAR	RO	0xFFFF_FFFF	GPTM Timer-A	213
0x004C	GPTM_TBR	RO	0xFFFF_FFFF	GPTM Timer-B	213

4.10.4.1 GPTM_CFG - GPTM CONFIGURATION

This register configures the global operation of the GPTM module. The value written to this register determines whether the GPTM is in 32 or 16 bit mode.

Before configuring this register, user must first stop timer's counting. There are two ways to stop the counting of the resespective timer:

1. Disable Timer n by clearing the *TnEN* bit in **GPTM_CTL** register. When Timer n is re-enabled, it starts counting from reset value.
2. Set *TnSTALL* bit in the **GPTM_CTL** register, and the Timer n counter's value is kept. Clear the *TnSTALL* bit after configuration, then the timer will starts counting from the kept value.



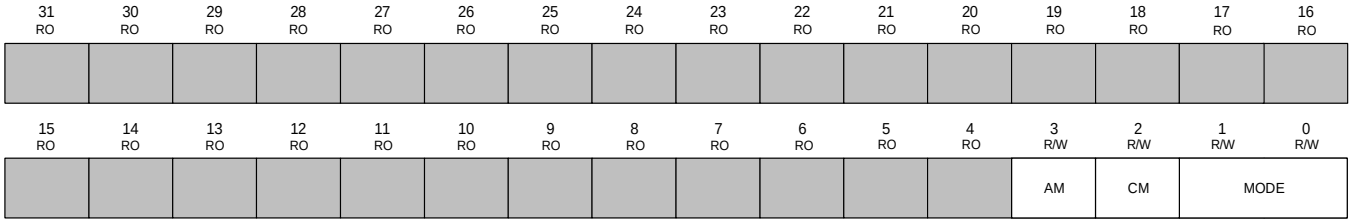
Offset: 0x0000

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	*CFG*	R/W	0x0	GPTM Configuration 0x0：Selects the 32-bit timer configuration 0x1- 0x3：Reserved. 0x4：Selects the 16-bit timer one-shot mode. 0x5：Selects the 16-bit timer periodic mode. 0x6：Selects the 16-bit timer capture mode. 0x7：Selects the 16-bit timer PWM mode.

4.10.4.2 GPTM_TAMR - GPTM TIMER-A MODE

This register configures the GPTM based on the configuration selected in the **GPTM_CFG** register. When in 16-bit PWM mode, set the *AMS* bit to 0x1, the *CM* bit to 0x0, and the *MODE* field to 0x0 or 0x2.

*Note: Bits in this register should only be changed when the **GPTM_CTL**.TAEN bit is cleared or the **GPTM_CTL**.TASTALL is set.*



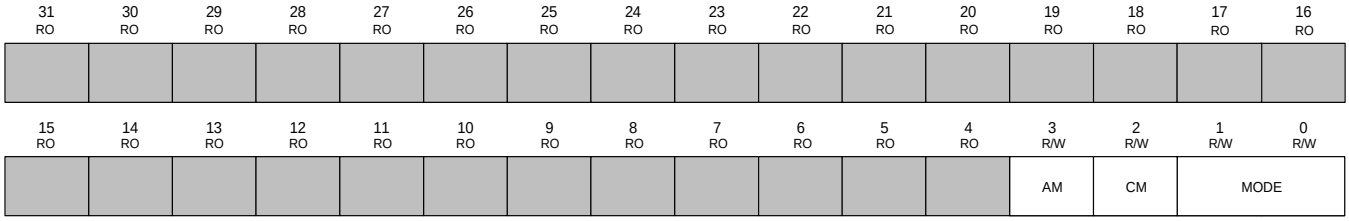
Offset: 0x0004

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	*AMS*	R/W	0	GPTM Timer A Alternate Mode Select 0: Capture mode is enabled. 1: PWM mode is enable.
2	*CAPM*	R/W	0	GPTM Timer A Capture Mode 0: Edge Count Mode. 1: Edge Time Mode.
1:0	*MODE*	R/W	0x0	GPTM Timer A Mode 0x0：PWM U/D mode. 0x1：One-Shot Timer mode. 0x2：Periodic Timer mode. 0x3：Capture mode.

4.10.4.3 GPTM_TBMR - GPTM TIMER-B MODE

This register configures the GPTM based on the configuration selected in the **GPTM_CFG** register. When in 16-bit PWM mode, set the *AMS* bit to 0x1, the *CM* bit to 0x0, and the *MODE* field to 0x0 or 0x2.

*Note: Bits in this register should only be changed when the **GPTM_CTL**.TBEN bit is cleared or the **GPTM_CTL**.TBSTALL is set.*



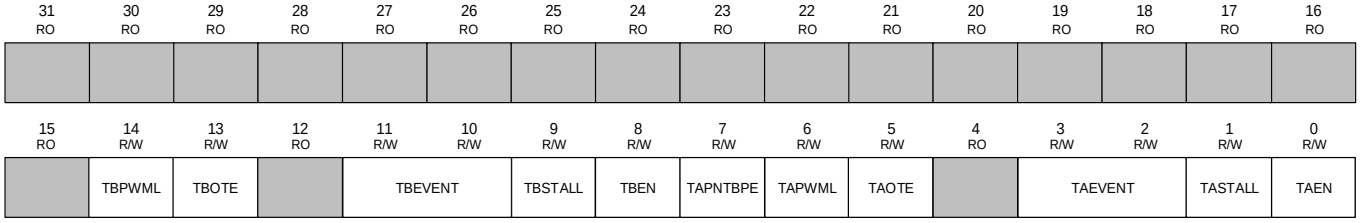
Offset: 0x0008

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	*AMS*	R/W	0	GPTM Timer B Alternate Mode Select 0: Capture mode is enabled. 1: PWM mode is enable.
2	*CM*	R/W	0	GPTM Timer B Capture Mode 0: Edge Count Mode. 1: Edge Time Mode.
1:0	*MODE*	R/W	0x0	GPTM Timer B Mode 0x0：PWM U/D mode. 0x1：One-Shot Timer mode. 0x2：Periodic Timer mode. 0x3：Capture mode.

4.10.4.4 GPTM_CTL - GPTM CONTROL

This register is used alongside the **GPTM_CFG** and **GPTM_TAMR/TBMR** registers to fine-tune the timer configuration, and to enable other features such as timer stall and the output trigger. The output trigger can be used to initiate transfers on the ADC module.

*Note: Bits in this register should only be changed when the **GPTM_CTL**.TnEN bit is cleared or the **GPTM_CTL**.TnSTALL is set.*

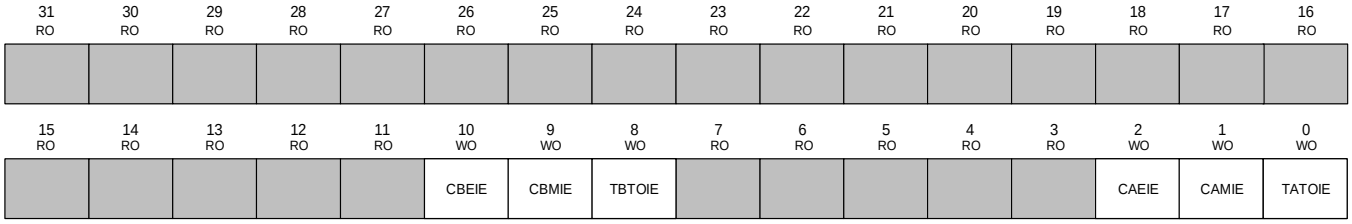


Offset: 0x000C

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	*TBPWML*	R/W	0	GPTM Timer B PWM Output Level. 0: Output is unaffected. 1: Output is inverted.
13	*TBOTE*	R/W	0	GPTM Timer B Output Trigger Enable. 0: The output Timer B ADC trigger is disabled. 1: The output Timer B ADC trigger is enabled.
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.
11:10	*TBEVENT*	R/W	0x0	GPTM Timer B Event mode. 0x0: Positive edge. 0x1: Negative edge. 0x2: Reserved. 0x3: Both edges.
9	*TBSTALL*	R/W	0	GPTM Timer B Stall Enable. 0: Timer B continues counting. 1: Timer B freezes counting.
8	*TBEN*	R/W	0	GPTM Timer B Enable 0: Timer B is disabled. 1: Timer B is enabled and begins counting.
7	*TAPNTBPE*	R/W	1	GPTM Timer A PWM Output and Timer-B PWM Output are Complementary 0: Timer A PWM Output and Timer-B PWM Output Complementary is disabled. 1: Timer A PWM Output and Timer-B PWM Output Complementary is enabled
6	*TAPWM*	R/W	0	GPTM Timer A PWM Output Level. 0: Output is unaffected. 1: Output is inverted.
5	*TAOTE*	R/W	0	GPTM Timer A Output Trigger Enable. 0: The output Timer A ADC trigger is disabled. 1: The output Timer A ADC trigger is enabled.

Bits	Name	Type	Reset	Description
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.
3:2	*TAEVENT*	R/W	0x0	GPTM Timer A Event mode. 0x0: Positive edge. 0x1: Negative edge. 0x2: Reserved. 0x3: Both edges.
1	*TASTALL*	R/W	0	GPTM Timer A Stall Enable. 0: Timer A continues counting. 1: Timer A freezes counting.
0	*TAEN*	R/W	0	GPTM Timer A Enable 0: Timer A is disabled. 1: Timer A is enabled and begins counting.

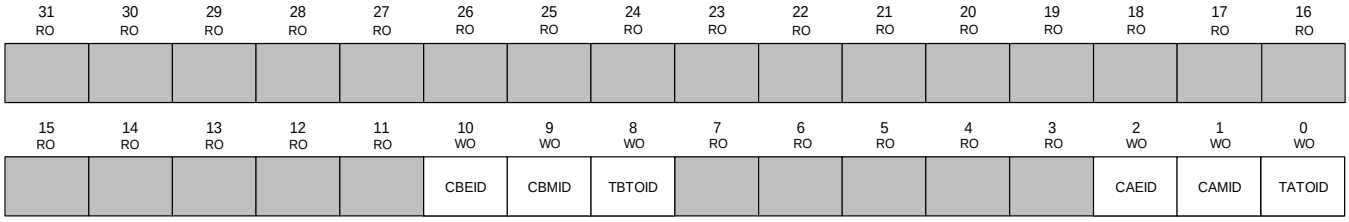
4.10.4.5 GPTM_IER - GPTM INTERRUPT ENABLE REGISTER



Offset: 0x0010

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	*CBEIE*	WO	0	GPTM Timer B Capture Mode Event Interrupt Enable 1: Interrupt is enabled.
9	*CBMIE*	WO	0	GPTM Timer B Capture Mode Match Interrupt Enable 1: Interrupt is enabled.
8	*TBTOIE*	WO	0	GPTM Timer B Time-Out Interrupt Enable 1: Interrupt is enabled.
7:3	*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	*CAEIE*	WO	0	GPTM Timer A Capture Mode Event Interrupt Enable 1: Interrupt is enabled.
1	*CAMIE*	WO	0	GPTM Timer A Capture Mode Match Interrupt Enable 1: Interrupt is enabled.
0	*TATOIE*	WO	0	GPTM Timer A Time-Out Interrupt Enable 1: Interrupt is enabled.

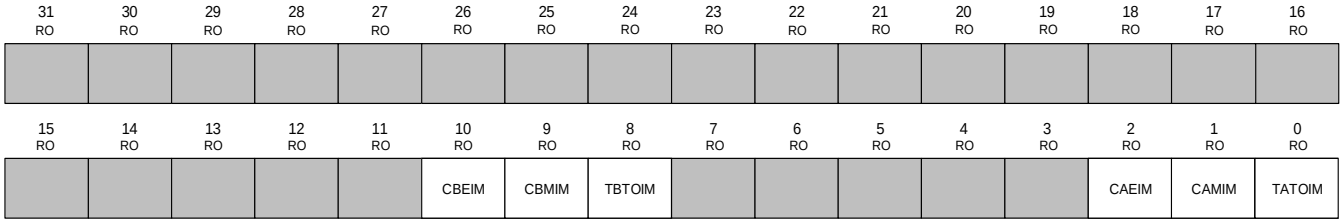
4.10.4.6 GPTM_IDR – GPTM INTERRUPT DISABLE REGISTER



Offset: 0x0014

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	*CBEID*	WO	0	GPTM Timer B Capture Mode Event Interrupt Disable 1: Interrupt is disabled.
9	*CBMID*	WO	0	GPTM Timer B Capture Mode Match Interrupt Disable 1: Interrupt is disabled.
8	*TBTOID*	WO	0	GPTM Timer B Time-Out Interrupt Disable 1: Interrupt is disabled.
7:3	*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	*CAEID*	WO	0	GPTM Timer A Capture Mode Event Interrupt Disable 1: Interrupt is disabled.
1	*CAMID*	WO	0	GPTM Timer A Capture Mode Match Interrupt Disable 1: Interrupt is disabled.
0	*TATOID*	WO	0	GPTM Timer A Time-Out Interrupt Disable 1: Interrupt is disabled.

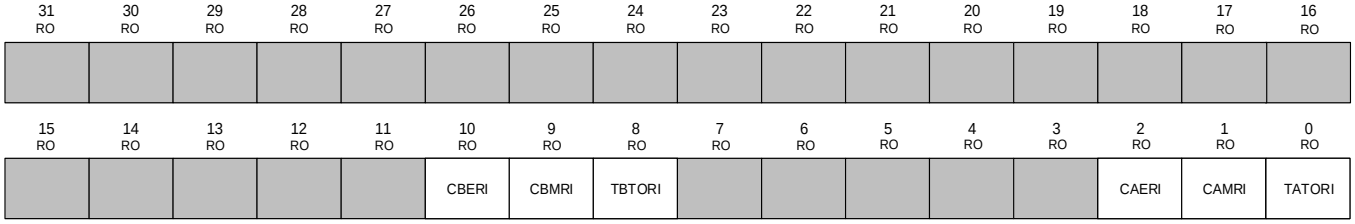
4.10.4.7 GPTM_IMR - GPTM INTERRUPT MASK STATUS



Offset: 0x0018

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	*CBEIM*	RO	0	GPTM Timer B Capture Mode Event Interrupt Mask 0: Interrupt will be masked. 1: Interrupt is not masked.
9	*CBMIM*	RO	0	GPTM Timer B Capture Mode Match Interrupt Mask 0: Interrupt will be masked. 1: Interrupt is not masked.
8	*TBTOIM*	RO	0	GPTM Timer B Time-Out Interrupt Mask 0: Interrupt will be masked. 1: Interrupt is not masked.
7:3	*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	*CAEIM*	RO	0	GPTM Timer A Capture Mode Event Interrupt Mask 0: Interrupt will be masked. 1: Interrupt is not masked.
1	*CAMIM*	RO	0	GPTM Timer A Capture Mode Match Interrupt Mask 0: Interrupt will be masked. 1: Interrupt is not masked.
0	*TATOIM*	RO	0	GPTM Timer A Time-Out Interrupt Mask 0: Interrupt will be masked. 1: Interrupt is not masked.

4.10.4.8 GPTM_RIS - GPTM RAW INTERRUPT STATUS

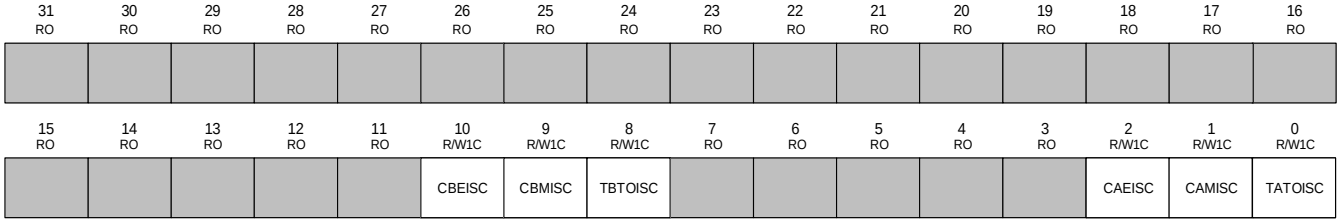


Offset: 0x001C

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	*CBERI*	RO	0	GPTM Timer B Capture Mode Event Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
9	*CBMRI*	RO	0	GPTM Timer B Capture Mode Match Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
8	*TBTORI*	RO	0	GPTM Timer B Time-Out Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
7:3	*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	*CAERI*	RO	0	GPTM Timer A Capture Mode Event Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
1	*CAMRI*	RO	0	GPTM Timer A Capture Mode Match Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
0	*TATORI*	RO	0	GPTM Timer A Time-Out Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.10.4.9 GPTM_ISC - GPTM 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.

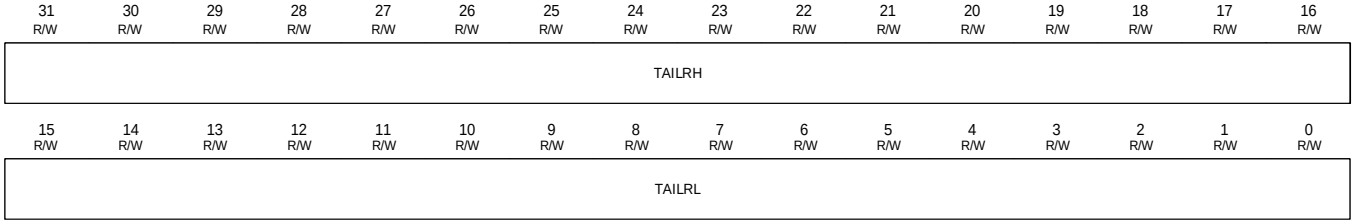


Offset: 0x0020

Bits	Name	Type	Reset	Description
31:11	*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	*CBEISC*	R/W1C	0	GPTM Timer B Capture Mode Event Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Timer B Capture Mode Event interrupt has been signaled.
9	*CBMISC*	R/W1C	0	GPTM Timer B Capture Mode Match Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Timer B Capture Mode Match interrupt has been signaled.
8	*TBTOISC*	R/W1C	0	GPTM Timer B Time-Out Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Timer B Time-Out interrupt has been signaled.
7:3	*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	*CAEISC*	R/W1C	0	GPTM Timer A Capture Mode Event Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Timer A Capture Mode Event interrupt has been signaled
1	*CAMISC*	R/W1C	0	GPTM Timer A Capture Mode Match Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Timer A Capture Mode Match interrupt has been signaled.
0	*TATOISC*	R/W1C	0	GPTM Timer A Time-Out Interrupt Status and clear 0: No interrupt or the interrupt has been masked. 1: Timer A Time-Out interrupt has been signaled.

4.10.4.10 GPTM_TAILR - GPTM TIMER-A INTERVAL LOAD

This register is used to load the starting count value into the Timer A.

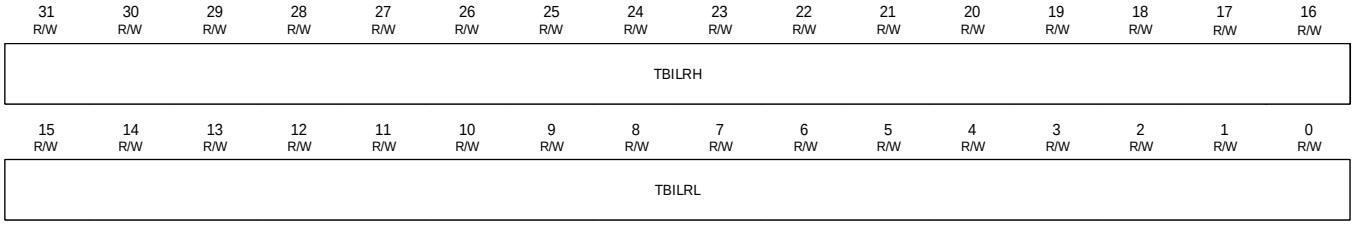


Offset: 0x0028

Bits	Name	Type	Reset	Description
31:16	*TAILRH*	R/W	0xFFFF	GPTM Timer-A Interval Load Register High High for 32-bit mode, writing this field loads the counter for Timer A. A read returns the current value of **GPTM_TAILR**. In 16-bit mode, a read returns 0.
15:0	*TAILRL*	R/W	0xFFFF	GPTM Timer-A Interval Load Register Low Low for both 16- and 32-bit modes, writing this field loads the counter for Timer A. A read returns the current value of **GPTM_TAILR**.

4.10.4.11 GPTM_TBILR - GPTM TIMER-B INTERVAL LOAD

This register is used to load the starting count value into the Timer B.



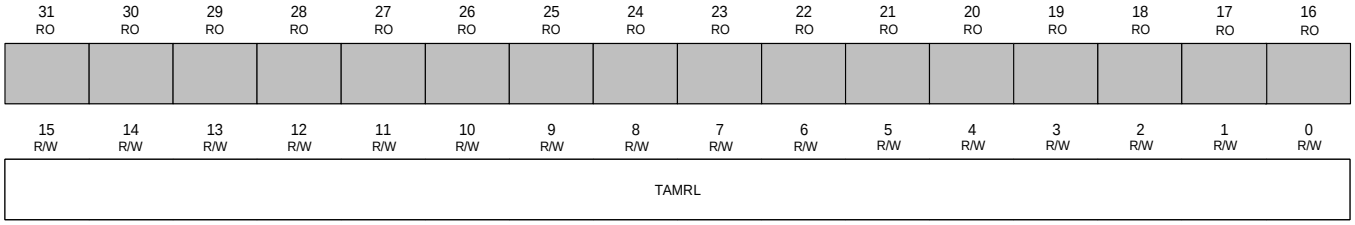
Offset: 0x002C

Bits	Name	Type	Reset	Description
31:16	*TBILRH*	R/W	0xFFFF	GPTM Timer B Interval Load Register High High for 32-bit mode, writing this field loads the counter for Timer B. A read returns the current value of **GPTM_TAILR**. In 16-bit mode, a read returns 0.
15:0	*TBILRL*	R/W	0xFFFF	GPTM Timer B Interval Load Register Low Low for both 16- and 32-bit modes, writing this field loads the counter for Timer B. A read returns the current value of **GPTM_TAILR**.

4.10.4.12 GPTM_TAMATCHR - GPTM TIMER A MATCH

This register is loaded with a match value. Interrupt can be generated when the timer value is equal to the value in this register in one-shot or periodic mode.
In edge count mode, this register along with **GPTM_TAILR**, determines how many edge events are counted. The total number of edge events counted is equal to the value in **GPTM_TAILR** minus this value.

In PWM mode, this register along with **GPTM_TAILR**, determines the duty cycle of the output PWM signal.



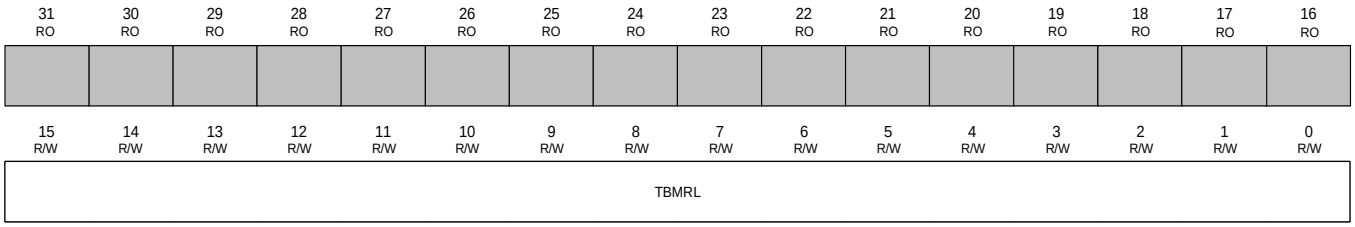
Offset: 0x0030

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	*TAMRL*	R/W	0xFFFF	GPTM Timer A Match Register Low.

4.10.4.13 GPTM_TBMATCHR - GPTM TIMER B MATCH

This register is loaded with a match value. Interrupt can be generated when the timer value is equal to the value in this register in one-shot or periodic mode.
In edge count mode, this register along with **GPTM_TBILR**, determines how many edge events are counted. The total number of edge events counted is equal to the value in **GPTM_TBILR** minus this value.

In PWM mode, this register along with **GPTM_TBLR**, determines the duty cycle of the output PWM signal.

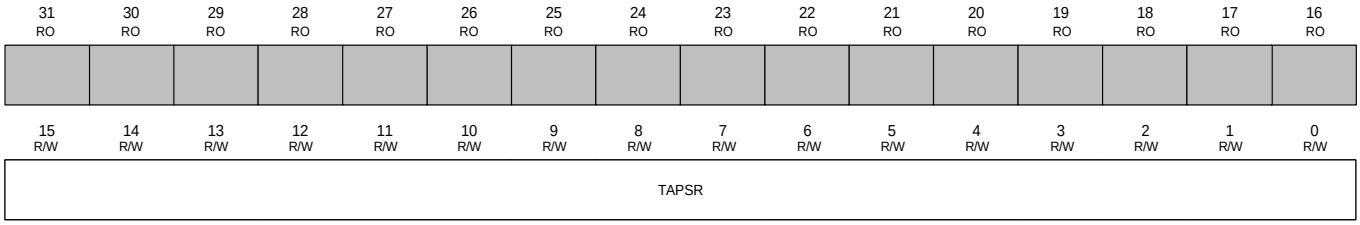


Offset: 0x0034

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	*TBMRL*	R/W	0xFFFF	GPTM Timer B Match Register Low

4.10.4.14 GPTM_TAPR - GPTM TIMER-A PRESCALE

This register allows software to extend the range of the 16-bit timers when operating in one-shot or periodic mode.

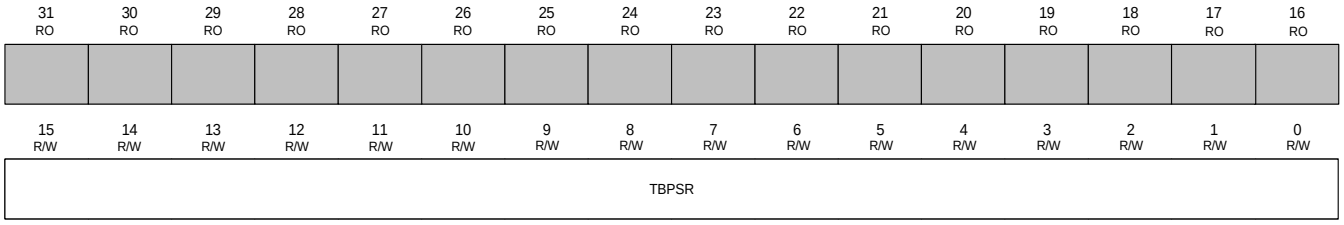


Offset: 0x0038

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	*TAPSR*	R/W	0xFFFF	GPTM Timer A Prescale The register loads this value on a write. A read returns the current value of the register.

4.10.4.15 GPTM_TBPR - GPTM TIMER-B PRESCALE

This register allows software to extend the range of the 16-bit timers when operating in one-shot or periodic mode.



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	*TBPSR*	R/W	0xFFFF	GPTM Timer B Prescale The register loads this value on a write. A read returns the current value of the register.

4.10.4.16 GPTM_DBC - GPTM TIMER DEBOUNCE

This register is only valid in Capture mode, the input signals is bounce-eliminated. For example, assume that the system clock frequency is 16MHz, and set 0x03 to *DBCNTA*. When the input signal come from Even ccp, signal above 4MHz is ignored. If the system clock frequency is 16MHz, RMS *DBCNTn* as follows:

DBCNTn: "0x0000_0000" The direct input.

DBCNTn: "0x0000_0001" After more than 8 MHz signal is ignored, and check the input.

DBCNTn: "0x0000_0011" After more than 4 MHz signal is ignored, and check the input.

DBCNTn: "0x0000_0111" After more than 2 MHz signal is ignored, and check the input.

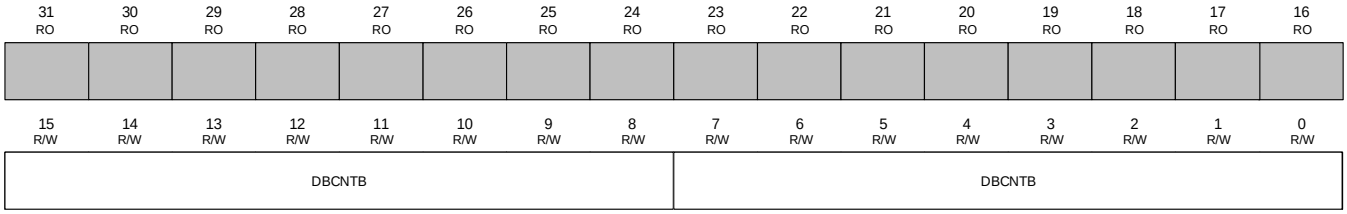
DBCNTn: "0x0000_1111" After more than 1 MHz signal is ignored, and check the input.

DBCNTn: "0x0001_1111" After more than 0.5 MHz signal is ignored, and check the input.

DBCNTn: "0x0011_1111" After more than 250 KHz signal is ignored, and check the input.

DBCNTn: "0x0111_1111" After more than 125 KHz signal is ignored, and check the input.

DBCNTn: "0x1111_1111" After more than 62.5 KHz signal is ignored, and check the input.

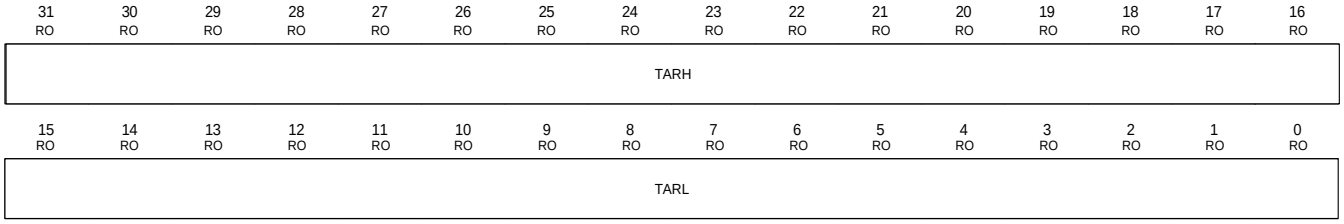


Offset: 0x0040

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.
15:8	*DBCNTB*	R/W	0x0	Debounce input for timer-B ccp.
7:0	*DBCNTA*	R/W	0x0	Debounce input for timer-A ccp.

4.10.4.17 GPTM_TAR - GPTM TIMER-A

This register shows the current value of the Timer-A counter.

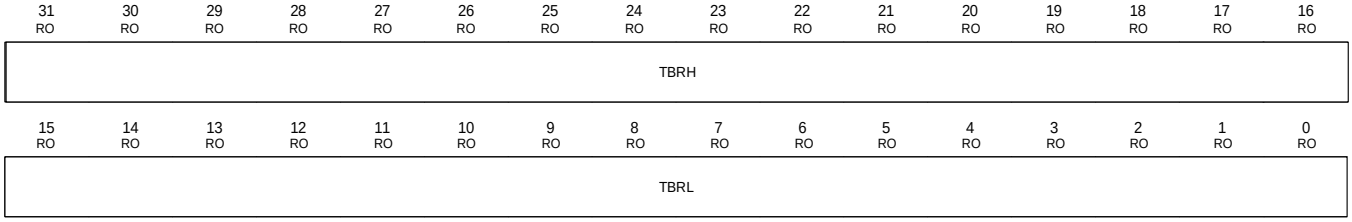


Offset: 0x0048

Bits	Name	Type	Reset	Description
31:16	*TARH*	RO	0xFFFF	GPTM Timer A Register High If CFG bit is set as 32-bit mode, Timer A value is read. If CFG bit is set as 16-bit mode, a read returns no meaning.
15:0	*TARL*	RO	0xFFFF	GPTM Timer A Register Low A read returns the current value of the Timer A counter in all cases except in Input-edge count mode, which returns the number of edges that have occurred.

4.10.4.18 GPTM_TBR - GPTM TIMER-B

This register shows the current value of the Timer-B counter.



Offset: 0x004C

Bits	Name	Type	Reset	Description
31:16	*TBRH*	RO	0xFFFF	GPTM Timer B Register High If CFG bit is set as 32-bit mode, Timer B value is read. If CFG bit is set as 16-bit mode, a read returns no meaning.
15:0	*TBRL*	RO	0xFFFF	GPTM Timer B Register Low A read returns the current value of the Timer B counter in all cases except in Input-edge count mode, which returns the number of edges that have occurred.

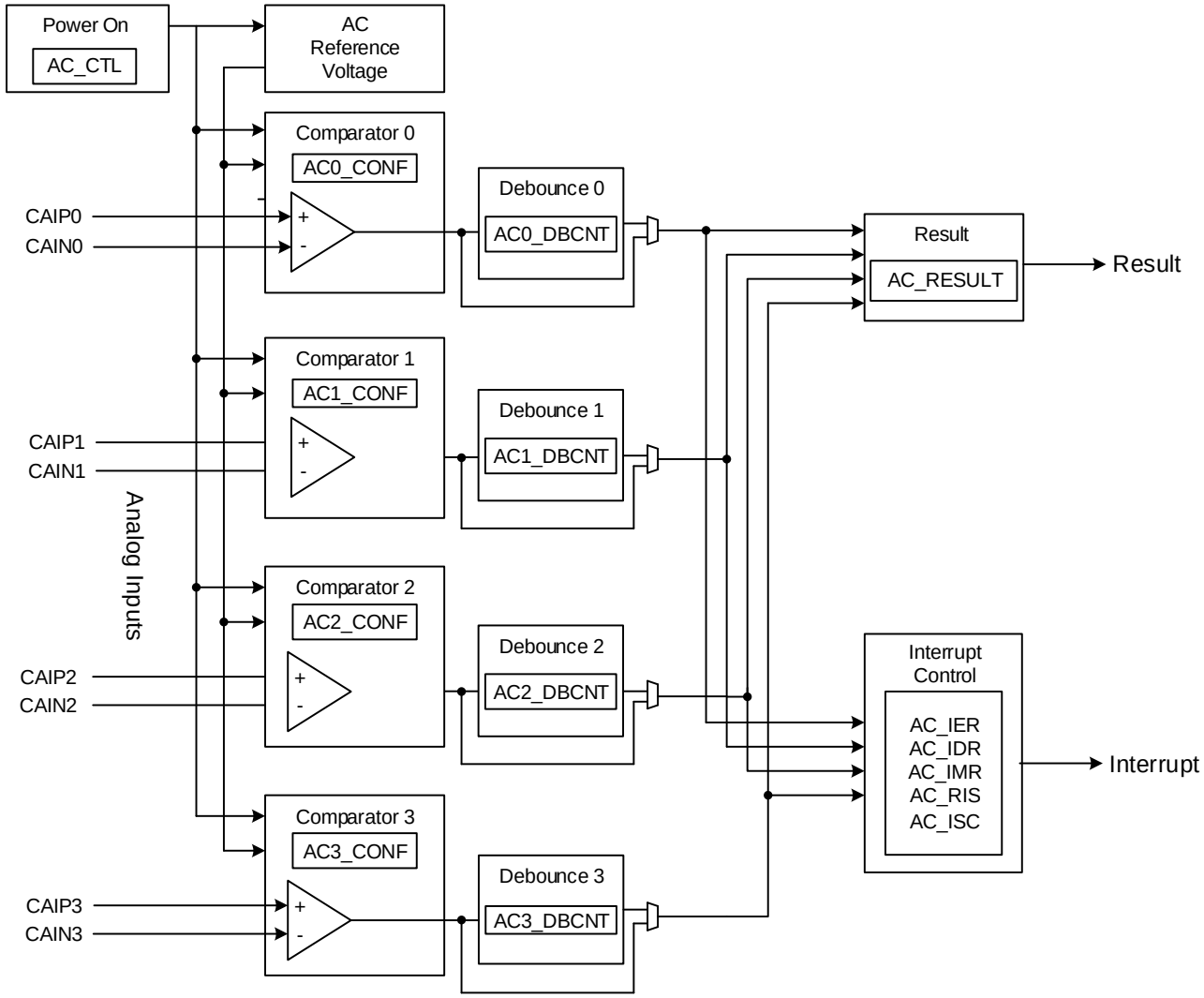
4.11 ANALOG COMPARATOR (AC)

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

4.11.1 BLOCK DIAGRAM

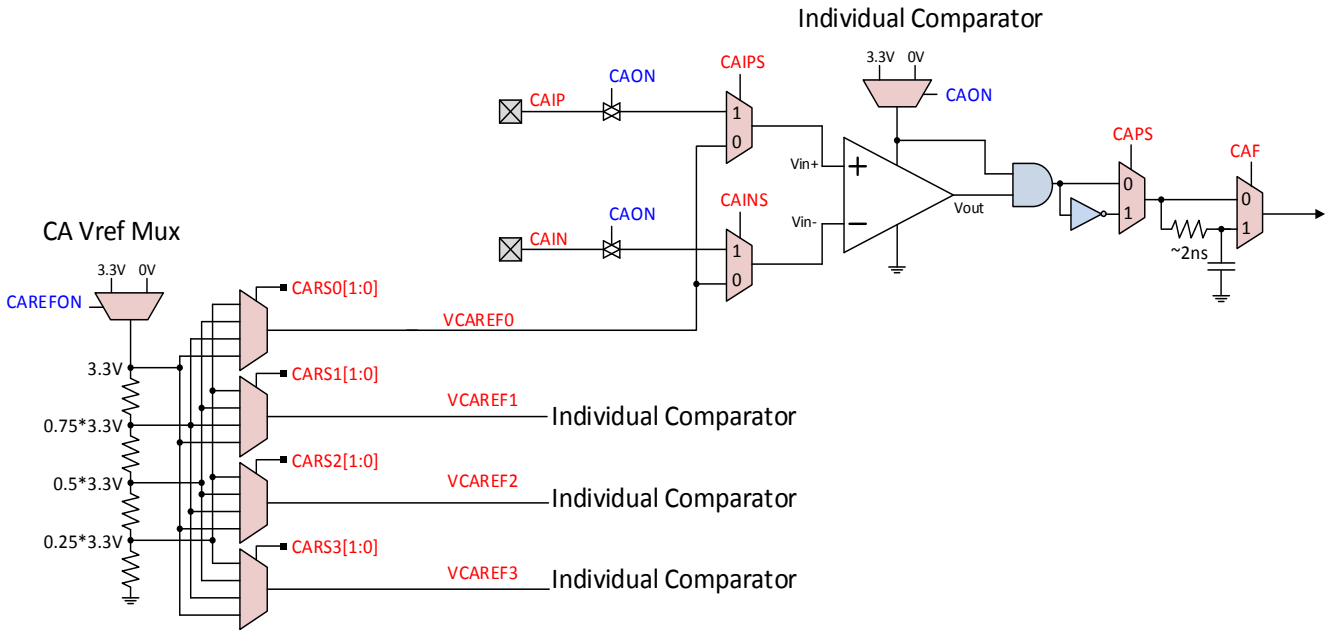
Figure 4.11-1: Analog Comparator Block Diagram



4.11.2 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 4.11-2: comparator unit



4.11.2.1 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.

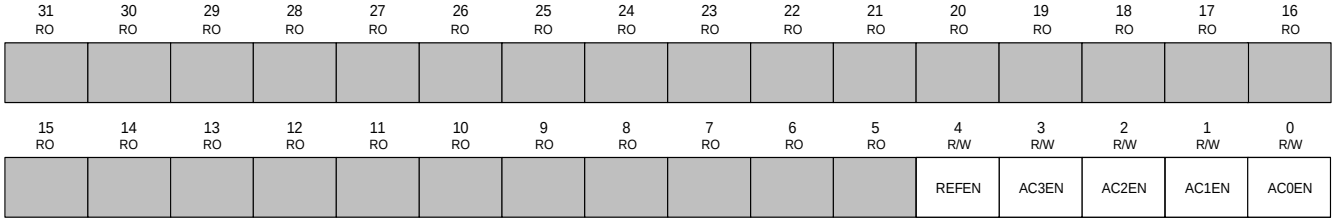
4.11.3 COMPARATOR REGISTER MAP

Base Address: 0x4001_C000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	AC_CTRL	R/W	0x0000_0000	AC Control Register	217
0x0004	AC0_CONF	R/W	0x0000_0000	AC0 Configuration Register	218
0x0008	AC0_BCNT	R/W	0x0000_0000	AC0 Debounce Register	219
0x000C	AC1_CONF	R/W	0x0000_0000	AC1 Configuration Register	218
0x0010	AC1_BCNT	R/W	0x0000_0000	AC1 Debounce Register	219
0x0014	AC2_CONF	R/W	0x0000_0000	AC2 Configuration Register	218
0x0018	AC2_BCNT	R/W	0x0000_0000	AC2 Debounce Register	219
0x001C	AC3_CONF	R/W	0x0000_0000	AC3 Configuration Register	218
0x0020	AC3_BCNT	R/W	0x0000_0000	AC3 Debounce Register	219
0x0024	AC_RESULT	RO	0x0000_0000	AC Output Result	220
0x0028	AC_IER	WO	0x0000_0000	AC Interrupt Enable Register	221
0x002C	AC_IDR	WO	0x0000_0000	AC Interrupt Disable Register	221
0x0030	AC_IMR	RO	0x0000_0000	AC Interrupt Mask Status	222
0x0034	AC_RIS	RO	0x0000_0000	AC Raw Interrupt Status	223
0x0038	AC_ISC	R/W1C	0x0000_0000	AC Interrupt Status and Clear	224

4.11.3.1 AC_CTL – AC CONTROL REGISTER

This register controls the comparator is valid. Each comparator can be individually enabled/disabled.

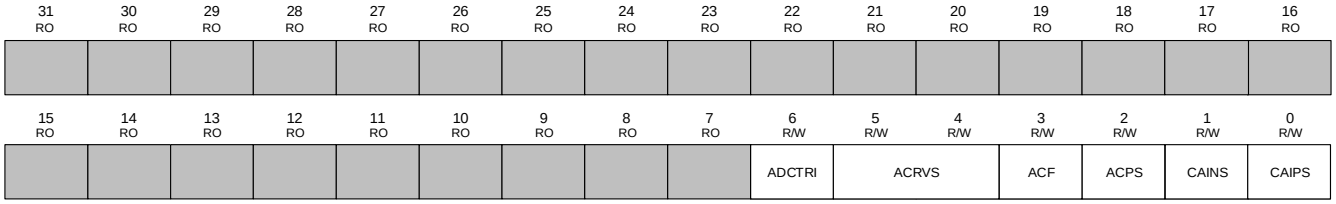


Offset: 0x0000

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.

4.11.3.2 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
31:7	*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	*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.

4.11.3.3 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.

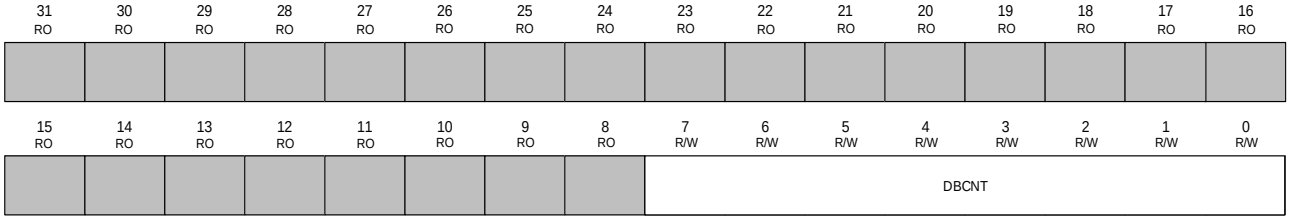
DBCNT: "0x0000_1111" After more than 1 MHz signal is ignored, and check the input.

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

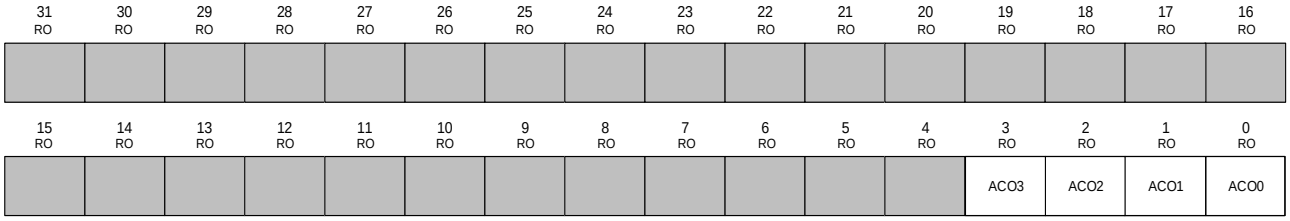
AC2_DBCNT: 0x0018

AC3_DBCNT: 0x0020

Bits	Name	Type	Reset	Description
31:8	*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	*DBCNT*	R/W	0x0	Comparator Debounce Counter Value

4.11.3.4 AC_RESULT – AC OUTPUT RESULT

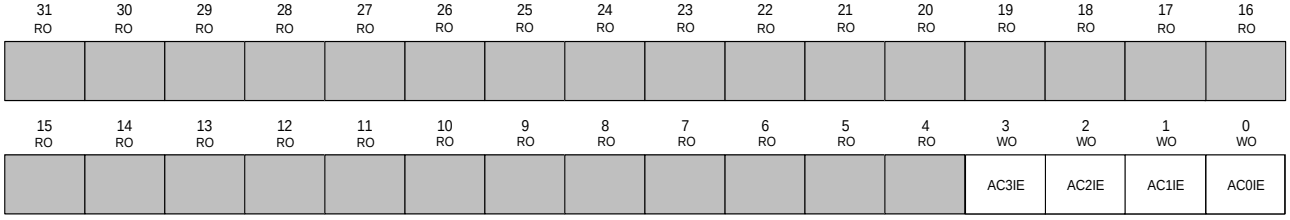
Reading this register to get the value of the comparator output.



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 -

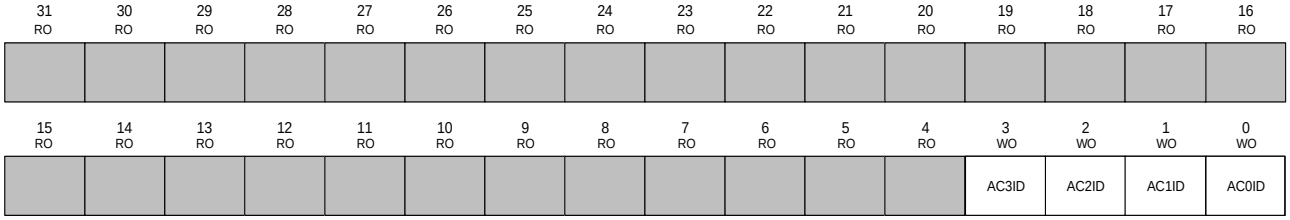
4.11.3.5 AC_IER - AC INTERRUPT ENABLE REGISTER



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.

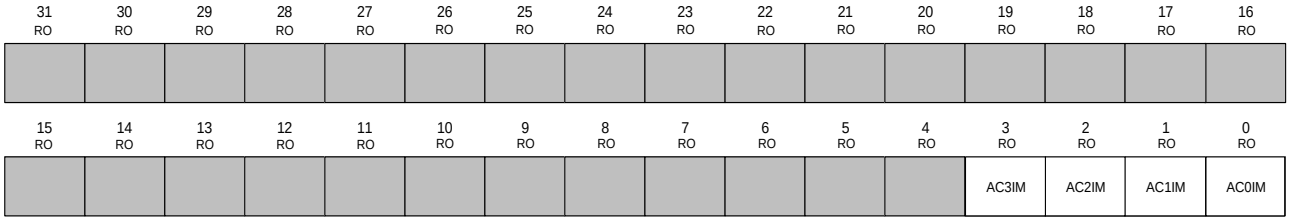
4.11.3.6 AC_IDR - AC INTERRUPT DISABLE REGISTER



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.

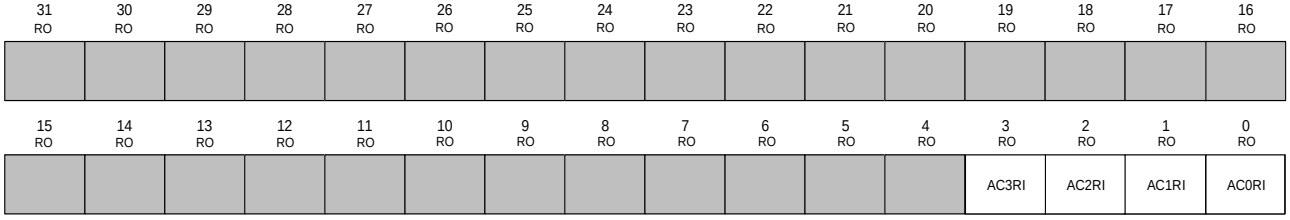
4.11.3.7 AC_IMR - AC INTERRUPT MASK STATUS



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.

4.11.3.8 AC_RIS - AC RAW INTERRUPT STATUS

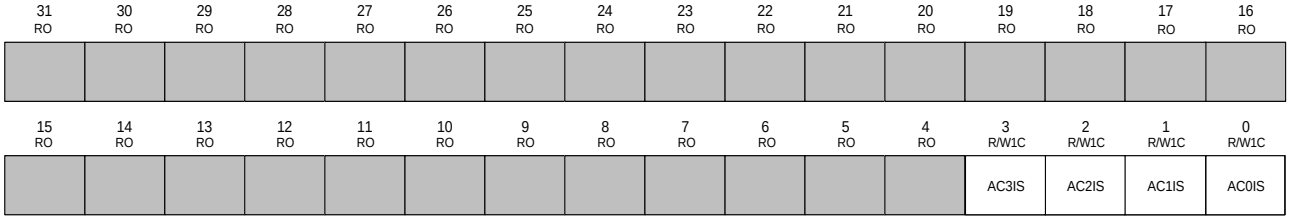


Offset: 0x0034

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.

4.11.3.9 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.

4.12 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.

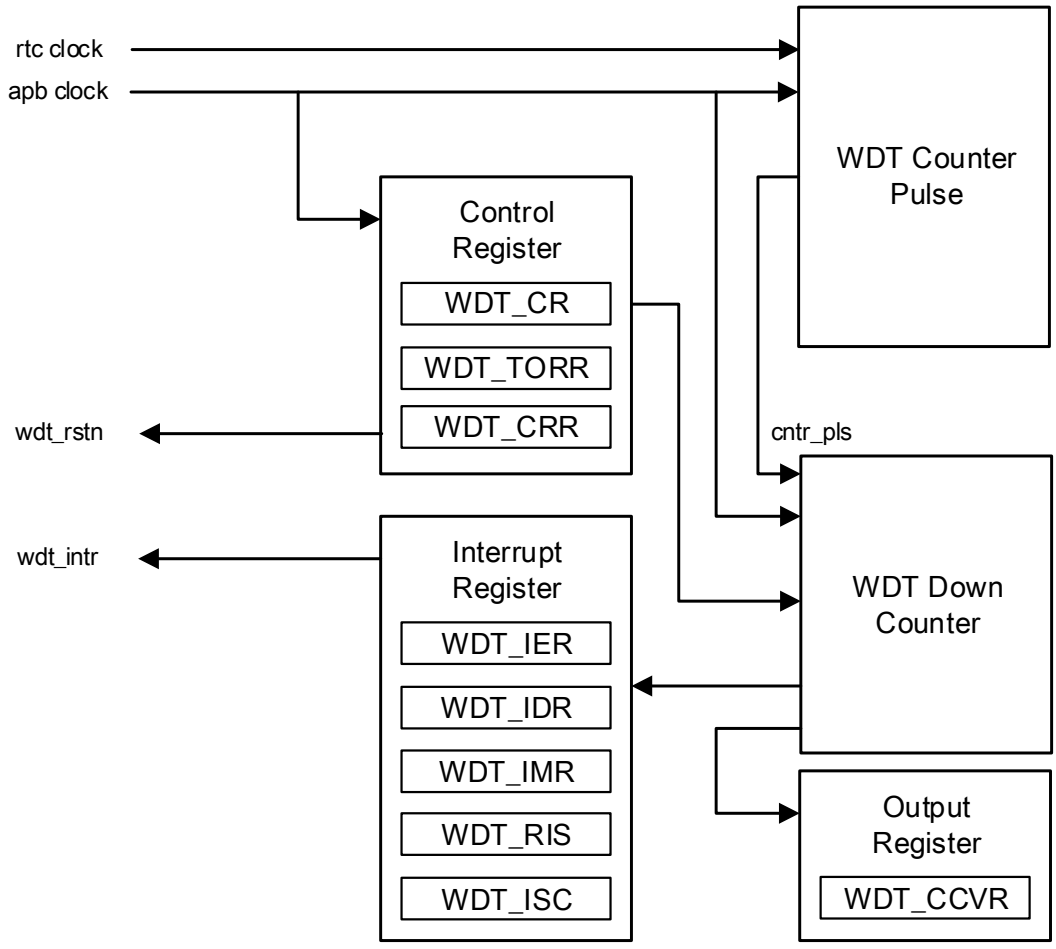
32-bit free-running WDT counter for the watchdog timer timeout interval.

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.

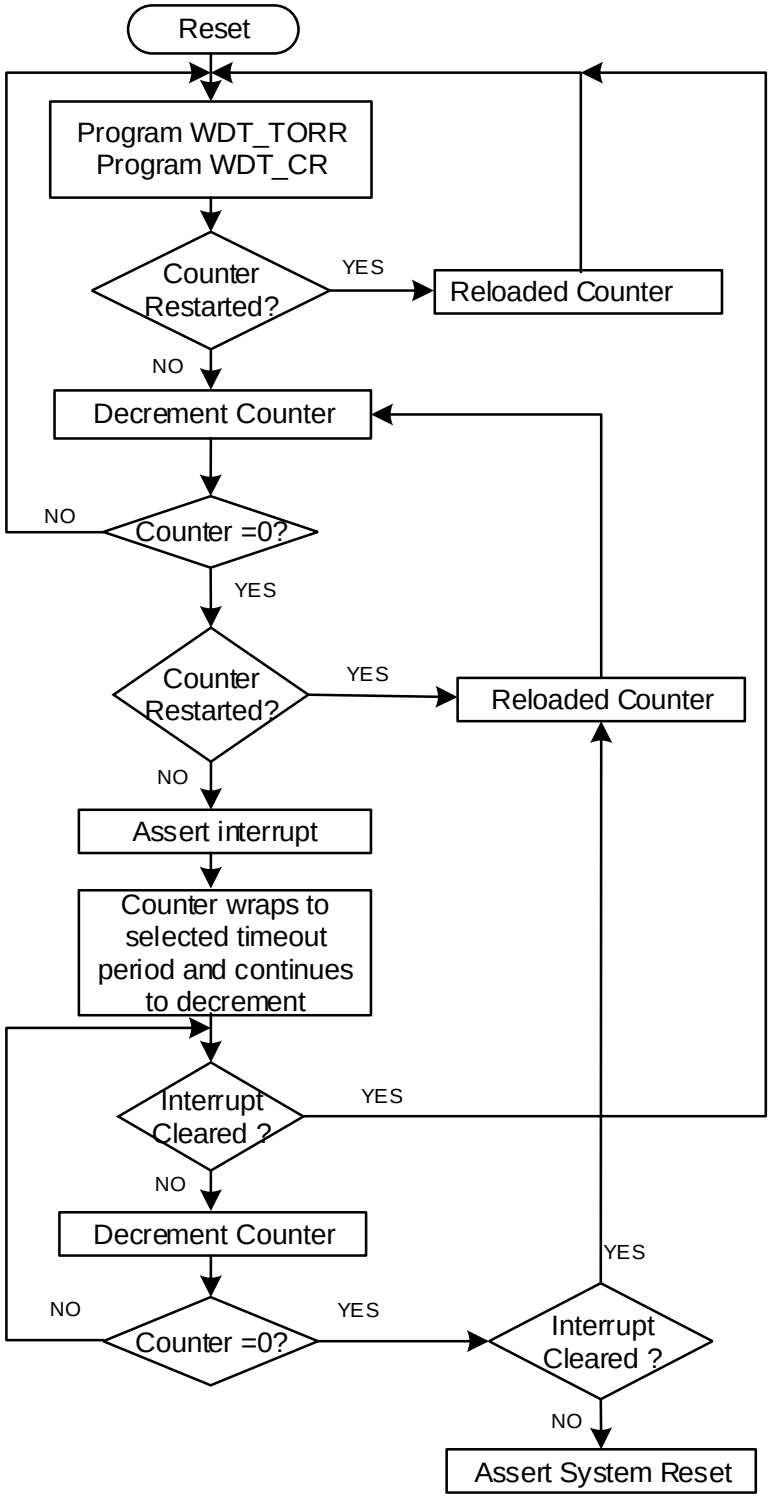
4.12.1 BLOCK DIAGRAM

Figure 4.12-1: WDT Block Diagram



4.12.2 OPERATING FLOWCHART

Figure 4.12-2: WDT Operating Flowchart



4.12.3 FUNCTIONAL DESCRIPTION

4.12.3.1 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)

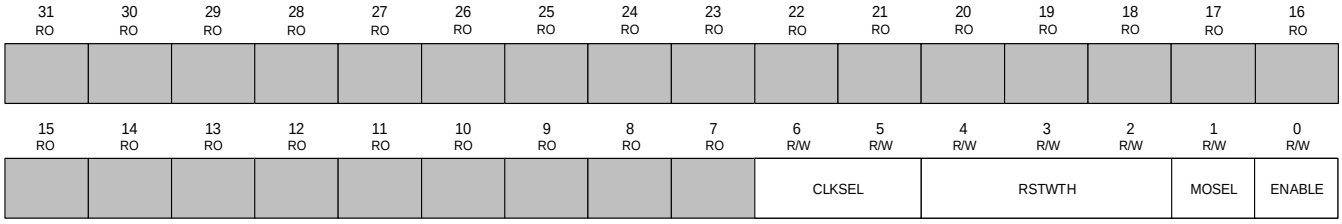
WDT Interrupt Status & Interrupt Clear Register (WDT_ISC) is used to read the masked interrupt status of the module, showing which interrupt is unmasked. Each bit in ISC is the logical AND of the respective bits in RIS and IMR. Writing a '1' to the bit in this register can clear the corresponding interrupt status but not the interrupt itself.

4.12.4 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	228
0x0004	WDT_TORR	R/W	0x0000_0000	WDT Timeout Range Register	229
0x0008	WDT_CCVR	RO	0x0000_FFFF	WDT Current Counter Value Register	230
0x000C	WDT_CRR	WO	0x0000_0000	WDT Counter Restart Register	230
0x0010	WDT_IER	WO	0x0000_0000	WDT Interrupt Enable Register	230
0x0014	WDT_IDR	WO	0x0000_0000	WDT Interrupt Disable Register	231
0x0018	WDT_IMR	RO	0x0000_0000	WDT Interrupt Mask Register	231
0x001C	WDT_RIS	RO	0x0000_0000	WDT Raw Interrupt Status Register	232
0x0020	WDT_ISC	R/W1C	0x0000_0000	WDT Interrupt Status and Clear Register	232

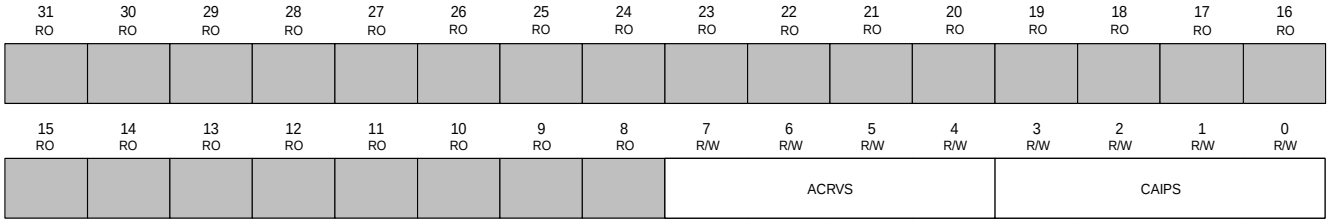
4.12.4.1 WDT_CTRL - WDT CONTROL REGISTER



Offset: 0x0000

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.
6:5	*CLKSEL*	R/W	0	WDT Clock Source Select 0: APB clock 1: -
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: Two phase mode is implemented 1: One 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.

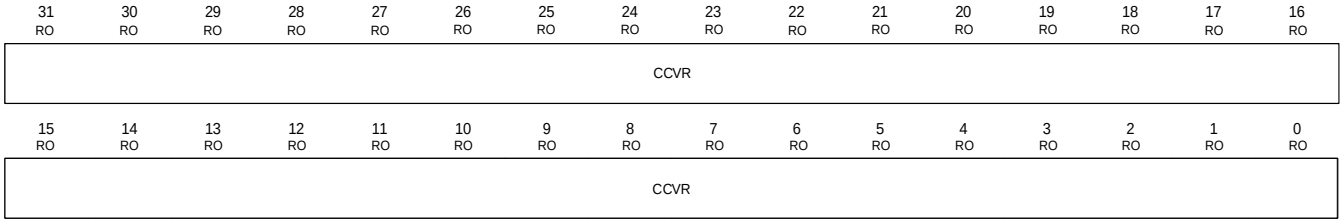
4.12.4.2 WDT_TORR - WDT TIMEOUT RANGE REGISTER



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.
7:4	*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

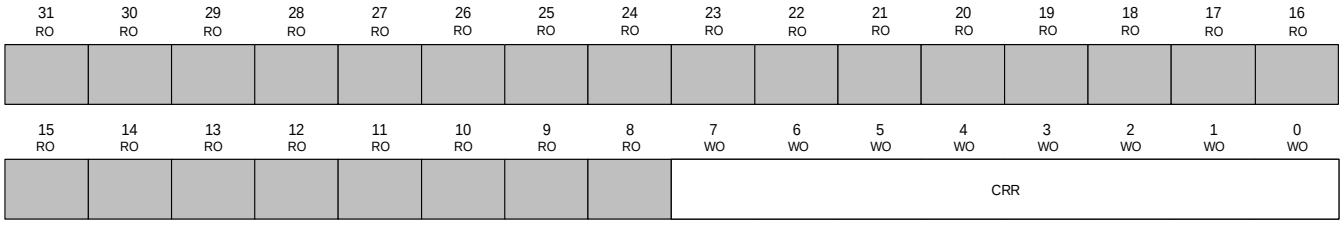
4.12.4.3 WDT_CCVR - WDT CURRENT COUNTER VALUE REGISTER



Offset: 0x0008

Bits	Name	Type	Reset	Description
31:0	*CCVR*	RO	0xFFFF	WDT Current Counter Value The current value of the internal counter.

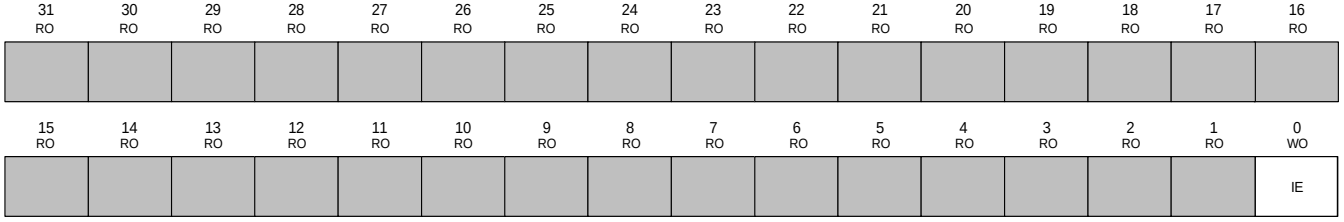
4.12.4.4 WDT_CRR - WDT COUNTER RESTART REGISTER



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.
7:0	*CRR*	WO	0x0	WDT Counter Restart This register is used to restart the WDT counter.

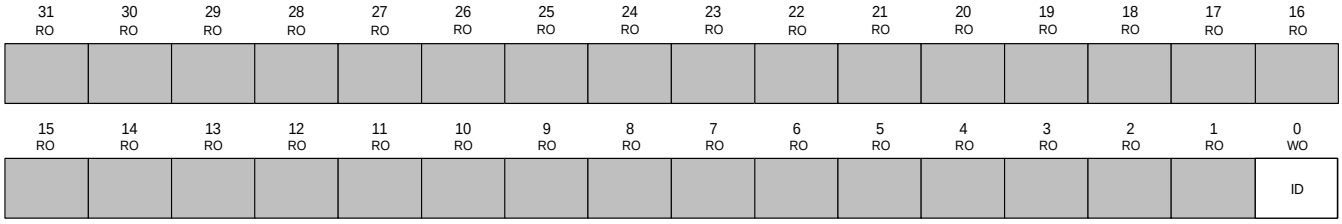
4.12.4.5 WDT_IER - WDT INTERRUPT ENABLE REGISTER



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	*IE*	WO	0	WDT Interrupt Enable 1: Interrupt is enabled.

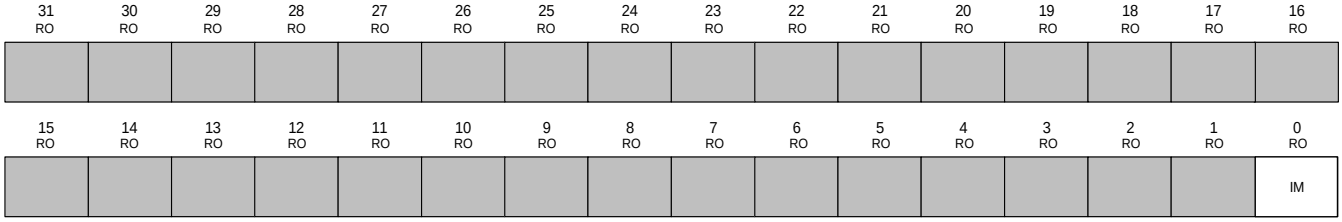
4.12.4.6 WDT_IDR - WDT INTERRUPT DISABLE REGISTER



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	*ID*	WO	0	WDT Interrupt Disable 1: Interrupt is disabled.

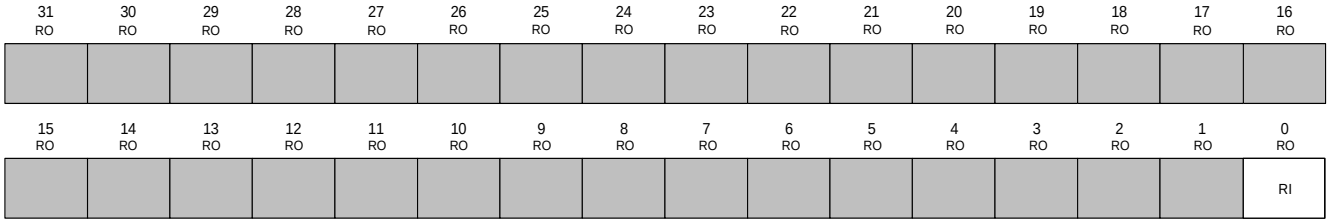
4.12.4.7 WDT_IMR - WDT INTERRUPT MASK REGISTER



Offset: 0x0018

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	*IM*	RO	0	WDT Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.

4.12.4.8 WDT_RIS - WDT RAW INTERRUPT STATUS

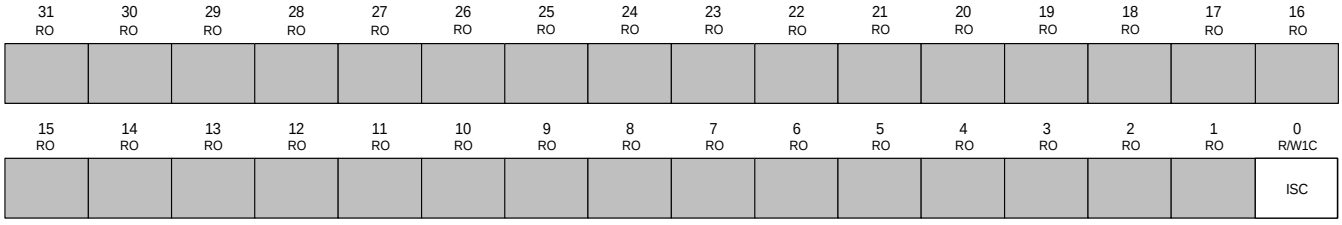


Offset: 0x001C

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	*RI*	RO	0	WDT Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.

4.12.4.9 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*	RO	0x0	Software should not 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.

4.13 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.

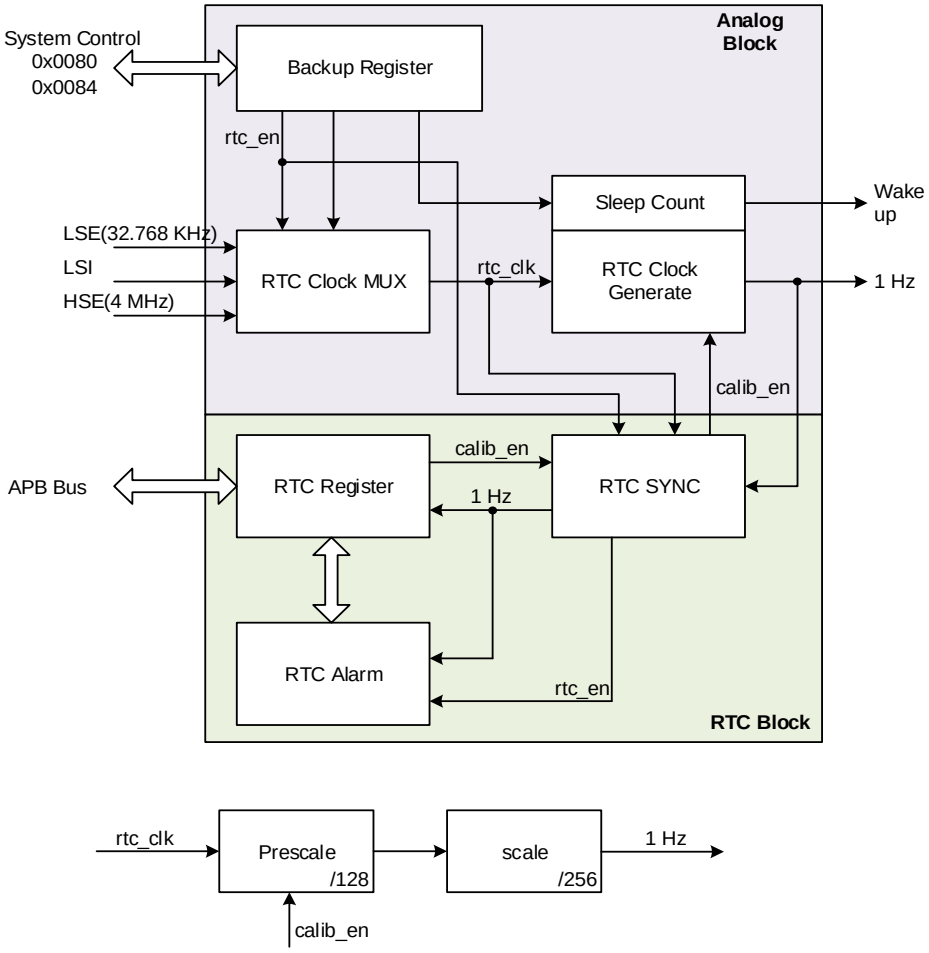
The clock source of RTC is from a 32.768 kHz Low Speed External crystal or ceramic resonator, LSI RC or PLL reference clock source divided by 128.

The RTC controller provides time message (hour, minute, second) as well as calendar message (century, 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.

- Calendar with century, 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: century, year, month, day, week, hour, minute and second.
- Automatic leap year compensation.
- All time and calendar message is expressed in BCD code.
- Rollover interrupts: rollover interrupts of century, year, month, day, week, hour, minute, second.
- Frequency compensative: clock deviation can be compensated by adjusting counter
- Backup Register

4.13.1 BLOCK DIAGRAM

Figure 4.13-1: RTC Block Diagram



4.13.2 FUNCTION DESCRIPTION

4.13.2.1 RTC ENABLE

When user want to use the RTC function, the enable is in the system controller register SC_BKRTC_CTRL. There can set the RTC enable, reset, prescale value, scale value, calibration value and calibration decrease /increase. For example: we set the RTC enable = 1, prescale = 0x7F and scale = 0xFF. At this configure, the timer will count 32768 times, than let the seconds increase one.

4.13.2.2 RTC TIME AND CALENDAR MESSAGE INITIALIZATION AND CONFIGURATION

When user want to change the RTC Time and Calendar message, user should set the RTC_CTRL register to 1, it means start written the data to the timer. Then user can setting the data to the RTC_TIME & RTC_CAL register. When user is setting done, set the RTC_CTRL register to 0, it means the timer can continue to count.

4.13.2.3 FREQUENCY COMPENSATIVE

The RTC clock source may not precise to exactly 32768 Hz and the RTC allows software to make digital compensation to the RTC source clock. Following are the compensation examples for higher or lower frequency clock input.

Example 1:

Frequency counter measurement: 32773.65 Hz (> 32768 Hz)

Integer part: 32773

FCR.Integer = 32773 - 32768 = 5, every 1 second compensate +5.

Fraction part: 0.65

FCR.Fraction = 0.65 * 60 = 39, every 60 second compensate +39.

The Register Configure as follows:

Step 1: Set 1 to the CALIBEN & CYCSEL in the RTC_CALIB register.

Step 2: Configure the SC_BKRTC_CTRL register.

Set 0x1 at COUNTEN to enable the RTC counting;

Set 0x1 at CKSEL to select the crystal;

Set 0x7F & 0xFF at CLKSCAL & CLKPSCAL to let RTC count 32768 Hz;

Set 0x0 & 0x5 at CRS & CRV, it means every 1 second, the RTC count value is 32773(32768+5).

Step 3: Every 60 second, change the CRV value from 0x5 to 0x2C (+5+39=44). At the other times, the CRV value is 0x5. This operation is want to compensate the fraction part error.

Example 2:

Frequency counter measurement: 32765.27 Hz (≤ 32768 Hz)

Integer part: 32765

FCR.Integer = 32768 - 32765 = -3, every 1 second compensate -3.

Fraction part: 0.27

FCR.Fraction = 0.27 * 60 = 16.2, every 60 second compensate +16.

The Register Configure as follows:

Step 1: Set 1 to the *CALIBEN* & *CYCSEL* in the **RTC_CALIB** register.

Step 2: Configure the **SC_BKRTC_CTRL** register.

Set 0x1 at *COUNTEN* to enable the RTC counting;

Set 0x1 at *CKSEL* to select the crystal;

Set 0x7F & 0xFF at *CLKSCAL* & *CLKPSCAL* to let RTC count 32768 Hz;

Set 0x1 & 0x3 at *CRS* & *CRV*, it means every 1 second, the RTC count value is 32765(32768-3).

Step 3 : Every 60 second, change the *CRS* & *CRV* value from 0x1 & 0x3 to 0x0 & 0xD(-3+16=13). At the other times, the *CRS* & *CRV* value is 0x1 & 0x3. This operation is want to compensate the fraction part error.

4.13.2.4 DAY OF THE WEEK COUNTER

The RTC controller provides day of week in **RTC_TIME** register. The value is define from 1 to 7 to represent Monday to Sunday respectively.

4.13.2.5 ALARM INTERRUPT

When alarm enable and RTC time in **RTC_TIME** & **RTC_CAL** equal to alarm setting time in **RTC_ALMTIME** & **RTC_ALMCAL**, the alarm interrupt flag (RTC_RIS [8:0]) is set and the alarm interrupt is requested if the alarm interrupts is enabled.

Periodic Alarm Interrupt:

If open the alarm in **RTC_ALMEN** and enable the interrupt in **RTC_IER**. Every time equal to alarm setting, the interrupt will be set. For example: Setting the alarm enable to be 0x2 at **RTC_ALMEN**, minute data 0x50 at **RTC_ALMTIME**, and enable the minute interrupt at **RTC_IER**, than every 50 minute 0 second, interrupt will be set.

All Alarm Interrupt:

If open all alarm in **RTC_ALMEN** and enable the all alarm interrupt in **RTC_IER**. Only at the time in **RTC_TIME & RTC_CAL** equal to alarm setting time in **RTC_ALMTIME** & **RTC_ALMCAL**, the all alarm interrupt will be set. For example: Set the alarm enable is 0xFF at **RTC_AREN**, calendar 0x20150808 at **RTC_ALMCAL**, time 0x06173000 at **RTC_ALMTIME**, and enable the all alarm interrupt at **RTC_IER**, then only at time is 2015/08/08, Saturday, 17:30:00, the interrupt will be set.

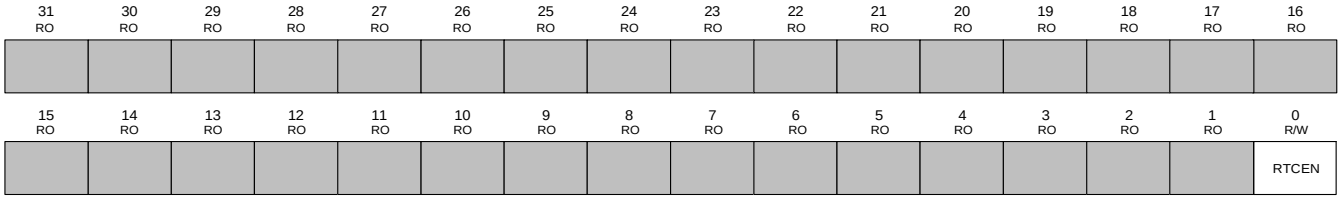
4.13.3 RTC REGISTER MAP

Base Address: 0x4000_8000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	RTC_CTRL	R/W	0x00FF_7F00	RTC Control Register	237
0x0004	RTC_TIME	R/W	0x0000_0000	RTC Calendar Time Register	238
0x0008	RTC_CAL	R/W	0x0000_0000	RTC Calendar Date Register	239
0x000C	RTC_ALMTIME	R/W	0x0000_0000	RTC Alarm Time Register	240
0x0010	RTC_ALMCAL	R/W	0x0000_0000	RTC Alarm Calendar Register	241
0x0014	RTC_ALMEN	R/W	0x0000_0000	RTC Alarm Enable Set Register	242
0x0018	RTC_CALIB	R/W	0x0000_0000	RTC Calibration Set Register	243
0x001C	RTC_TRIG	R/W	0x0000_0000	RTC Trigger Select Register	244
0x0020	RTC_IER	WO	0x0000_0000	RTC Interrupt Enable Register	246
0x0024	RTC_IDR	WO	0x0000_0000	RTC Interrupt Disable Register	248
0x0028	RTC_IMR	RO	0x0000_0000	RTC Interrupt Mask Register	250
0x002C	RTC_RIS	RO	0x0000_0000	RTC Raw Interrupt Status Register	252
0x0030	RTC_ISC	R/W1C	0x0000_0000	RTC Interrupt Status and Clear Register	254

4.13.3.1 RTC_CTRL - RTC CONTROL REGISTER

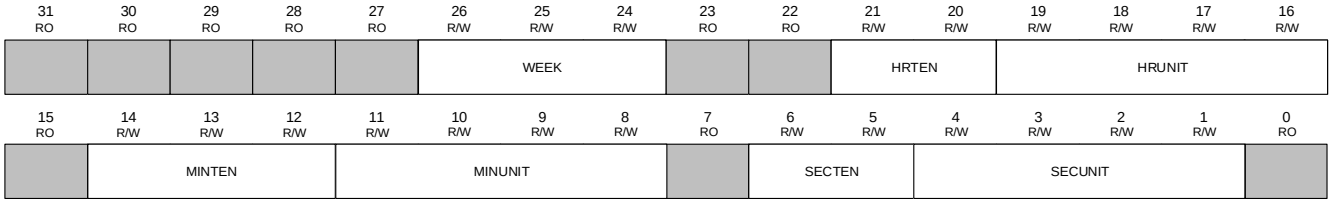
This register is used to temporarily disable any updates in RTC timer and calendar while writing registers in RTC.



Offset: 0x0000

Bits	Name	Type	Reset	Description
31:1	*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.
0	*RTCEN*	R/W	0	RTC Initialization 0: RTC counter is disabled. 1: RTC counter is enabled.

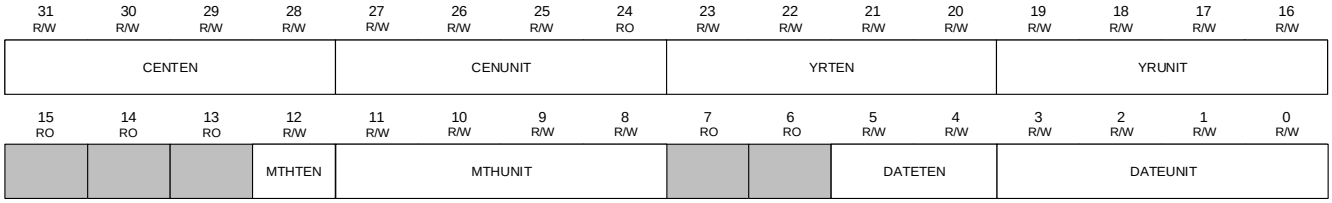
4.13.3.2 RTC_TIME - RTC TIME SET REGISTER AND VALUE



Offset: 0x0004

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.

4.13.3.3 RTC_CAL - RTC CALENDAR SET REGISTER AND VALUE

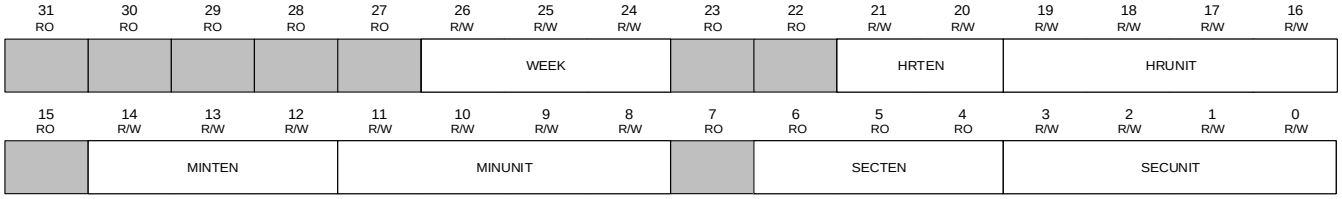


Offset: 0x0008

Bits	Name	Type	Reset	Description
31:28	*CENTEN*	R/W	0x0	Century Ten Century tens in BCD format.
27:24	*CENUNIT*	R/W	0x0	Century Unit Century units in BCD format.
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.

4.13.3.4 RTC_ALMTIME - RTC Alarm Time Set Register

This register is used along with **RTC_ARCAL** to set the RTC alarm.

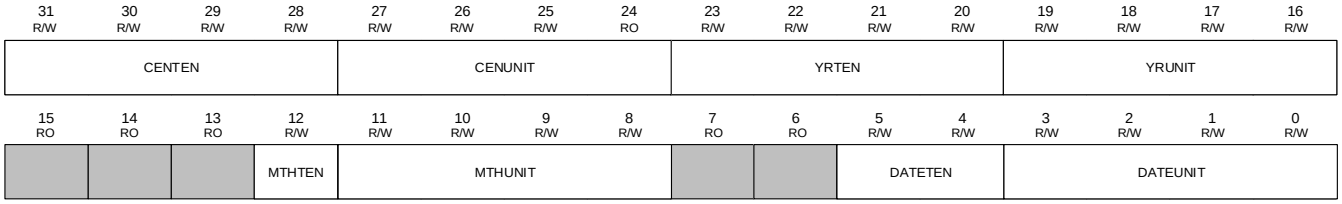


Offset: 0x000C

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.

4.13.3.5 RTC_ALMCAL - RTC Alarm Calendar Set Register

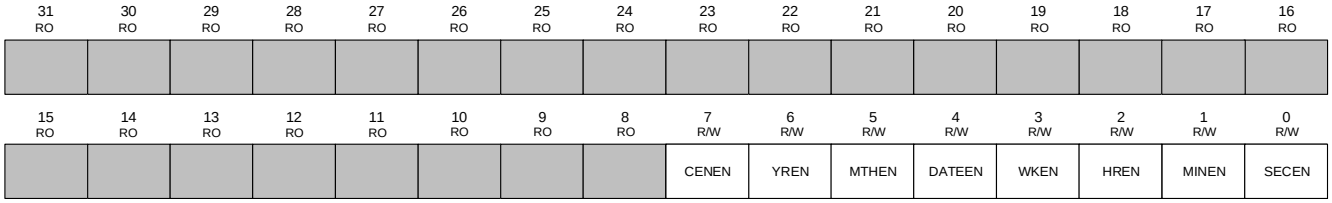
This register is used along with **RTC_ARTIME** to set the RTC alarm.



Offset: 0x0010

Bits	Name	Type	Reset	Description
31: 28	*CENTEN*	R/W	0x0	Alarm Century Ten Century tens in BCD format.
27:24	*CENUNIT*	R/W	0x0	Alarm Century Unit Century units in BCD format.
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*	RO	0x0	Software should not 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*	RO	0x0	Software should not 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.

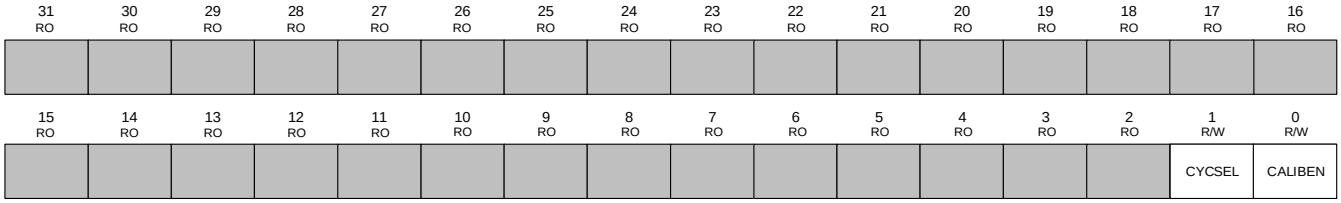
4.13.3.6 RTC_ALMEN - RTC Alarm Enable Set Register



Offset: 0x0014

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	*CENEN*	R/W	0	Alarm Function of Century Enable 0: Alarm function of century is disabled. 1: Alarm function of century is enabled
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

4.13.3.7 RTC_CALIB - RTC Calibration Set Register



Offset: 0x0018

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	*CYCSEL*	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

4.13.3.8 RTC_TRIG - RTC Trigger Select Register

RTC_TRIG register is used to enable individual bit for triggering the ADC.

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	R/W	R/W
														1HZFLAG	RCEN

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	RO	R/W
RYR	RMTH	RDATE	RWK	RHR	RMIN	RSEC	AALL	ACEN	AYR	AMTH	ADATE	AWK	AHR	AMIN	ASEC

Offset: 0x001C

Bits	Name	Type	Reset	Description
31: 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	*1HZFLAG*	R/W	0	RTC Trigger 1Hz Flag This bit is to control the output of signal RTC_CLK. 1: 1Hz output ADC trigger is enabled.
16	*RCEN*	R/W	0	Rollover Century Enable 1: Century rollover ADC trigger is enabled.
15	*RYR*	R/W	0	Rollover Year Enable 1: Year rollover ADC trigger is enabled.
14	*RMTH*	R/W	0	Rollover Month Enable 1: Month rollover ADC trigger is enabled.
13	*RDATE*	R/W	0	Rollover Date Enable 1: Date rollover ADC trigger is enabled.
12	*RWK*	R/W	0	Rollover Week Enable 1: Week rollover ADC trigger is enabled.
11	*RHR*	R/W	0	Rollover Hour Enable 1: Hour rollover ADC trigger is enabled.
10	*RMIN*	R/W	0	Rollover Minute Enable 1: Minute rollover ADC trigger is enabled.
9	*RSEC*	R/W	0	Rollover Second Enable 1: Second rollover ADC trigger is generated.
8	*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.
7	*ACEN*	R/W	0	Century Alarm Enable 1: ADC trigger is generated when the alarm century value is matched.
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

Bits	Name	Type	Reset	Description
				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.

4.13.3.9 RTC_IER - RTC Interrupt Enable Register

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 WO	16 WO
														1HZFLAGIE	ROLCENIE

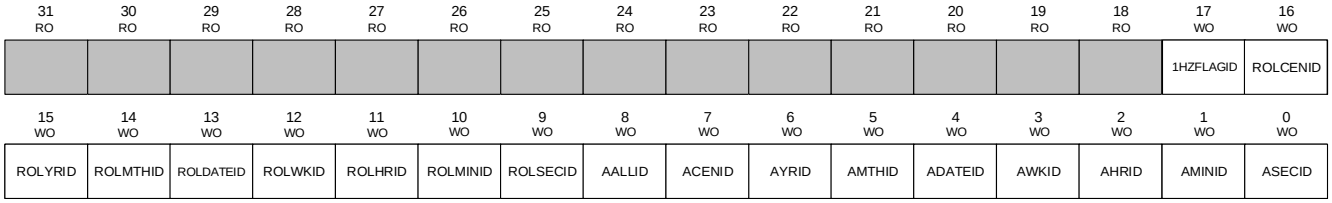
15 WO	14 WO	13 WO	12 WO	11 WO	10 WO	9 WO	8 WO	7 WO	6 WO	5 WO	4 WO	3 WO	2 WO	1 WO	0 WO
ROLYRIE	ROLMTHIE	ROLDATEIE	ROLWKIE	ROLHRIE	ROLMINIE	ROLSECIE	AALLIE	ACENIE	AYRIE	AMTHIE	ADATEIE	AWKIE	AHRIE	AMINIE	ASECIE

Offset: 0x0020

Bits	Name	Type	Reset	Description
31: 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	*1HZFLAGIE*	WO	0	RTC Trigger 1Hz Flag Interrupt Enable This bit is to control the output of signal RTC_CLK 1: 1Hz output interrupt is enabled.
16	*ROLCENIE*	WO	0	Rollover Century Interrupt Enable 1: Century rollover interrupt is enabled.
15	*ROLYRIE*	WO	0	Rollover Year Interrupt Enable 1: Year rollover interrupt is enabled.
14	*ROLMTHIE*	WO	0	Rollover Month Interrupt Enable 1: Month rollover interrupt is enabled.
13	*ROLDATEIE*	WO	0	Rollover Date Interrupt Enable 1: Date rollover interrupt is enabled.
12	*ROLWKIE*	WO	0	Rollover Week Interrupt Enable 1: Week rollover interrupt is enabled.
11	*ROLHRIE*	WO	0	Rollover Hour Interrupt Enable 1: Hour rollover interrupt is enabled.
10	*ROLMINIE*	WO	0	Rollover Minute Interrupt Enable 1: Minute rollover interrupt is enabled.
9	*ROLSECIE*	WO	0	Rollover Second Interrupt Enable 1: Second rollover interrupt is enabled.
8	*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.
7	*ACENIE*	WO	0	Century Alarm Interrupt Enable 1: Interrupt is enabled when the alarm century value is matched.
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.

Bits	Name	Type	Reset	Description
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.

4.13.3.10 RTC_IDR - RTC Interrupt Disable Register



Offset: 0x0024

Bits	Name	Type	Reset	Description
31: 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	*1HZFLAGID*	WO	0	RTC Trigger 1Hz Flag Interrupt Disable This bit is to control the output of signal RTC_1HZ. 1: 1Hz output interrupt is disabled.
16	*ROLCENID*	WO	0	Rollover Century Interrupt Disable 1: Century rollover interrupt is disabled.
15	*ROLYRID*	WO	0	Rollover Year Interrupt Disable 1: Year rollover interrupt is disabled.
14	*ROLMTHID*	WO	0	Rollover Month Interrupt Disable 1: Month rollover interrupt is disabled.
13	*ROLDATEID*	WO	0	Rollover Date Interrupt Disable 1: Date rollover interrupt is disabled.
12	*ROLWKID*	WO	0	Rollover Week Interrupt Disable 1: Week rollover interrupt is disabled.
11	*ROLHRID*	WO	0	Rollover Hour Interrupt Disable 1: Hour rollover interrupt is disabled.
10	*ROLMINID*	WO	0	Rollover Minute Interrupt Disable 1: Minute rollover interrupt is disabled.
9	*ROLSECID*	WO	0	Rollover Second Interrupt Disable 1: Second rollover interrupt is disabled.
8	*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.
7	*ACENID*	WO	0	Century Alarm Interrupt Disable 1: Interrupt is disabled when the alarm century value is matched.
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.

Bits	Name	Type	Reset	Description
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.

4.13.3.11 RTC_IMR - RTC Interrupt Mask Register

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
														1HZFLAGIM	ROLCENIM

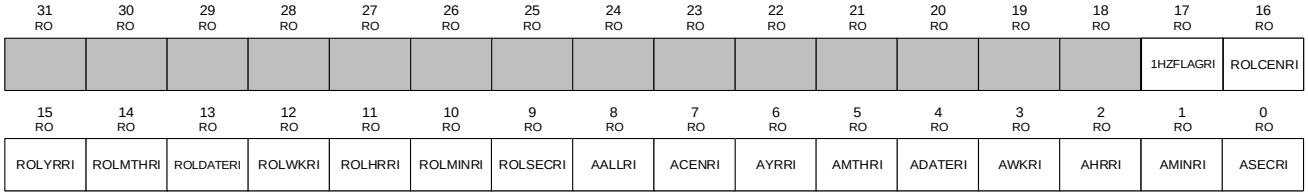
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ROLYRIM	ROLMTHIM	ROLDATEIM	ROLWKIM	ROLHRIM	ROLMINIM	ROLSECIM	AALLIM	ACENIM	AYRIM	AMTHIM	ADATEIM	AWKIM	AHRIM	AMINIM	ASECIM

Offset: 0x0028

Bits	Name	Type	Reset	Description
31: 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	*1HZFLAGIM*	RO	0	RTC Trigger 1Hz Flag Interrupt Mask Status 0: Interrupt will be masked. 1: 1Hz output interrupt will not be masked.
16	*RCENIM*	RO	0	Rollover Century Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
15	*ROLYRIM*	RO	0	Rollover Year Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
14	*ROLMTHIM*	RO	0	Rollover Month Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
13	*ROLDATEIM*	RO	0	Rollover Date Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
12	*ROLWKIM*	RO	0	Rollover Week Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
11	*ROLHRIM*	RO	0	Rollover Hour Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
10	*ROLMINIM*	RO	0	Rollover Minute Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
9	*ROLSECIM*	RO	0	Rollover Second Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
8	*AALLIM*	RO	0	Alarm All Interrupt Mask Status 0: Interrupt will be masked. 1: Interrupt will not be masked.
7	*ACENIM*	RO	0	Century Alarm 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.

Bits	Name	Type	Reset	Description
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.
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.

4.13.3.12 RTC_RIS - RTC Raw Interrupt Status Register



Offset: 0x002C

Bits	Name	Type	Reset	Description
31: 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	*1HZFLAGRI*	RO	0	RTC Trigger 1Hz Flag Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
16	*ROLCENRI*	RO	0	Rollover Century Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
15	*ROLYRRI*	RO	0	Rollover Year Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
14	*ROLMTHRI*	RO	0	Rollover Month Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
13	*ROLDATERI*	RO	0	Rollover Date Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
12	*ROLWKRI*	RO	0	Rollover Week Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
11	*ROLHRRI*	RO	0	Rollover Hour Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
10	*ROLMINRI*	RO	0	Rollover Minute Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
9	*ROLSECRI*	RO	0	Rollover Second Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
8	*AALLRI*	RO	0	Alarm All Raw Interrupt Status 0: No interrupt is generated. 1: Interrupt is asserting.
7	*ACENRI*	RO	0	Century Alarm 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.

Bits	Name	Type	Reset	Description
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.
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.

4.13.3.13 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	R/W1C	R/W1C
														1HZFLAGIS	ROLCENIS

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
ROLYRIS	ROLMTHIS	ROLDATEIS	ROLWKIS	ROLHRIS	ROLMINIS	ROLSECIS	AALLIS	ACENIS	AYRIS	AMTHIS	ADATEIS	AWKIS	AHRIS	AMINIS	ASECIS

Offset: 0x0030

Bits	Name	Type	Reset	Description
31: 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	*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.
16	*ROLCENIS*	R/W	0	Rollover Century Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
15	*ROLYRIS*	R/W1C	0	Rollover Year Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
14	*ROLMTHIS*	R/W1C	0	Rollover Month Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
13	*ROLDATEIS*	R/W1C	0	Rollover Date Raw Interrupt Status 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
12	*ROLWKIS*	R/W1C	0	Rollover Week Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
11	*ROLHRIS*	R/W1C	0	Rollover Hour Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
10	*ROLMINIS*	R/W1C	0	Rollover Minute Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
9	*ROLSECIS*	R/W1C	0	Rollover Second Interrupt Status and Clear 0: No interrupt has occurred or the interrupt is masked. 1: Interrupt has been signaled.
8	*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.
7	*ACENIS*	R/W1C	0	Century 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
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.
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.

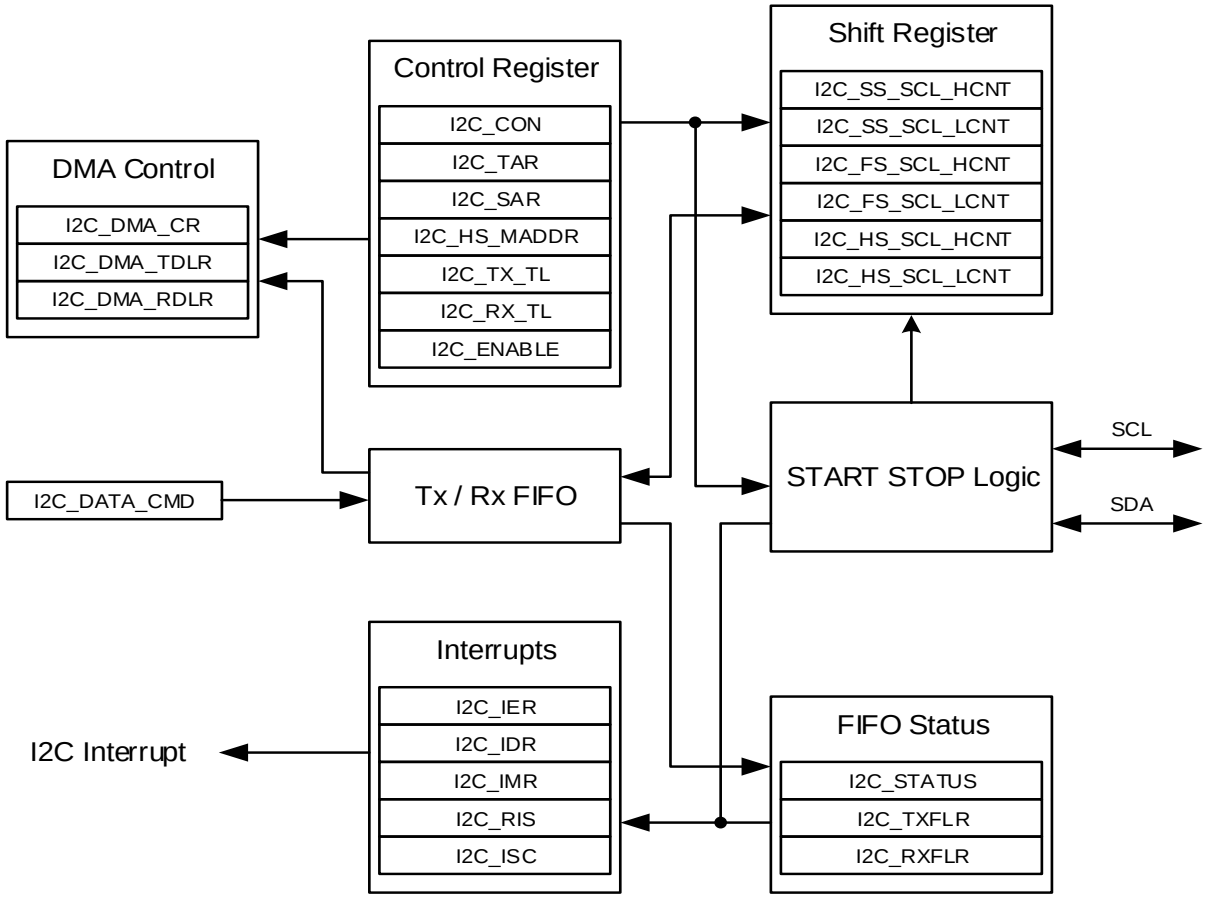
4.14 INTER INTEGRATED CIRCUIT (I2C)

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 : 1 Mb/s
- Master / Slave operation.
- 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.

4.14.1 BLOCK DIAGRAM

Figure 4.14-1: I2C Block Diagram



4.14.2 FUNCTIONAL DESCRIPTION

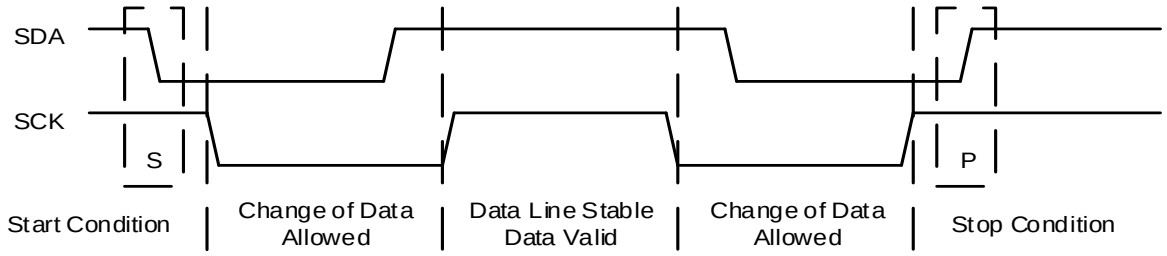
4.14.2.1 I2C Bus Protocol

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.

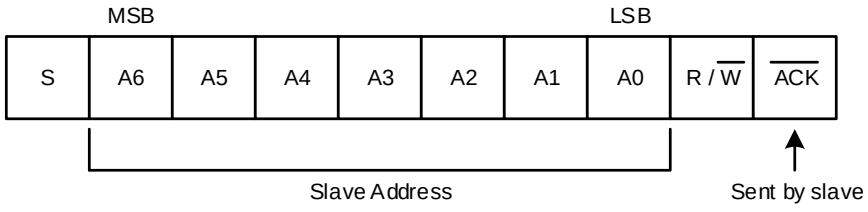
Figure 4.14-2: Start and Stop Condition



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 4.14-3: 7-bit address format

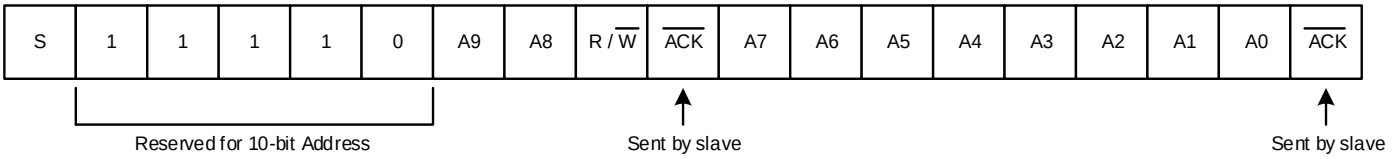


S = start condition

R / $\overline{W}$ = Read / Write Pulse

$\overline{ACK}$ = Acknowledge

Figure 4.14-4: 10-bit address format



S = start condition

R / $\overline{W}$ = Read / Write Pulse

$\overline{ACK}$ = Acknowledge

Table 4.14-1: Definition of Bits in First Byte

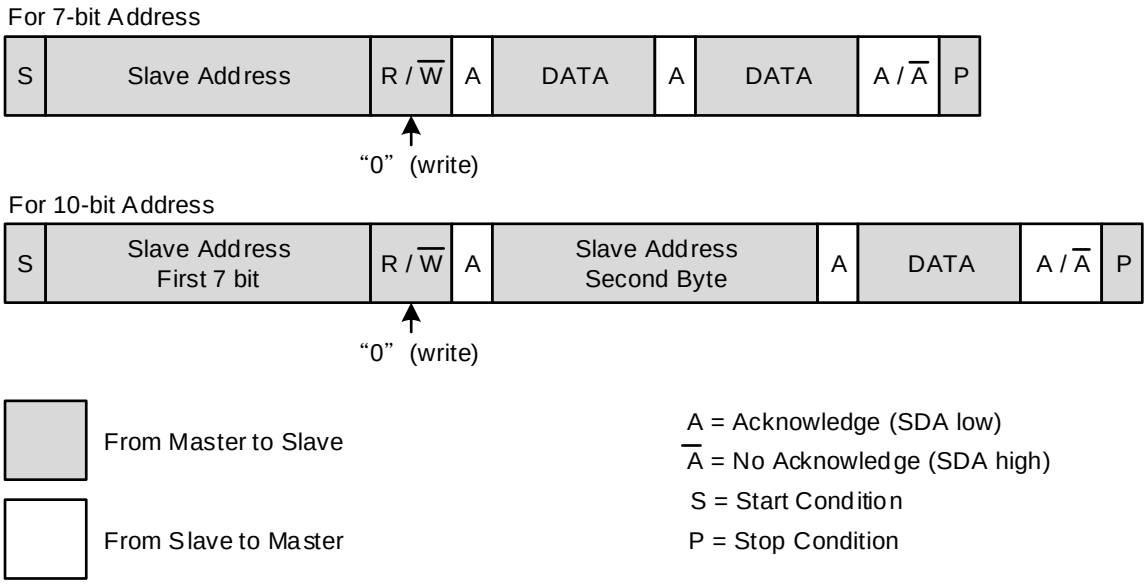
Slave Address	R/W Bit	Description
0000 000	0	General Call Address.
0000 000	1	START byte
0000 001	X	CBUS address
1111 0XX	X	10-bit slave addressing

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.

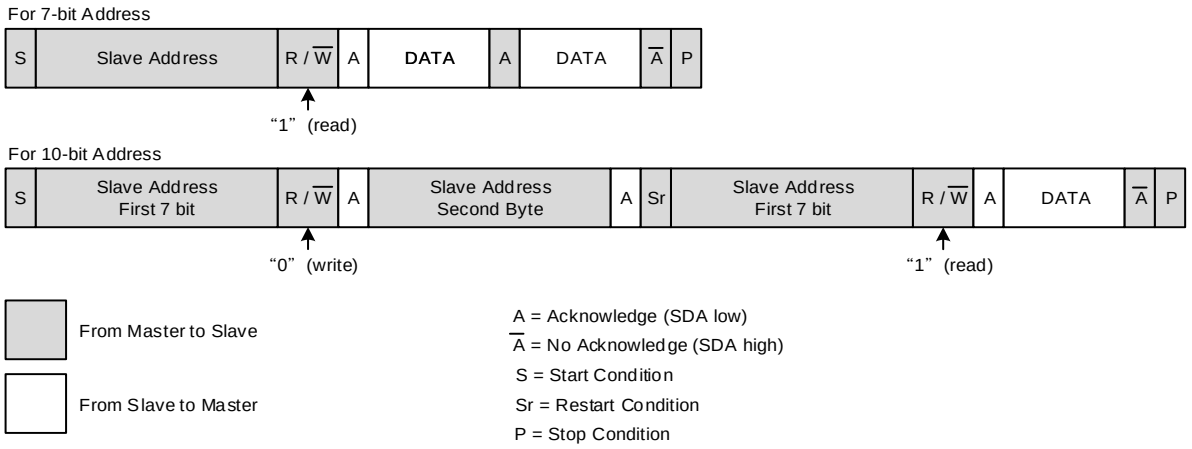
If the master-transmitter is transmitting data, then the slave-receiver responds to the master-transmitter with an acknowledge pulse after every byte of data is received. The transmitting format as follows:

Figure 4.14-5: 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 4-6: Master – Receiver Protocol

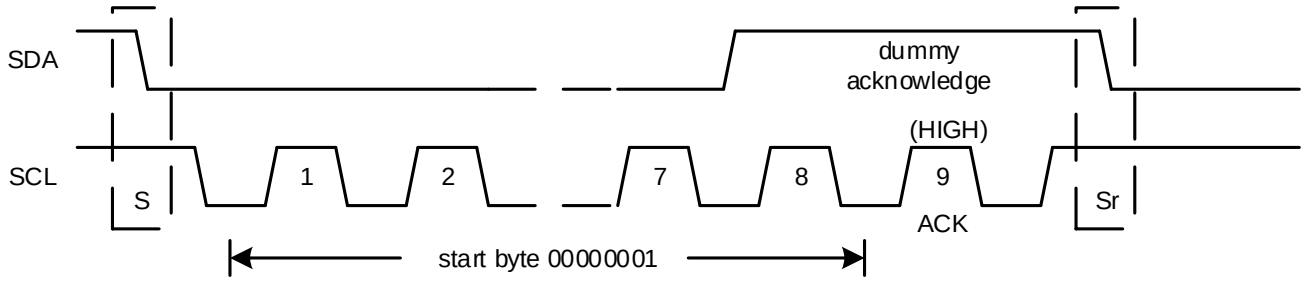


I2C START BYTE Transfer Protocol

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 4.14-7: 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(1 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)

4.14.2.2 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.

4.14.2.3 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.

4.14.3 I2C REGISTER MAP

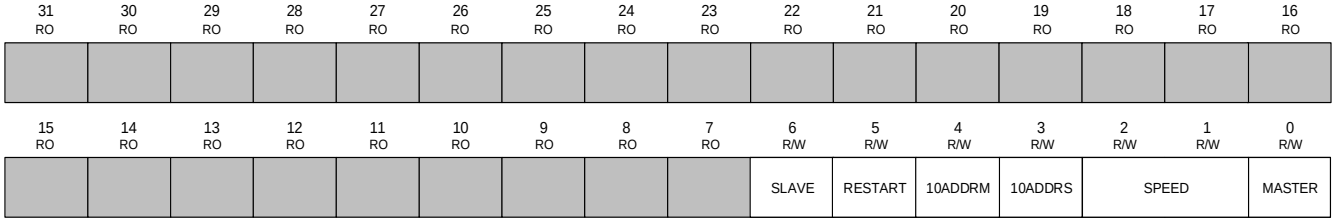
I2C Base Address:

- I2C0: 0x4800_8000
- I2C1: 0x4800_8100

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	I2C_CON	R/W	0x0000_0025	I2C Control Register	266
0x0004	I2C_TAR	R/W	0x0000_0055	I2C Target Address Register	267
0x0008	I2C_SAR	R/W	0x0000_0055	I2C Slave Address Register	267
0x000C	I2C_HS_MADDR	R/W	0x0000_0000	I2C Master Code Address Register	268
0x0010	I2C_DATA_CMD	R/W	0x0000_0000	I2C Tx/Rx Data Buffer and Commend	268
0x0014	I2C_SS_SCL_HCNT	R/W	0x0000_0040	I2C Standard Speed Clock SCL High Count	269
0x0018	I2C_SS_SCL_LCNT	R/W	0x0000_004B	I2C Standard Speed Clock SCL Low Count	269
0x001C	I2C_FS_SCL_HCNT	R/W	0x0000_000A	I2C Fast Speed Clock SCL Hight Count	270
0x0020	I2C_FS_SCL_LCNT	R/W	0x0000_0015	I2C Fast Speed Clock SCL Low Count	270
0x0024	I2C_HS_SCL_HCNT	R/W	0x0000_0001	I2C High Speed Clock Mode SCL High Count	271
0x0028	I2C_HS_SCL_LCNT	R/W	0x0000_0007	I2C High Speed Clock Mode SCL Low Count	271
0x002C	I2C_IER	WO	0x0000_0000	I2C Interrupt Enable Register	272
0x0030	I2C_IDR	WO	0x0000_0000	I2C Interrupt Disable Register	273
0x0034	I2C_IMR	RO	0x0000_0000	I2C Interrupt Mask Status Register	274
0x0038	I2C_RIS	RO	0x0000_0000	I2C Raw Interrupt Status Register	276
0x003C	I2C_ISC	R/W1C	0x0000_0000	I2C Interrupt Status and Clear Register	278
0x0040	I2C_RX_TL	R/W	0x0000_0000	I2C Receive FIFO Threshold Register	279
0x0044	I2C_TX_TL	R/W	0x0000_0000	I2C Transmit FIFO Threshold Register	279
0x006C	I2C_ENABLE	R/W	0x0000_0000	I2C Enable Register	280
0x0070	I2C_STATUS	RO	0x0000_0006	I2C Status Register	280
0x0074	I2C_TXFLR	RO	0x0000_0000	I2C Transmit FIFO Level Register	281
0x0078	I2C_RXFLR	RO	0x0000_0000	I2C Receive FIFO Level Register	281
0x007C	I2C_TIMEOUT	R/W	0x0000_0000	I2C Timeout Enable Register	282
0x0080	I2C_TX_ABRT	R	0x0000_0000	I2C Transmit Abort Status Register	283

4.14.3.1 I2C_CON - I2C Control Register

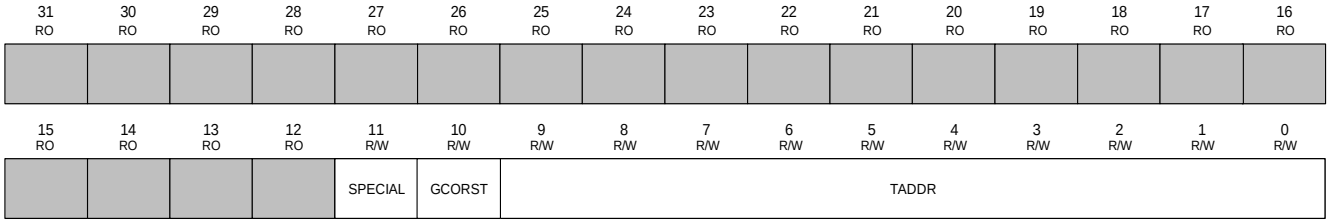
This register can be written only when the I2C is disabled. Writes at other times have no effect.



Offset: 0x0000

Bits	Name	Type	Reset	Description
31: 7	*reserved*	RO	0x0	Software should not 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 (1 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.

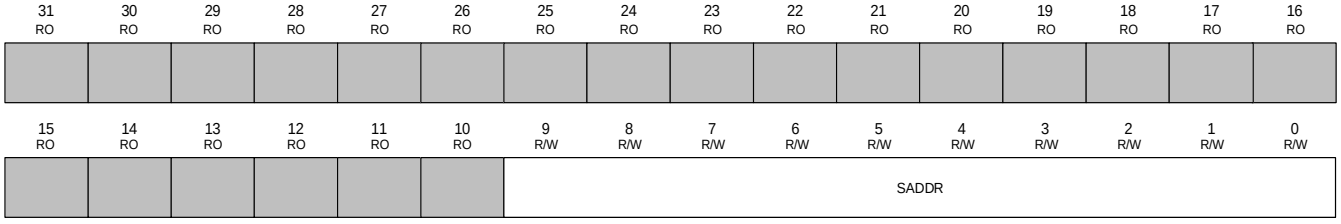
4.14.3.2 I2C_TAR - I2C Target Address Register



Offset: 0x0004

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	*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.

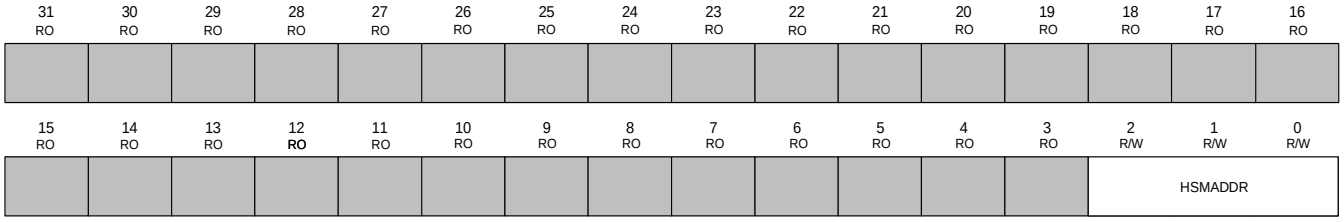
4.14.3.3 I2C_SAR - I2C Slave Address Register



Offset: 0x0008

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.
9:0	*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.

4.14.3.4 I2C_HS_MADDR - I2C High Speed Master Code Address Register

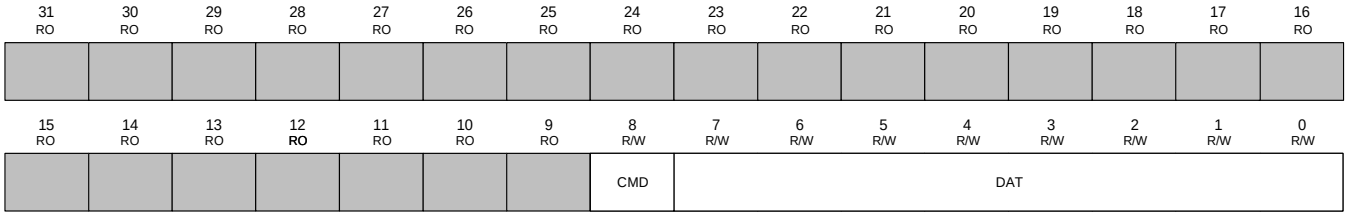


Offset: 0x000C

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	*HSMADDR*	R/W	0x1	I2C High Speed Master Code Address This field holds the value of the I2C HS mode master code.

4.14.3.5 I2C_DATA_CMD - I2C Tx/Rx Data Buffer and Command

This is the register the CPU writes to when filling the TX FIFO. Reading from this register returns bytes from RX FIFO.

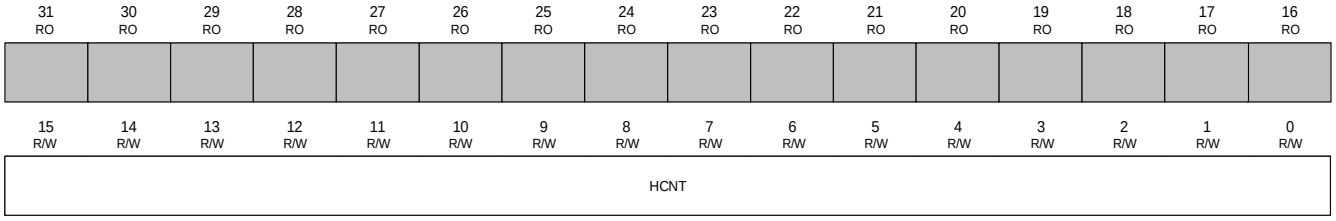


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:9	*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	*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.

4.14.3.6 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.

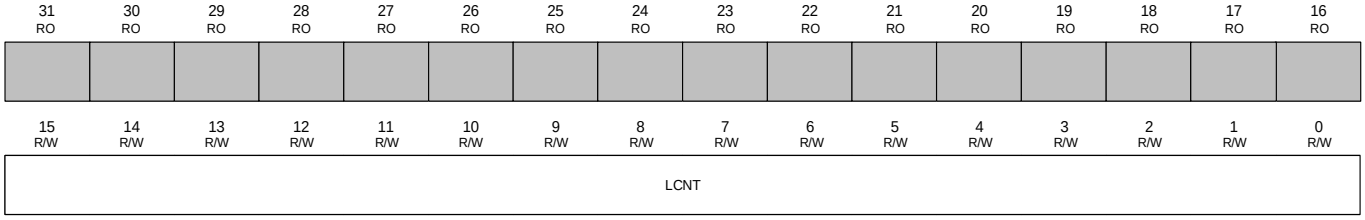


Offset: 0x0014

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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.

4.14.3.7 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.

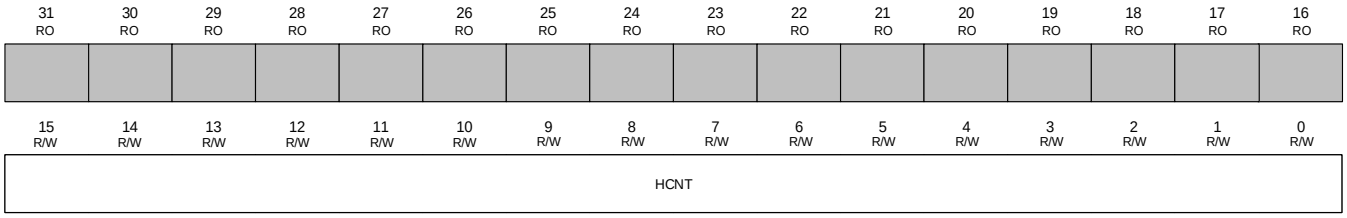


Offset: 0x0018

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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.

4.14.3.8 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.

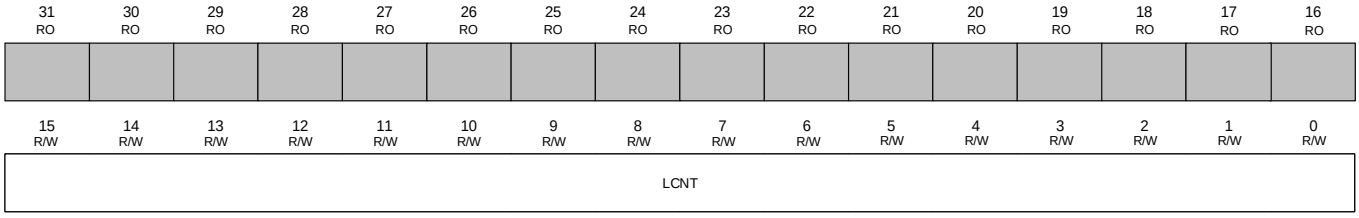


Offset: 0x001C

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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.

4.14.3.9 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.

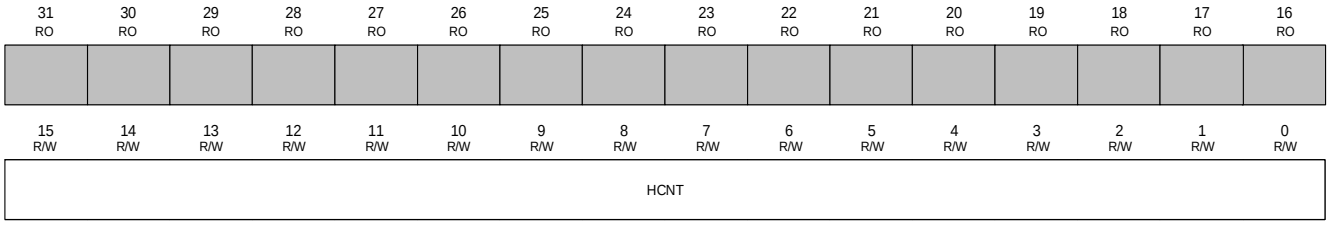


Offset: 0x0020

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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.

4.14.3.10 I2C_HS_SCL_HCNT - I2C High Speed Mode SCL High Count

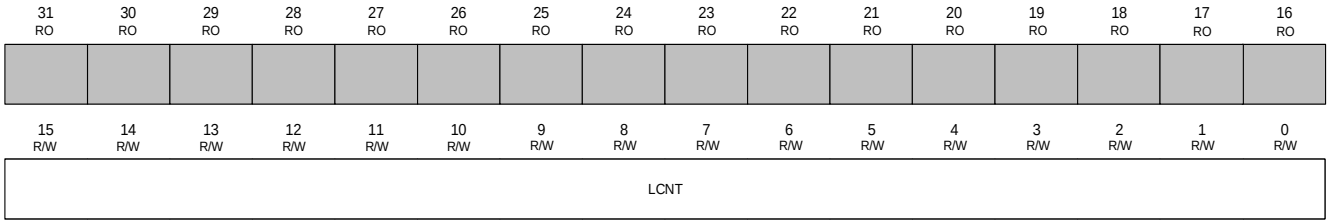
This register sets the SCL clock high-period count for high speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing.



Offset: 0x0024

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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.

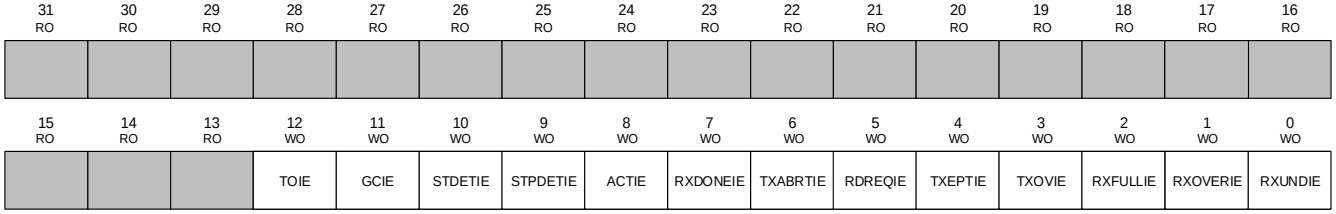
4.14.3.11 I2C_HS_SCL_LCNT - I2C High Speed Mode SCL Low Count



Offset: 0x0028

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	*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.

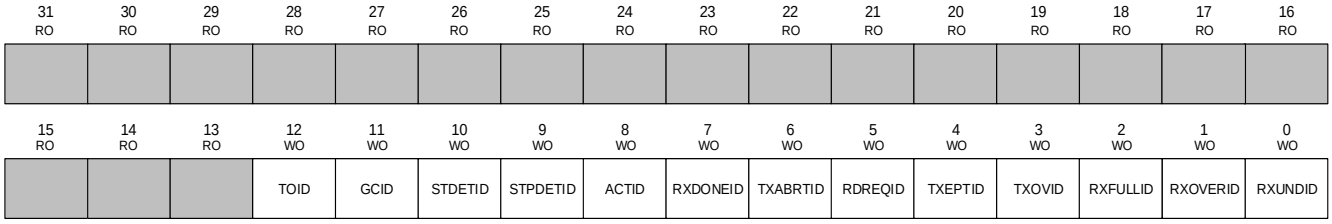
4.14.3.12 I2C_IER - I2C Interrupt Enable Register



Offset: 0x002C

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	*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.

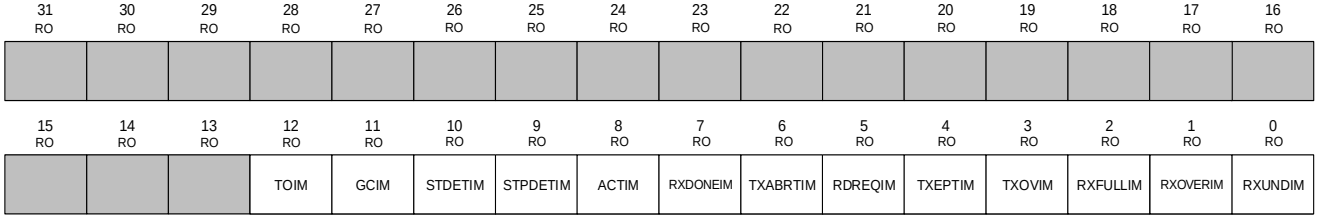
4.14.3.13 I2C_IDR - I2C Interrupt Disable Register



Offset: 0x0030

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	*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.

4.14.3.14 I2C_IMR - I2C Interrupt Mask Status Register

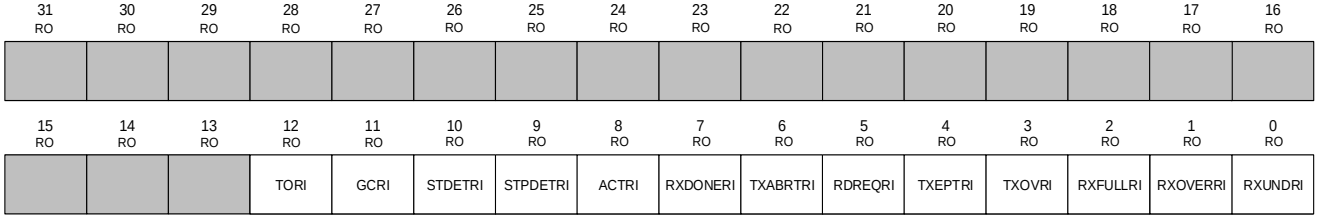


Offset: 0x0034

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	*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

Bits	Name	Type	Reset	Description
				1: Interrupt will not be masked.
0	*RXUNDIM*	RO	0	Rx Underflow Interrupt Mask Status 0: Interrupt will be masked 1: Interrupt will not be masked.

4.14.3.15 I2C_RIS - I2C Raw Interrupt Status Register



Offset: 0x0038

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	*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.
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

Bits	Name	Type	Reset	Description
				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.

4.14.3.16 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

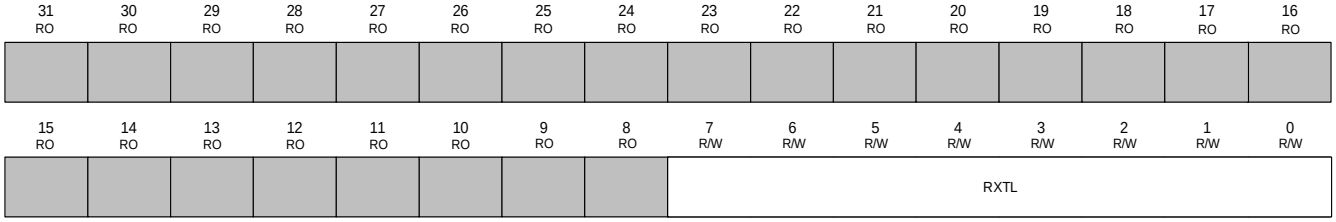
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
			TOIS	GCIS	STDETIS	STPDETIS	ACTIS	RXDONEIS	TXABRTIS	RDREQIS	TXEPTIS	TXOVIS	RXFULLIS	RXOVERIS	RXUNDIS

Offset: 0x003C

Bits	Name	Type	Reset	Description
31:13	*reserved*	RO	0x0	Software should not 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.

Bits	Name	Type	Reset	Description
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.
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.

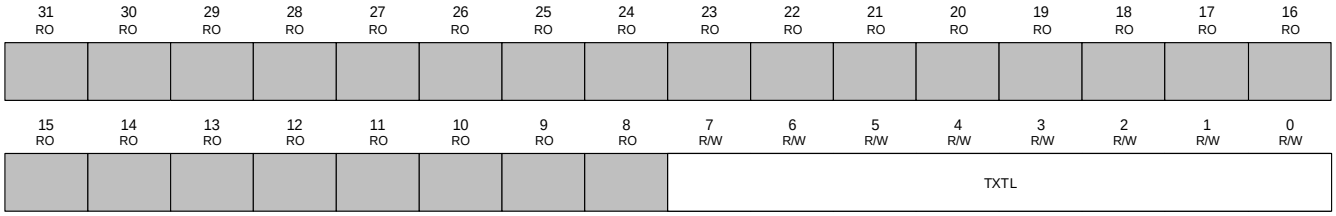
4.14.3.17 I2C_RX_TL - I2C Receive FIFO Threshold Register



Offset: 0x0040

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	*RXTL*	R/W	0x0	Receive FIFO Threshold Level This field controls the level of entries that triggers the RX FULL interrupt.

4.14.3.18 I2C_TX_TL - I2C Transmit FIFO Threshold Register

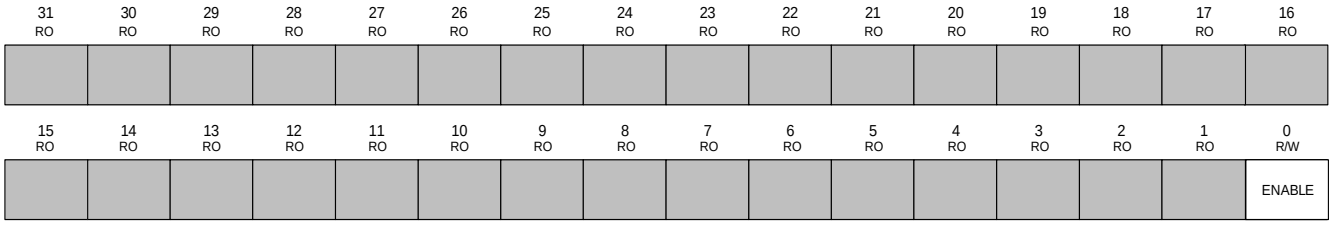


Offset: 0x004C

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	*TXTL*	R/W	0x0	Transmit FIFO Threshold Level This field controls the level of entries that triggers the TX EMPTY interrupt.

4.14.3.19 I2C_ENABLE - I2C Enable Register

Enable and disable i2c operation

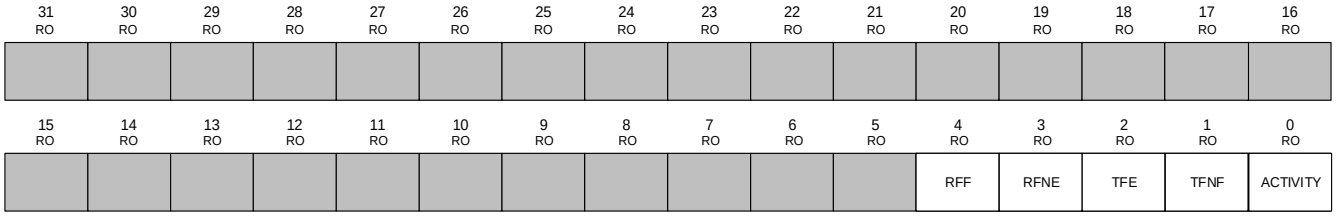


Offset: 0x006C

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	*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.

4.14.3.20 I2C_STATUS - I2C Status Register

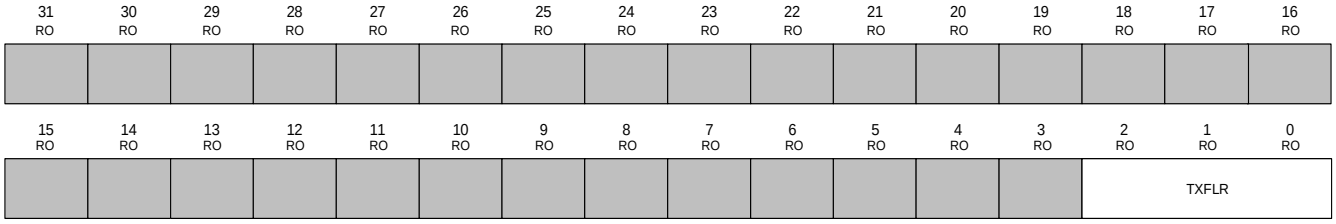
This is a read-only register used to indicate the current transfer status and FIFO status.



Offset: 0x0070

Bits	Name	Type	Reset	Description
31:5	*reserved*	RO	0x0	Software should not 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.

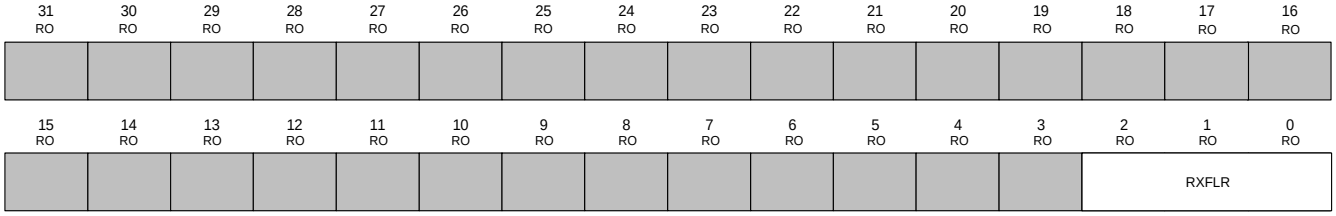
4.14.3.21 I2C_TXFLR - I2C Transmit FIFO Level Register



Offset: 0x0074

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	*TXFLR*	RO	0x0	Transmit FIFO Level This field contains the number of valid data entries in the transmit FIFO.

4.14.3.22 I2C_RXFLR - I2C Receive FIFO Level Register

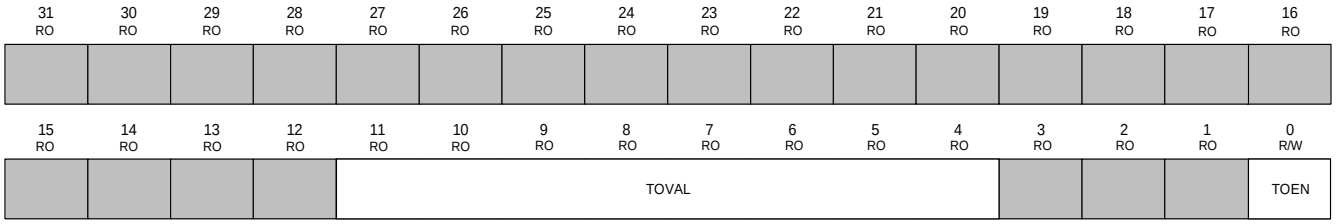


Offset: 0x0078

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	*RXFLR*	RO	0x0	Receive FIFO Level This field contains the number of valid data entries in the receive FIFO.

4.14.3.23 I2C_TIMEOUT - I2C Timeout Enable Register

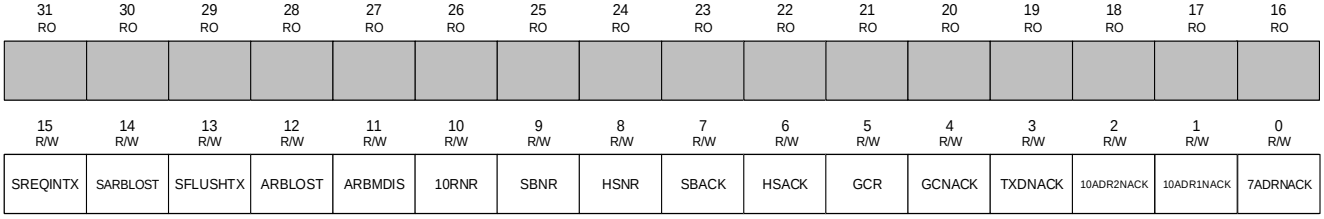
This register contains the timeout function enable and the timeout counter value.



Offset: 0x007C

Bits	Name	Type	Reset	Description
31:12	*reserved*	RO	0x0	Software should not 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.

4.14.3.24 I2C_TX_ABRT - I2C Transmit Abort Status Register



Offset: 0x0080

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	*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

Bits	Name	Type	Reset	Description
				acknowledgement from the remote slave.
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 SERIAL PERIPHERAL INTERFACE (SPI)

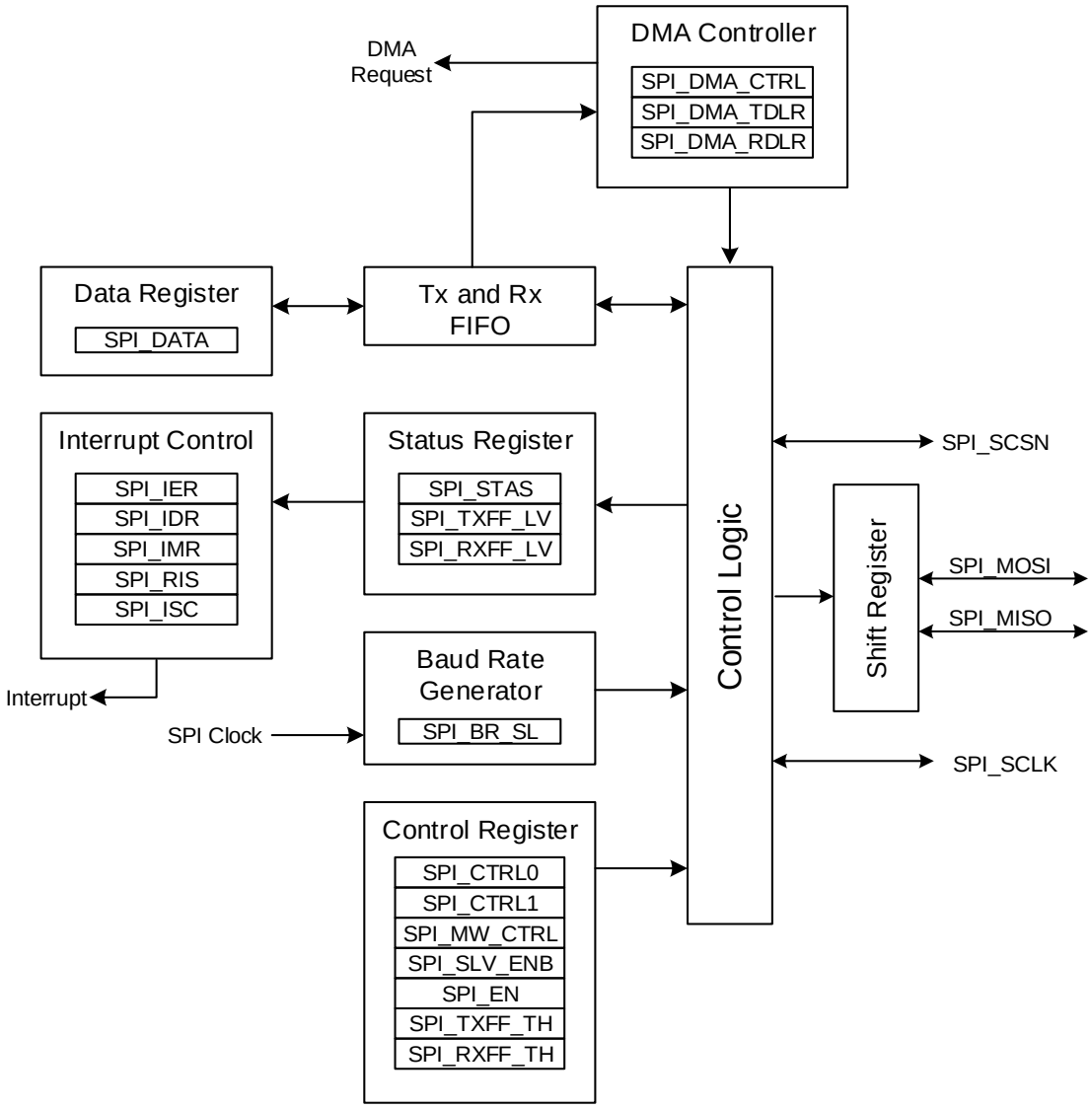
The Serial Peripheral Interface (SPI) provides an SPI protocol data transmit and receive functions in both master or slave mode. SPI supports full-duplex, half-duplex and simple synchronous, serial communication with external devices. On the principle of full duplex data transmission, 4bit to 16bit data sent by the master to the slave or the slave sent to the master. In fact, usually only the information flow in one direction contain meaningful information.

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
- Synchronous serial communication.
- Single data direction operation allows alternate function on MISO or MOSI pin
- 4 - 16 bits per frame.

4.15.1 BLOCK DIAGRAM

Figure 4.15-1: 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 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

4.15.2 SPI FUNCTION DESCRIPTION

4.15.2.1 Serial Bit-Rate Clocks

The maximum frequency of the SPI master bit-rate clock (SPI_SCLK) is one-half the frequency of the SPI master clock (SPI clock).

The frequency of the SPI_SCLK is defined by:

$F_{SPI_SCLK} = F_{SPI\ Clock} / SCKDV$

SCKDV is a bit field in the register **I2C_BR_SL**, holding any even value in the range 2 to 65534. If *SCKDV* is equal or smaller than 1, then SPI_SCLK is disabled.

4.15.2.2 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.

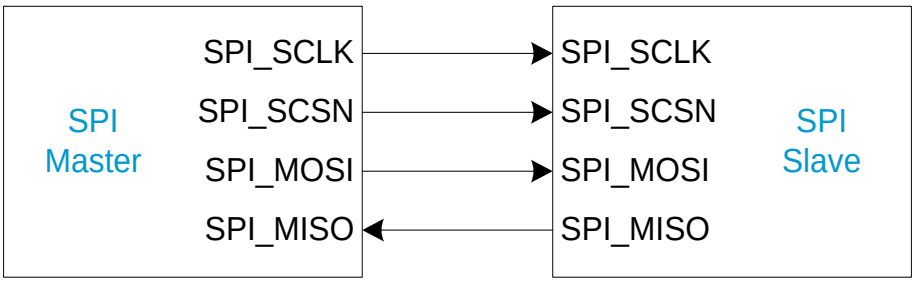
When **SPI_CTRL0.**TMOD = 3, the transmit data is used to transmit an opcode and/or an address to the EEPROM device. Typically this takes three data frames (8-bit opcode followed by 8-bit upper address and 8-bit lower address). During the transmission of the opcode and address, no data is captured by 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.

4.15.2.3 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 4.15-2: 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 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.

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.

4.15.2.4 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 4.15-1: SPI 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 4.15-3: SPI Serial Format in Case 1

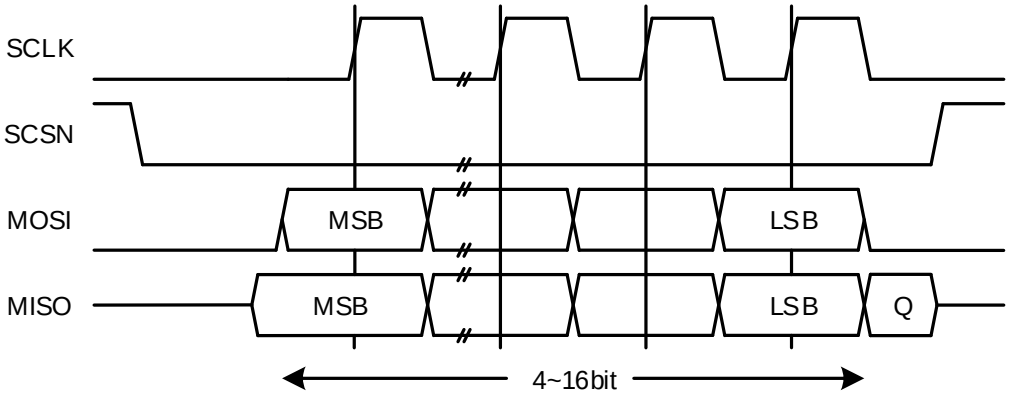
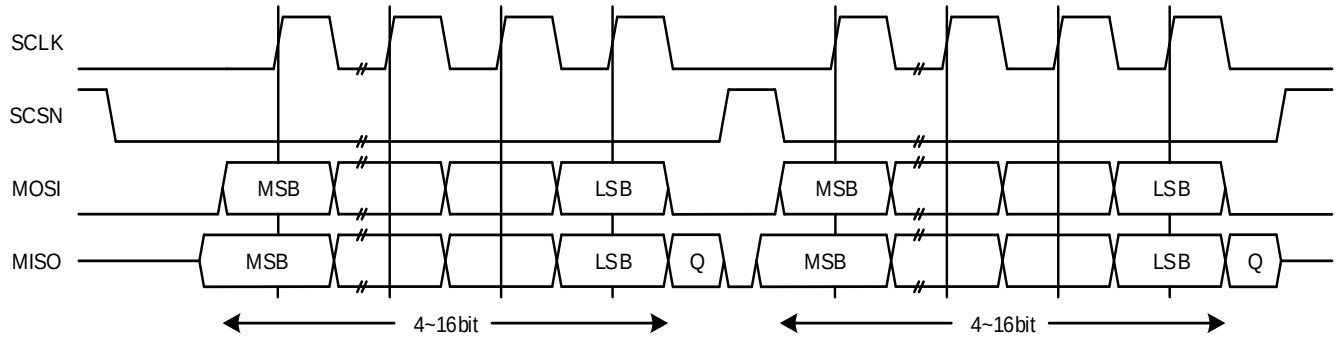


Figure 4.15-4: SPI Serial Format Continuous Transfer in Case 1



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 4.15-5: SPI Serial Format in Case 2

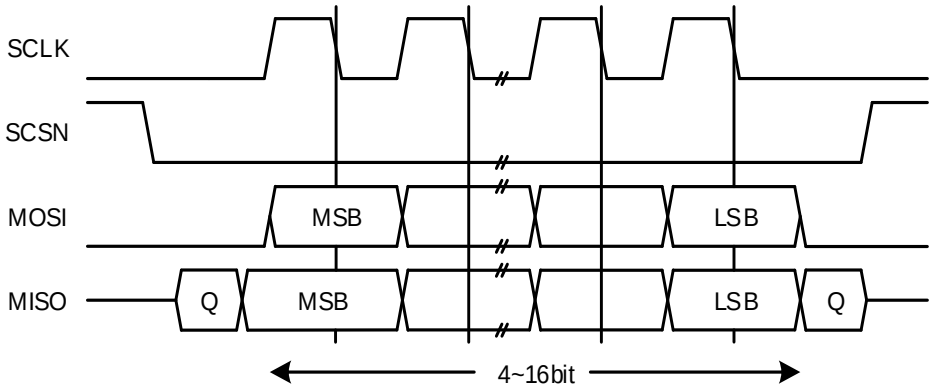
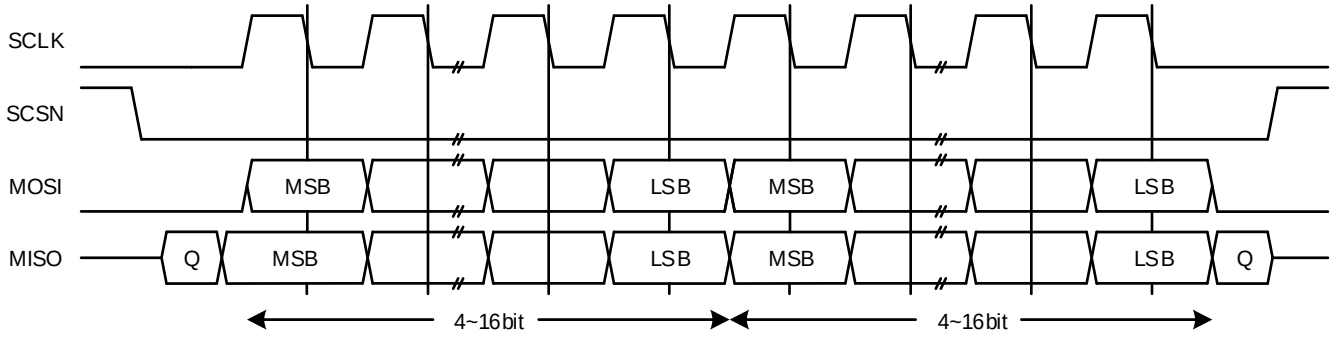


Figure 4.15-6: SPI Serial Format Continuous Transfer in Case 2



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 4.15-7: SPI Serial Format in Case 3

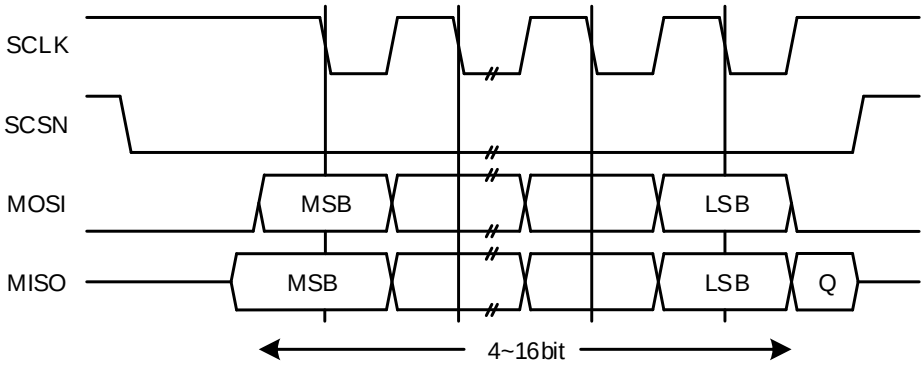
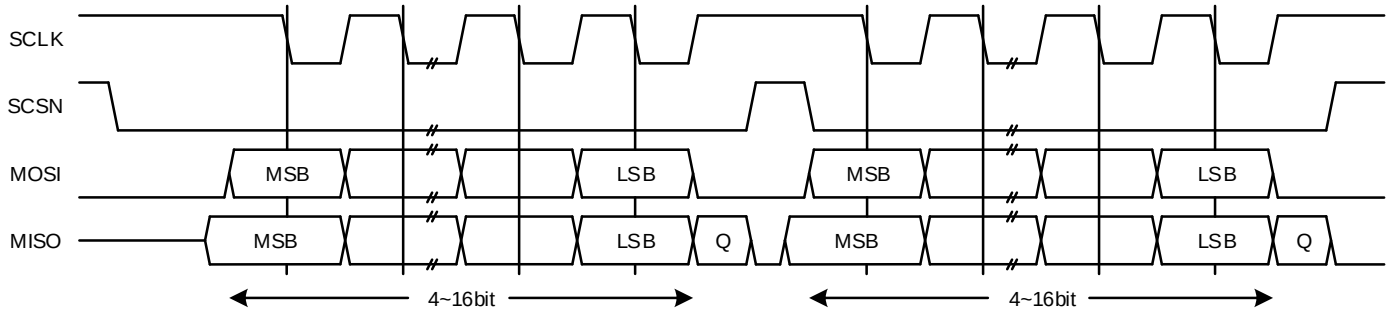


Figure 4.15-8: SPI Serial Format Continuous in Case 3



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 4.15-9: SPI Serial Format in Case 4

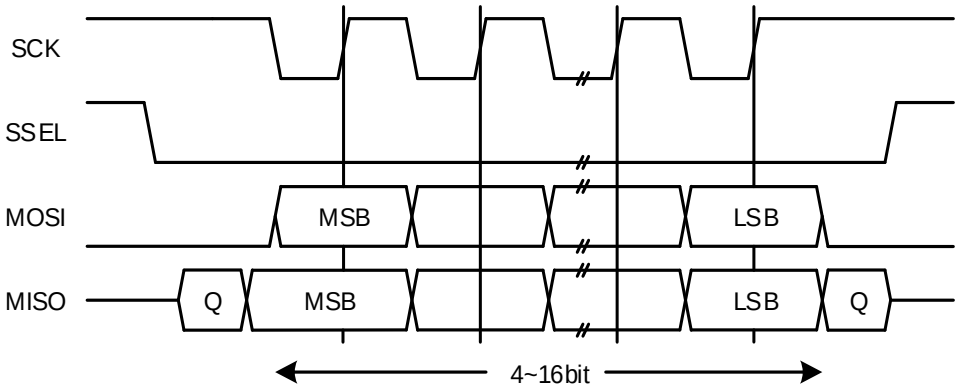
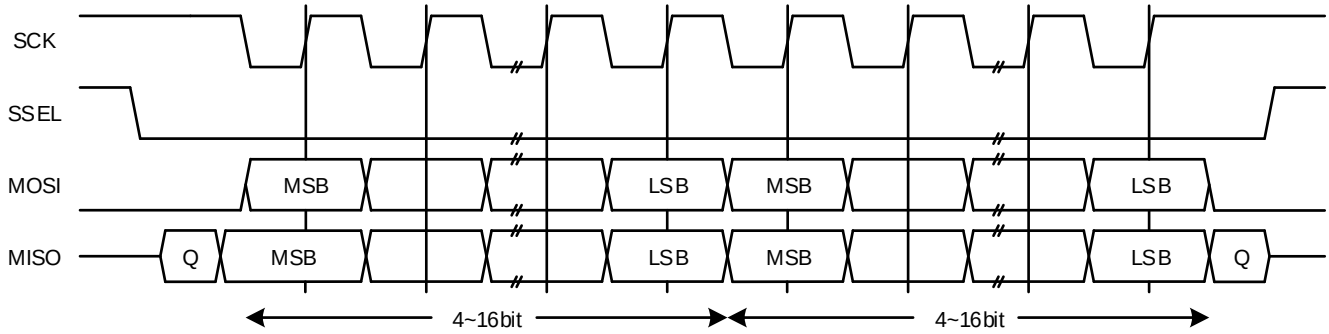


Figure 4.15-10: SPI Serial Format Continuous Transfer in Case 4



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 4.15-11: SSP Serial Format

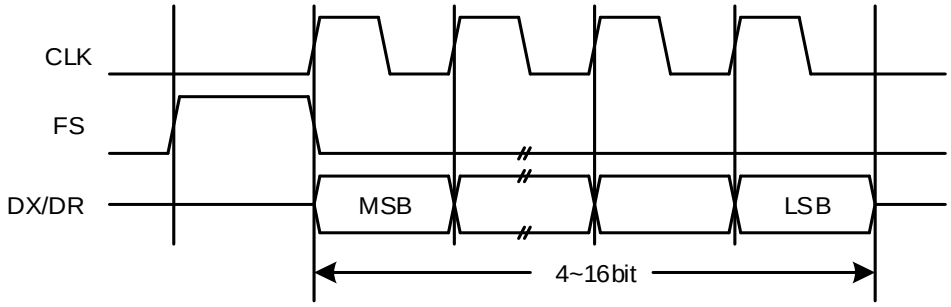
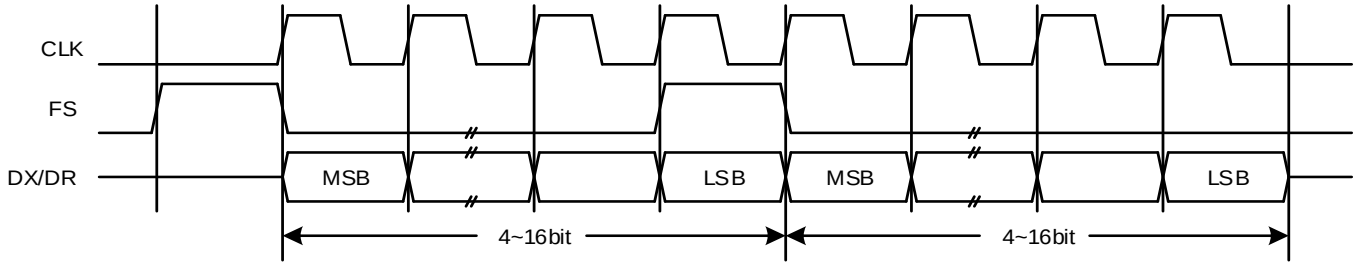


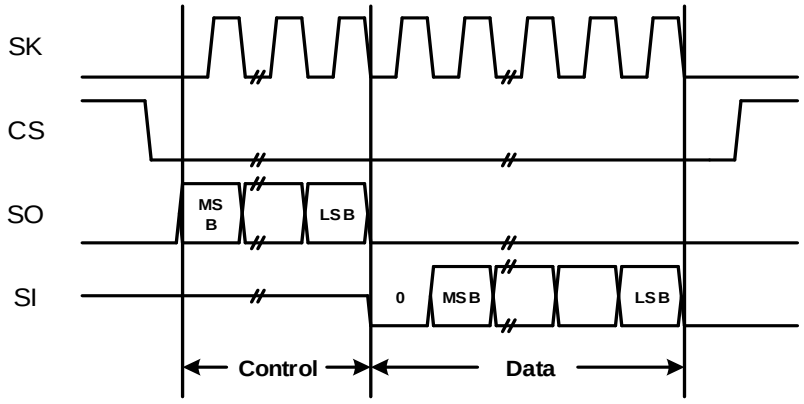
Figure 4.15-12: SSP Serial Format Continuous Transfer



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 4.15-13: 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.

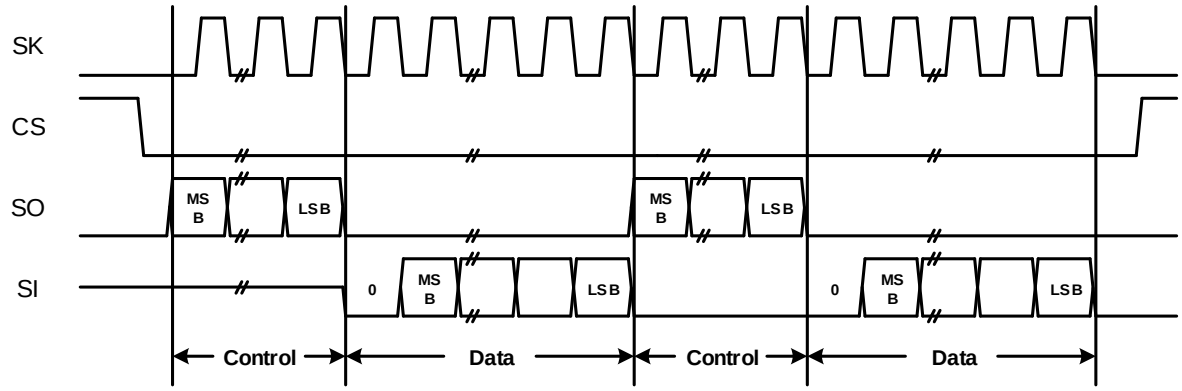
The slave-select signal is held active-low during the transfer and is de-asserted one-half clock cycle later, after the data are transferred.

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 4.15-14: Microwire Serial Format in Case 1

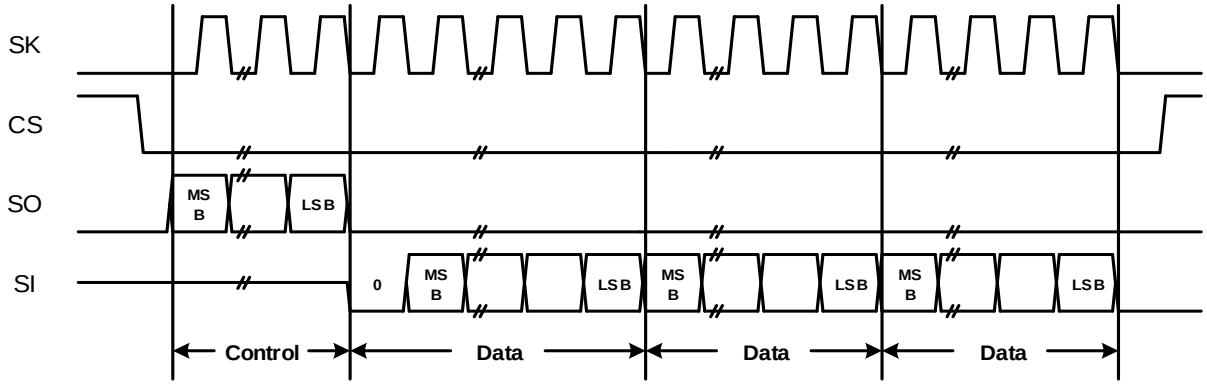


In this case of continuous transmission of this configuration, the transfer is the same with the single format. After the current data LSB byte is received, the control byte of the next frame will be sent immediately. After a frame LSB byte is latched into the SPI shift register, each received data is 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 4.15-15: Microwire Serial Format in Case 2

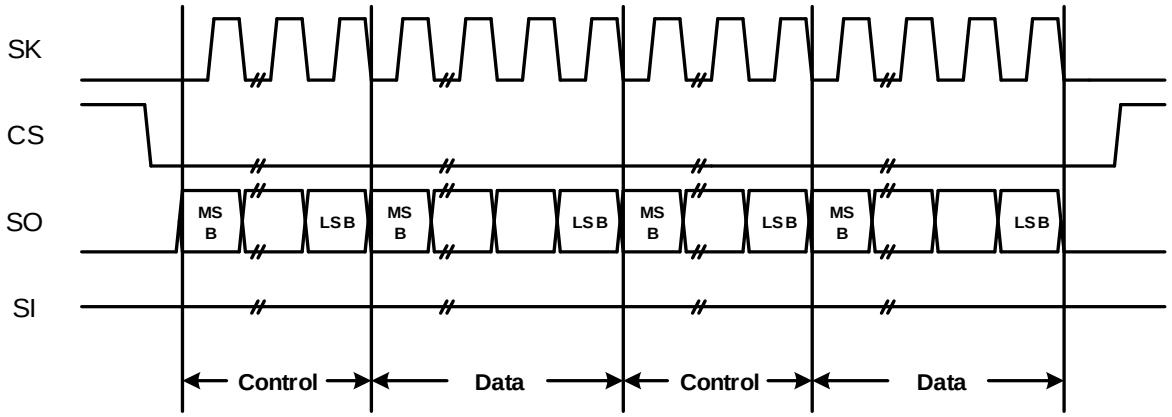


In this case of continuous transmission of this configuration, the transfer is the same with the single format. After the current data LSB byte is received, the control byte of the next frame will be sent immediately. The number of receive data bytes can setting by **SPI_CTRL1** register. After a frame 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 4.15-16: Microwire Serial Format in Case 3

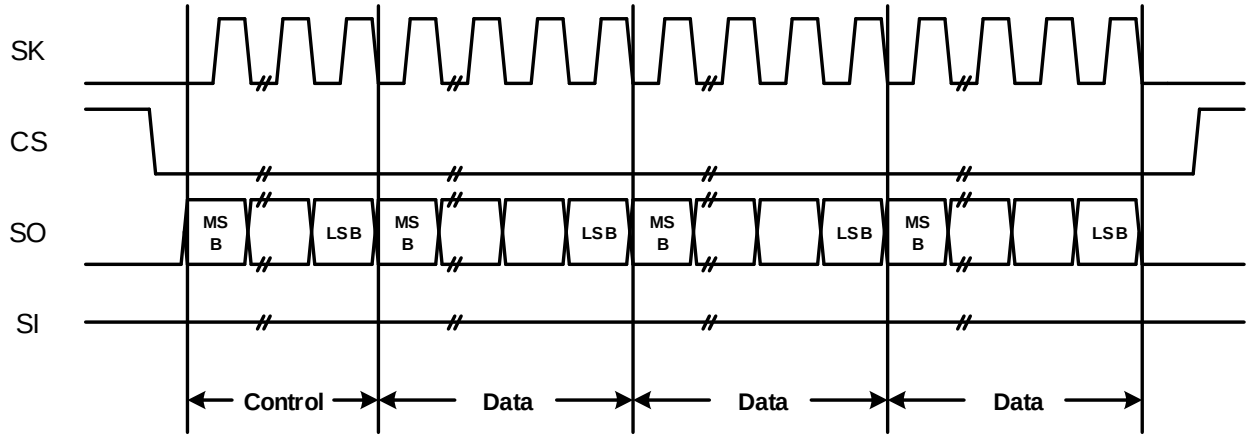


In this case of continuous transmission of this configuration, after the LSB current source frame has been transmitted, the next control byte MSB would then transmitted. At this condition is not open receive.

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 4.15-17: Microwire Serial Format in Case 4



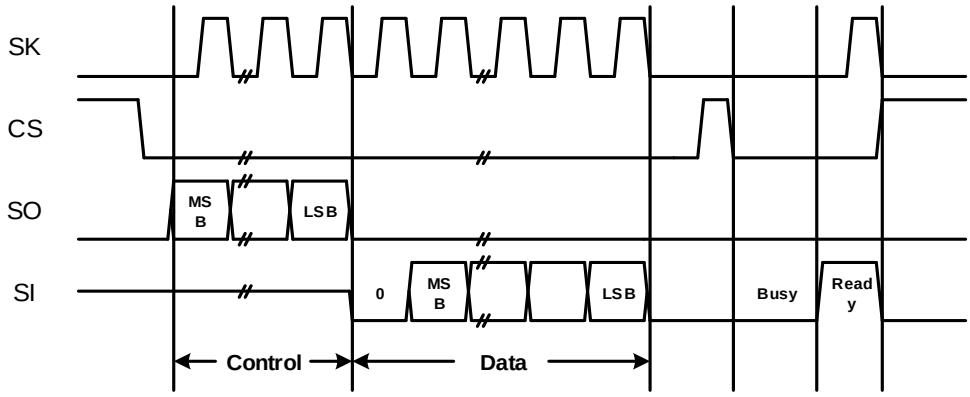
In this case of continuous transmission of this configuration, after the LSB current source frame has been transmitted, the next control byte MSB would then transmitted. At this condition is not open receive.

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 4.15-18: 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.

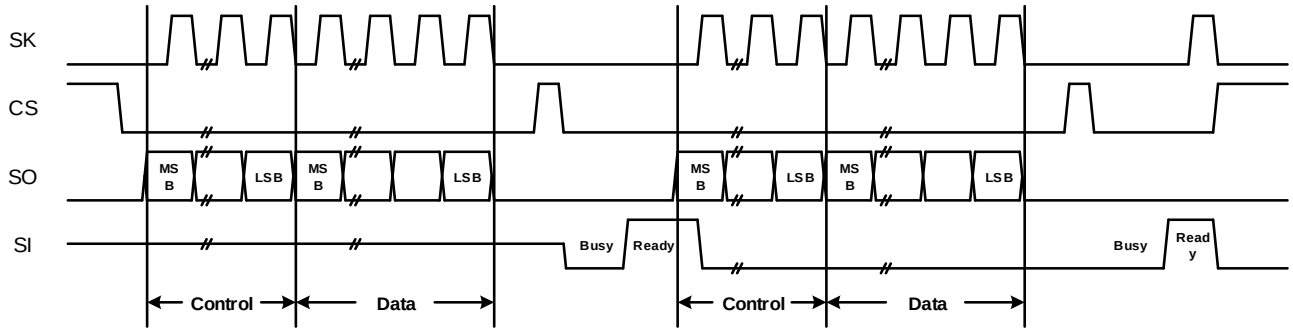
Note: If set *MWMOD* = 1, the handshake signal issuing after receive data transmission completely (in the **SPI_CTRL1** set the number of data).

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 4.15-19: Microwire Serial Format in Case 7



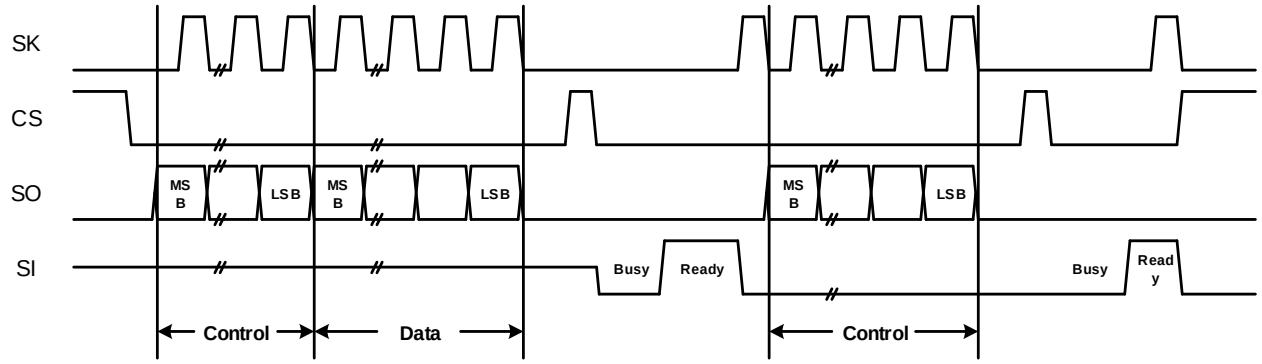
In this case of continuous transmission of this configuration, after the current data byte LSB is received, it toggle the CS, and began waiting outside the machine to respond to Ready. After receiving the response, will continue to send the control and data byte until all the data are be send. At this 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 4.15-20: 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.

4.15.3 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

4.15.3.1 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 – **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.

4.15.3.2 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 – **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.

4.15.3.3 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.

4.15.3.4 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.

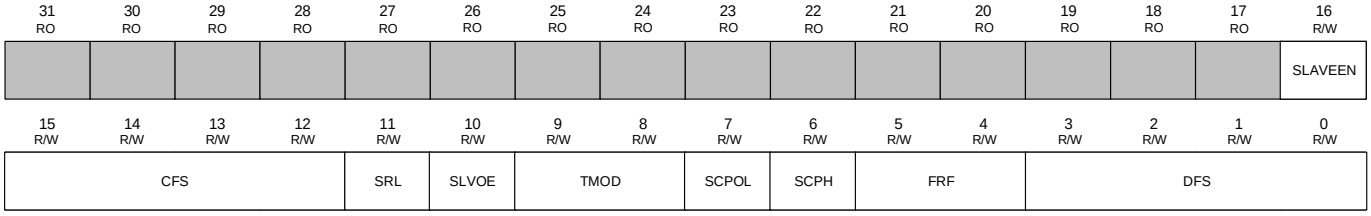
4.15.4 SPI REGISTER MAP

Base address: 0x4800_4000

Offset	Symbol	Type	Reset Value	Description	See page
0x0000	SPI_CTRL0	R/W	0x0000_0007	SPI Control Register 0	304
0x0004	SPI _CTRL1	R/W	0x0000_0000	SPI Control Register1	306
0x0008	SPI _ENABLE	R/W	0x0000_0000	SPI Enable Register	307
0x000C	SPI _MW_CTRL	R/W	0x0000_0000	Microwire Control Register	308
0x0010	SPI_SLV_ENB	R/W	0x0000_0000	SPI Slave Enable Register	309
0x0014	SPI_BR_SL	R/W	0x0000_0000	SPI Baud Rate Select	310
0x0018	SPI_TXFF_TH	R/W	0x0000_0000	SPI Transmit FIFO Threshold Level	311
0x001C	SPI_RXFF_TH	R/W	0x0000_0000	SPI Receive FIFO Threshold Level	312
0x0020	SPI_TXFF_LV	RO	0x0000_0000	SPI Transmit FIFO Level Register	313
0x0024	SPI_RXFF_LV	RO	0x0000_0000	SPI Receive FIFO Level Register	313
0x0028	SPI_STAS	RO	0x0000_0006	SPI Status Register	314
0x002C	SPI_IER	WO	0x0000_0000	SPI Interrupt Enable Register	315
0x0030	SPI_IDR	WO	0x0000_0000	SPI Interrupt Disable Register	316
0x0034	SPI_IMR	RO	0x0000_0000	SPI Interrupt Mask Status Register	317
0x0038	SPI_RIS	RO	0x0000_0000	SPI Raw Interrupt Status Register	318
0x003C	SPI_ISC	R/W1C	0x0000_0000	SPI Interrupt Status and Clear Register	319
0x0060	SPI_DATA	R/W	0x0000_0000	SPI Data Register	319

4.15.4.1 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.



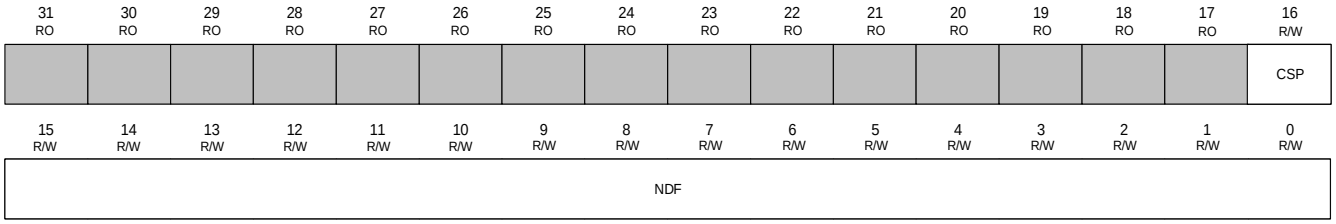
Offset: 0x0000

Bits	Name	Type	Reset	Description
31:17	*reserved*	RO	0x0	Software should not 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.

Bits	Name	Type	Reset	Description
				0x2: Receive Only. 0x3: EEPROM Read.
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

4.15.4.2 SPI _CTRL1 - SPI Control Register 1

This register controls the end of serial transfers when in receive-only mode.

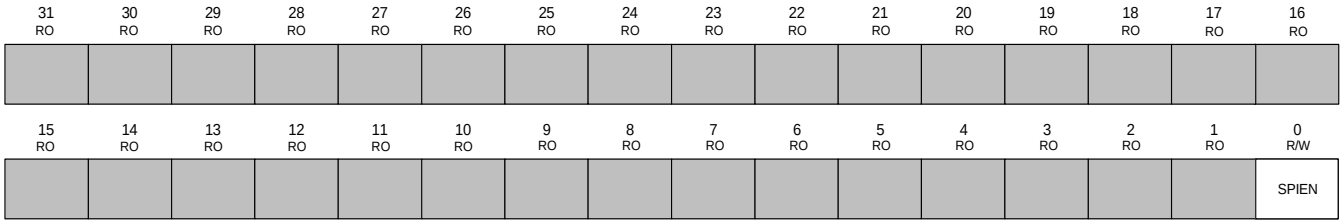


Offset: 0x0004

Bits	Name	Type	Reset	Description
31:17	*reserved*	RO	0x0	Software should not 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.

4.15.4.3 SPI_ENABLE - SPI Enable Register

This register is used to enable SPI.

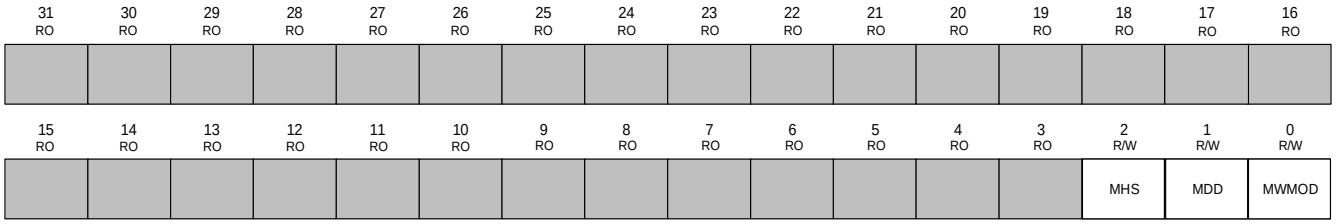


Offset: 0x0008

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	*SPIEN*	R/W	0	SPI 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

4.15.4.4 SPI_MW_CTRL - Microwire Control Register

This register controls the direction of the data word for the half-duplex Microwire serial protocol.

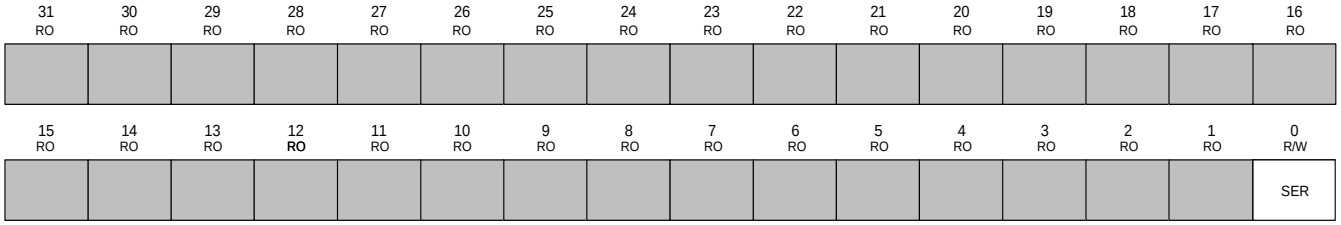


Offset: 0x000C

Bits	Name	Type	Reset	Description
31:3	*reserved*	RO	0x0	Software should not 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.

4.15.4.5 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.

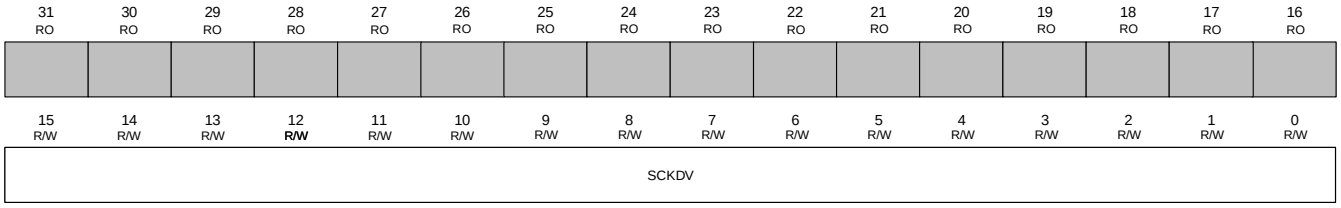


Offset: 0x0010

Bits	Name	Type	Reset	Description
31:1	*reserved*	RO	0x0	Software should not 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.

4.15.4.6 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.

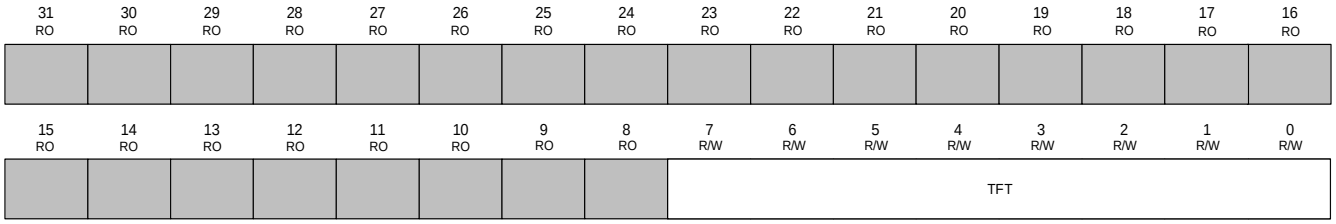


Offset: 0x0014

Bits	Name	Type	Reset	Description
31:16	*reserved*	RO	0x0	Software should not 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 2, 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 2 and 65534. For example: $F_{SPI\ Clock}$ = 16 MHz, then SCKDV = 4, F_{SPI_SCLK} = 16 / 4 = 4 MHz

4.15.4.7 SPI_TXFF_TH - SPI Transmit FIFO Threshold Level

This register controls the threshold value for the transmit FIFO memory.

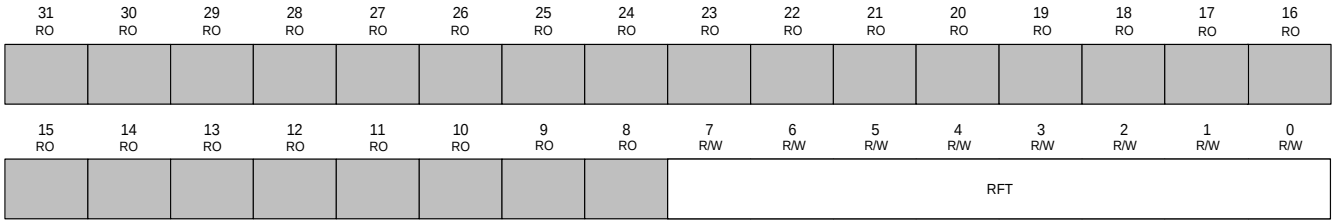


Offset: 0x0018

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not 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.

4.15.4.8 SPI _RXFF_TH - SPI Receive FIFO Threshold Level

The register controls the threshold value for the receive FIFO memory

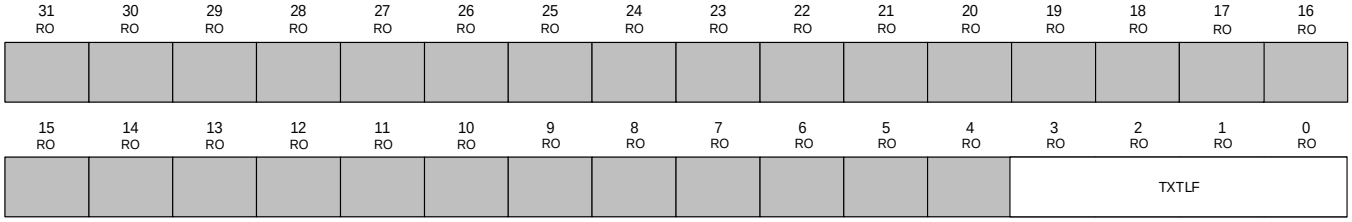


Offset: 0x001C

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not 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.

4.15.4.9 SPI_TXFF_LV - SPI Transmit FIFO Level Register

This register contains the number of valid data entries in the transmit FIFO memory.

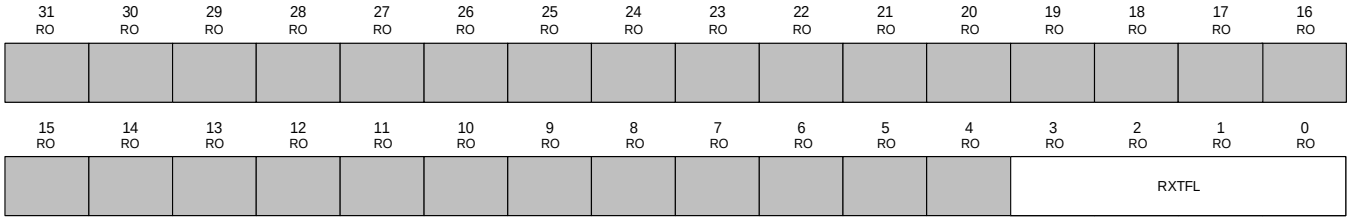


Offset: 0x0020

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not 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.

4.15.4.10 SPI _RXFF_LV - SPI Receive FIFO Level Register

This register contains the number of valid data entries in the receive FIFO memory.

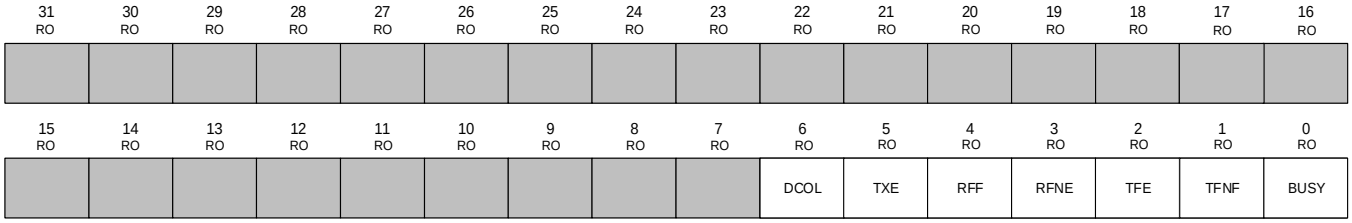


Offset: 0x0024

Bits	Name	Type	Reset	Description
31:8	*reserved*	RO	0x0	Software should not 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.

4.15.4.11 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

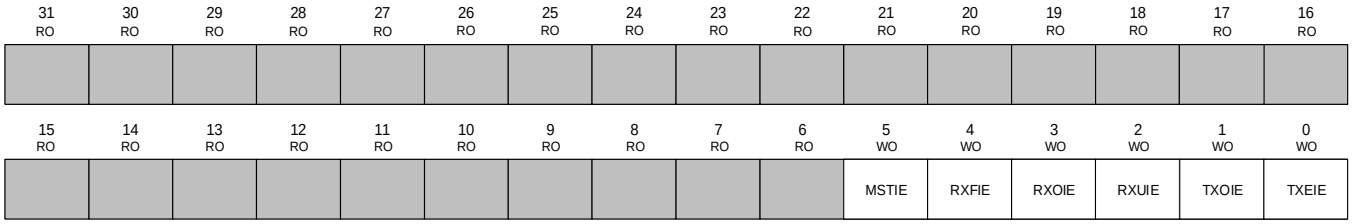


Offset: 0x0028

Bits	Name	Type	Reset	Description
31:7	*reserved*	RO	0x0	Software should not 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.

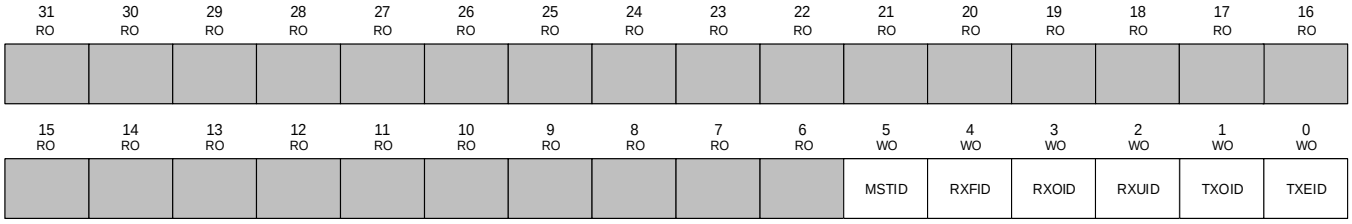
4.15.4.12 SPI _IER - SPI Interrupt Enable Register



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.

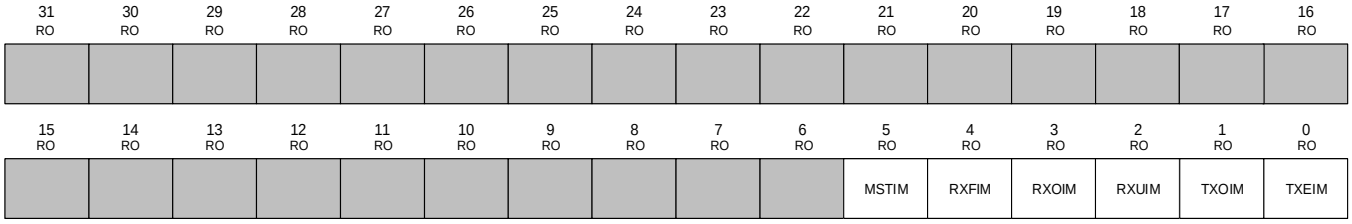
4.15.4.13 SPI_IDR - SPI Interrupt Disable Register



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.

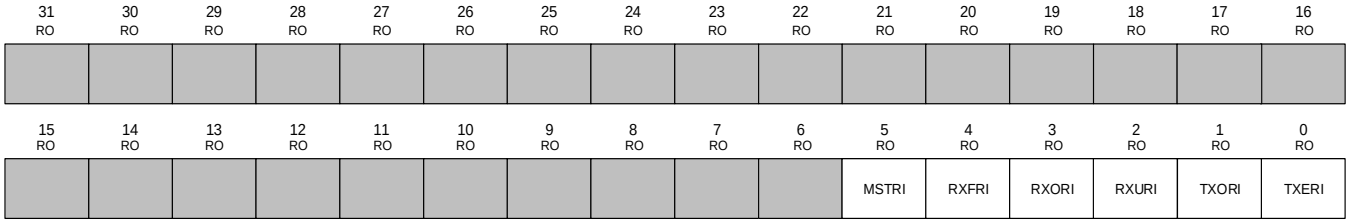
4.15.4.14 SPI _IMR - SPI Interrupt Mask Status Register



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

4.15.4.15 SPI_RIS - SPI Raw Interrupt Status Register

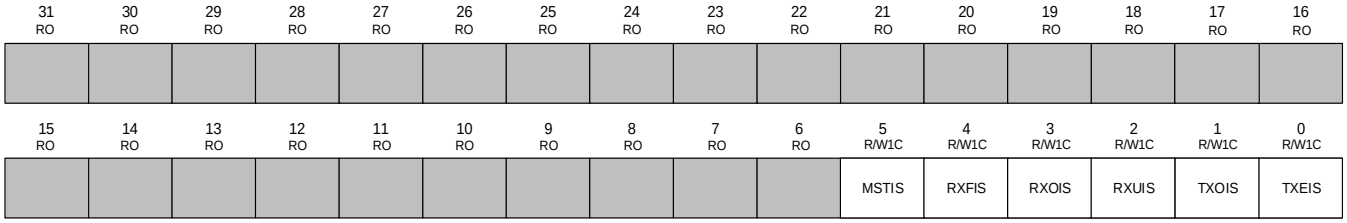


Offset: 0x0038

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.

4.15.4.16 SPI _ISC - SPI Interrupt Status and Clear Register

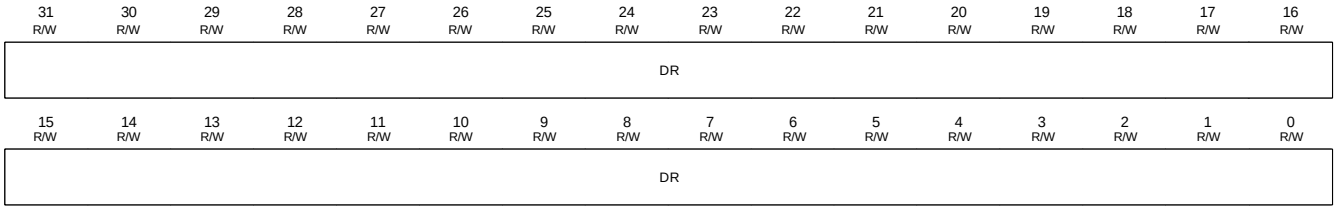
Note: This register is the read and write to clear register. A write of '1' to individual bit clears the respective interrupt.



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.

4.15.4.17 SPI _DATA - SPI 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.

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.

Parameter	Symbol	Min.	Max.	Unit
V_{DD} Supply Voltage	V_{DD} -V_{SS}	-0.3	3.6	V
Input Voltage	V_I/V_O		3.6	V
Operating Ambient Temperate	T_A	-40	+85	℃
Storage Temperate	T_{STG}	-40	150	℃

5.2 OPERATING CONDITIONS

All the electrical characteristics are applicable to the following conditions unless otherwise specified:

- Operating temperature range: TA = -40°C to 85°C and for a junction temperature up to T_J = 125°C.

Typical values are based on TA = 25°c and VI= 3,3V unless otherwise specified

5.3 I/O PIN CHARACTERISTICS

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Pull-up resistor	R_{PU}	Weak pull-up equivalent resistor		34K	49K	74K	Ω
Pull-down resistor	R_{PD}	Weak pull-down equivalent resistor		30K	47K	86K	Ω
Input leakage current	I_{LKG}					±1μ	A
Output low-level voltage	V_{OL}					0.4	
Output high-level voltage	V_{OH}			2.4			
Input low-level voltage	V_{IL}			-0.3		0.8	V
Input high-level voltage	V_{IH}			2		3.6	V
Schmitt Trigger Low to High Threshold Point	V_{T+}			1.53		1.66	V
Schmitt Trigger High to Low Threshold Point	V_{T-}			1.13		1.27	V
Low Level Output Current	I_{OL}	V_{OL}(Max)	Output Drive: 2mA	2.4	3.8	5.3	mA
			Output Drive: 4mA	4.7	7.6	10.6	
High Level Output Current	I_{OH}	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
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
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

5.6 POWER-ON RESET CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
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
NRST Input low level voltage	$V_{IL(NRST)}$ (1)		-0.5		0.8	V
NRST Input high level voltage	$V_{IH(NRST)}$		2		V_{DD} + 0.5	
NRST Schmitt trigger voltage hysteresis	$V_{hys(NRST)}$ (1)			200		mV
Weak pull-up equivalent resistor(1)	R_{PU}	$V_{IN} = V_{SS}$	30	40	50	kΩ
NRST Input filtered pulse	$V_{F(NRST)}$ (1)				100	ns
NRST Input not filtered pulse	$V_{NF(NRST)}$ (1)		300			ns
External load capacitance	C_L			0.1		μF
Native reset time	T_R	C_L =0.1u		2.7		ms

(1). The pull-up is designed with a true resistance in series with a switchable PMOS. This PMOS contribution to the series resistance must be minimum (~10% order)

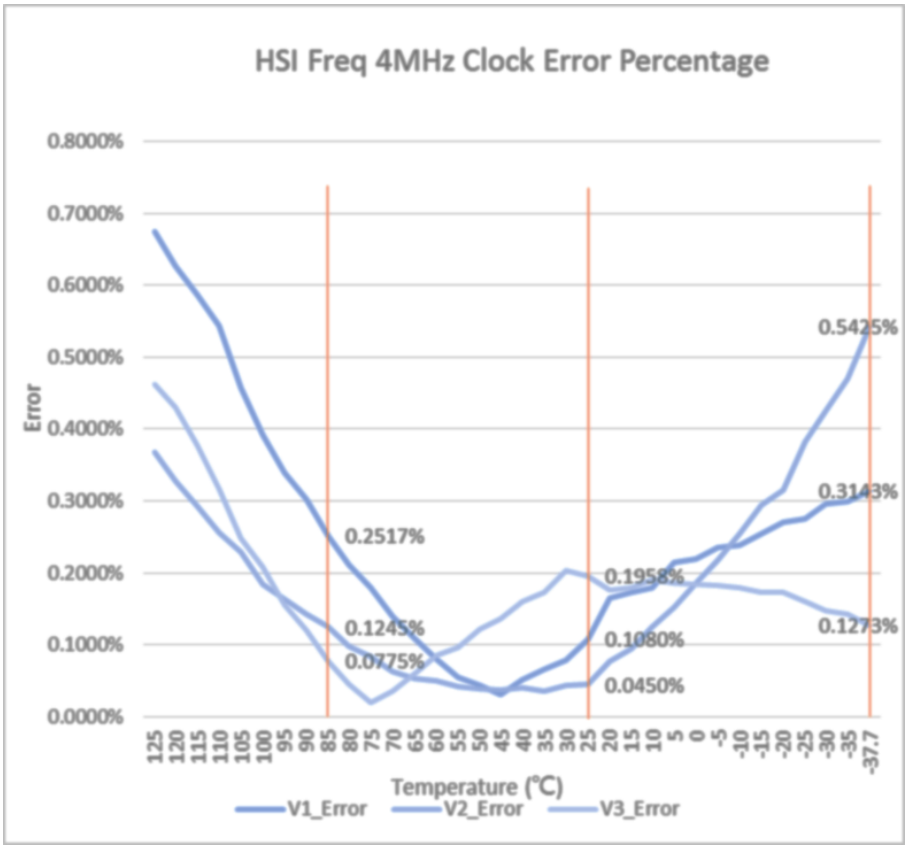
5.8 8 MHZ XTAL CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
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 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/℃
Accuracy of the RCoscillator (factory calibrated)	ACCrcosc	T_A = 25℃		±2.5		%
Output startup time	Tsu				10	CLK
Trim settling time	Tst				5	CLK

Figure 5.9-1: HSI Frequency 4MHz Error Percentage



The operating temperature has an impact on the accuracy of the 4MHz RC oscillators. At 25℃, the HIS oscillators have an accuracy of ±0.2% typically, but in the temperature range of -40 to 125℃, the accuracy decreas.

5.10 32 KHZ XTAL

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
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 TEMPERATURE SENSOR CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Current Consumption	I_{DD}		70	100	160	μA
Standby Current	I_{STBY}	EN = LOW			1	μA
Temperature Range	T_J		-40		125	℃
Output Voltage	V_O	T_J= - 40℃	0.85	0.86	0.87	V
		T_J = +125℃	1.44	1.45	1.46	V
Start-Up Time	t_{start}				50	μs

5.12 ADC+ PGA CHARACTERISTICS

5.12.1 RECOMMENDED OPERATING CONDITIONS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Supply current(ADC+PGA)	Idd			3.5		mA
Clock frequency	Fclk			20	48	MHz

5.12.2 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	%

5.12.3 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(Note1)	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

5.12.4 PGA CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Vswing(Vpp)	Vswin		0.3		3	V
PGA on current	Ivdd_on			400		μA
PGA off current	Ivdd_off				1	μA
PGA Bandwidth	BW1	GAIN=1, -3dB		26		MHz
	BW2	GAIN=2, -3dB		13		
	BW3	GAIN=3, -3dB		9		
	BW4	GAIN=5, -3dB		6.5		
Total harmonic distortion	Thd	Fin=100khz		0.1		%
Phase shift	ps	Fin=100khz		0		degree
Gain option1	G1			1		V/V
Gain option2	G2			2		V/V
Gain option3	G3			3		V/V
Gain option4	G4			5		V/V

5.13 COMPARATOR CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
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.

5.14 RCOSC_32K CHARACTERISTICS

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
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/℃

5.15 POWER CONSUMPTION TABLE

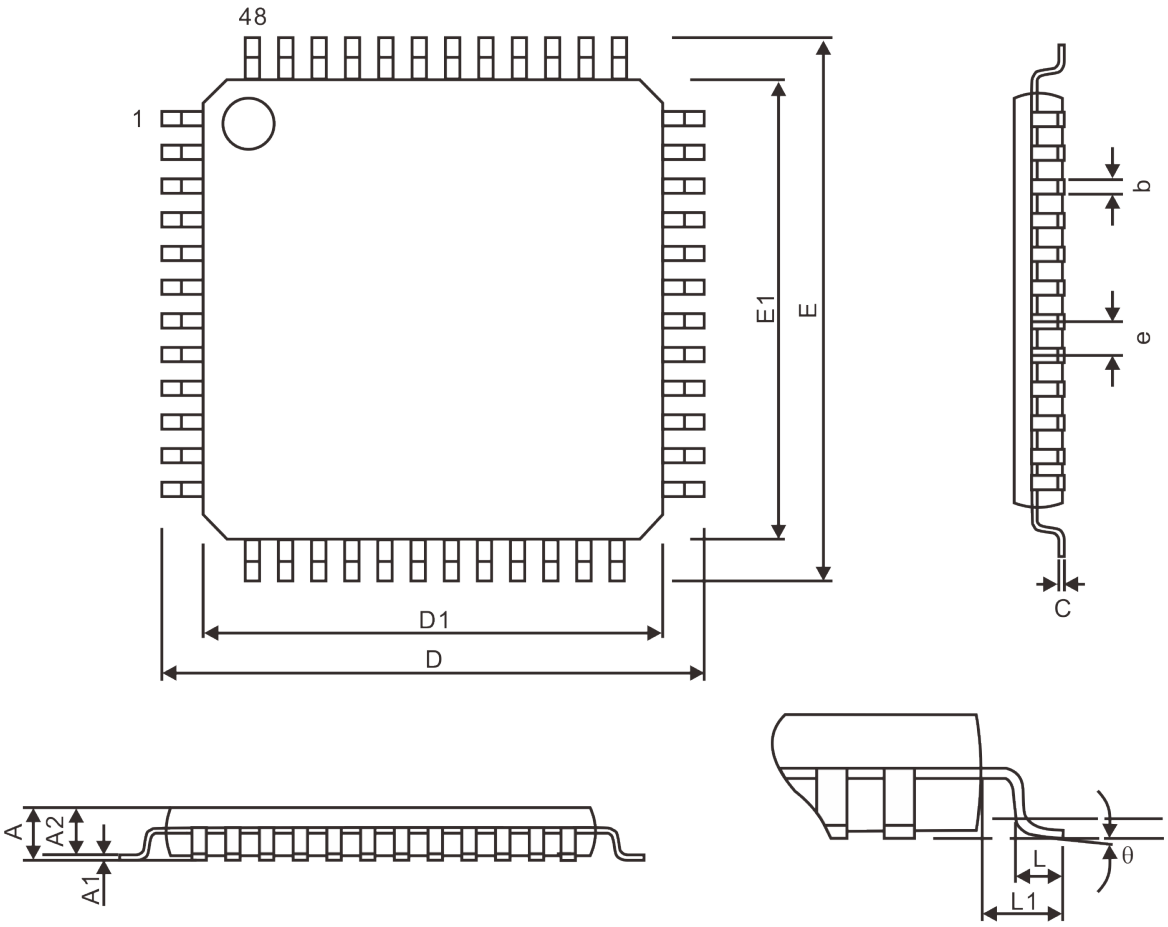
This table shows the power consumption data under three different modes: Standby Mode, Sleep Mode, and Normal Mode.

Clock Enable	Frequency	Normal		Sleep Mode, LDO ON		Unit
		All IP Clock ON	All IP Clock OFF	All IP Clock ON	All IP Clock OFF	
LSI+HSI		1.84 1.82	1.285 1.276	1.313 1.309	0.813 0.81	mA
LSE+HSI		1.83 1.81	1.288 1.276	1.312 1.309	0.815 0.813	mA
HSI		1.82 1.80	1.288 1.277	1.302 1.309	0.815 0.812	mA
HSE+HSI		2.22 2.19	1.775 1.756	1.793 1.791	1.2 1.199	mA
PLL+HSI	4 MHz	2.43 2.40	0.1763 0.1958	1.932 1.928	1.44 1.43	mA
	8 MHz	2.48 2.47	2.456 2.448	2.525 2.522	1.519 1.516	mA
	16 MHz	5.65 5.63	3.585 3.576	3.696 3.681	1.697 1.694	mA
	32 MHz	8.87 8.83	4.824 4.807	6.037 6.014	2.301 2.027	mA
	48 MHz	11.87 11.81	5.736 5.695	8.256 8.218	2.265 2.263	mA
PLL+HSE	4 MHz	2.76 2.73	2.328 2.308	2.338 2.334	1.735 1.734	mA
	8 MHz	3.845 3.833	2.916 2.903	2.853 2.848	1.83 1.826	mA
	16 MHz	6.043 6.032	4.032 4.026	4.056 4.047	2.007 2.003	mA
	32 MHz	9.3829 9.3759	5.322 5.303	6.446 6.428	2.372 2.361	mA
	48 MHz	12.345 12.296	6.309 6.289	8.706 8.695	2.624 2.62	mA

Clock enable	Standby Mode, LDO OFF			unit
	RTC enable	RTC disable	All clock off	
LSI	3.2 3.1	3.1 3.0	2.5 2.4 (LSI and LSE disable)	uA
LSE	4.5 4.4	4.5 4.4		uA

6. PACKAGE INFORMATION

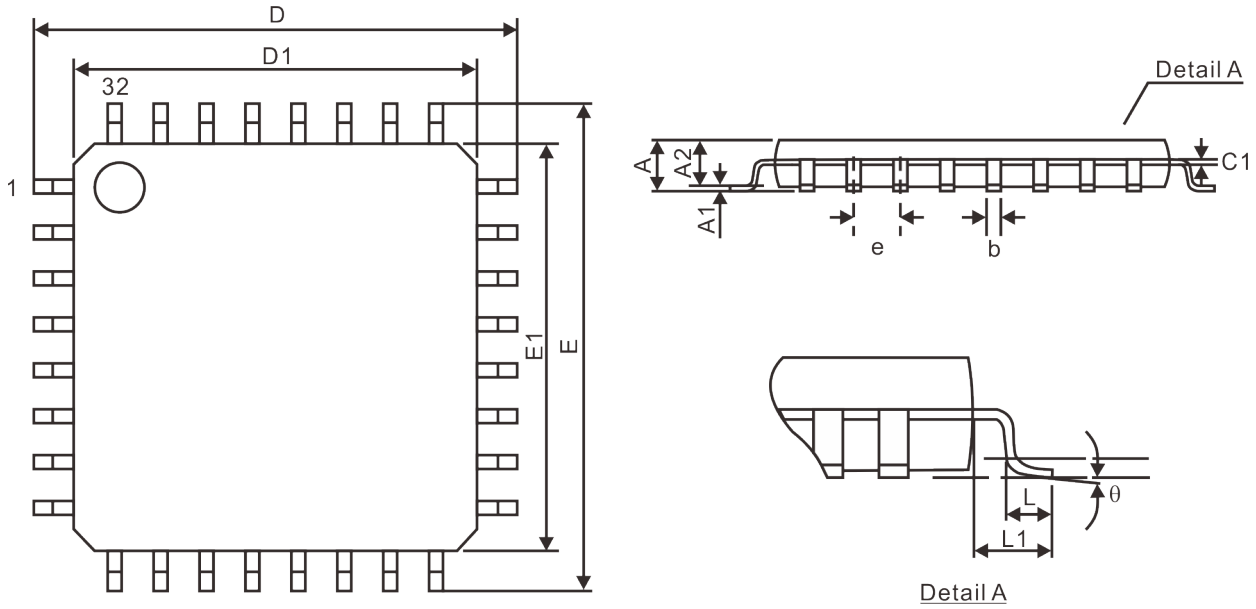
48-PIN, LQFP, 7X7



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
c	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

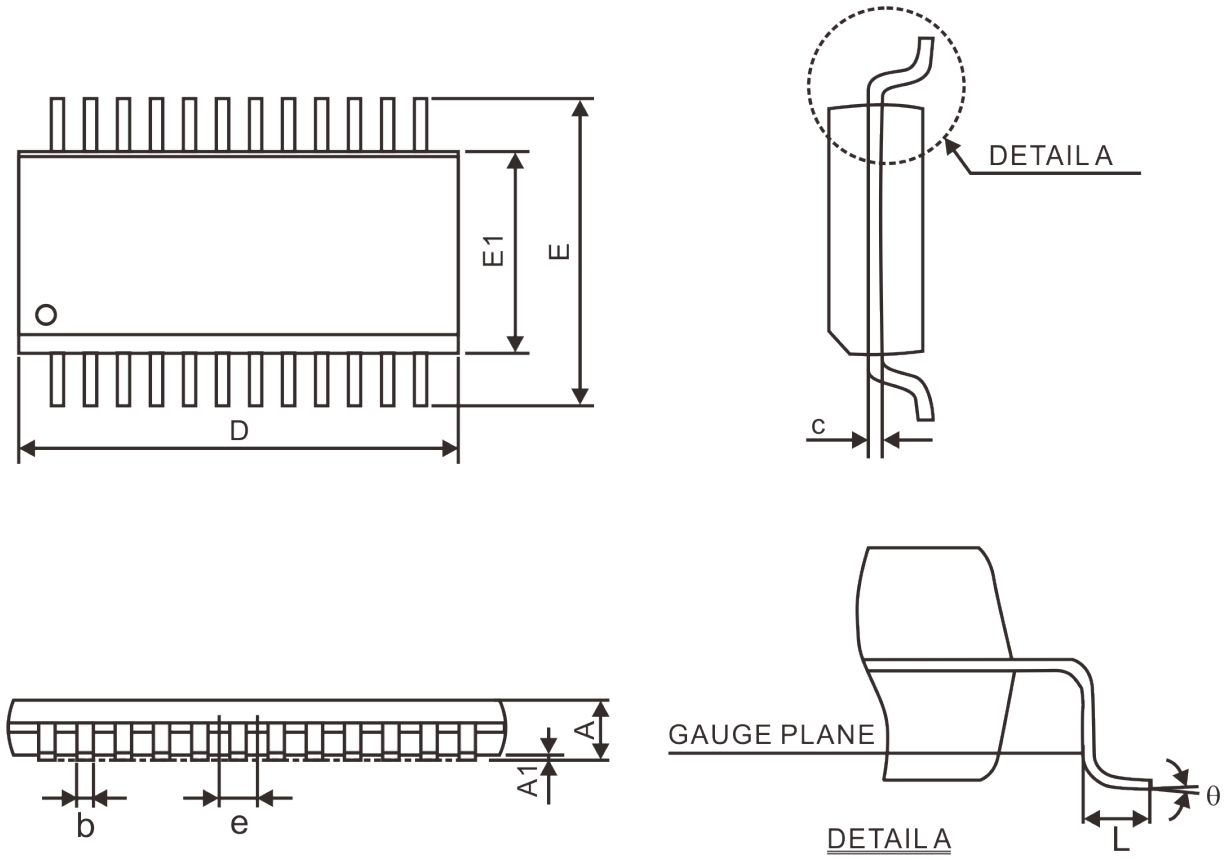
32-PIN, LQFP, 7X7



Symbol	Dimensions (mm)		
	Min.	Nom.	Max.
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
b	0.30	0.37	0.45
c1	0.09	-	0.20
D		9.00 BSC	
D1		7.00 BSC	
E		9.00 BSC	
E1		7.00 BSC	
e		0.80 BSC	
L	0.45	0.60	0.75
L1		1.00 REF	
θ	0°	3.5°	7°

Note: Refer to JEDEC MS-026 BBC

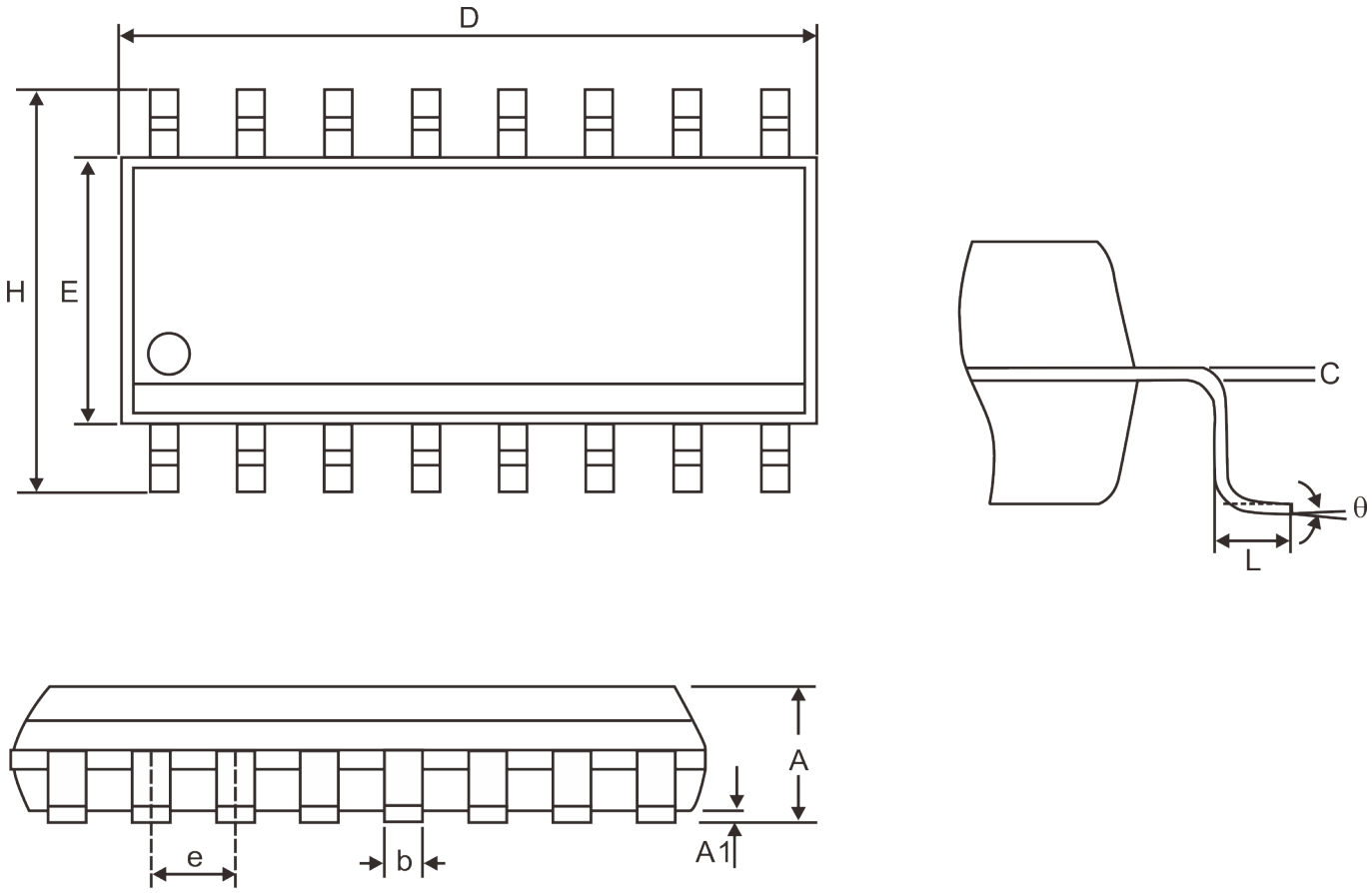
24-PIN, SSOP



Symbol	Dimensions (mm)		
	Min.	Nom.	Max.
A	-	-	1.750
A1	0.100	-	0.250
b	0.200	-	0.300
c	0.100	-	0.250
e	0.635 BSC		
D	8.66 BSC		
E	5.99 BSC		
E1	3.91 BSC		
L	0.410	-	1.270
θ	0°	-	8°

Note: Refer to JEDEC MO-137 AE

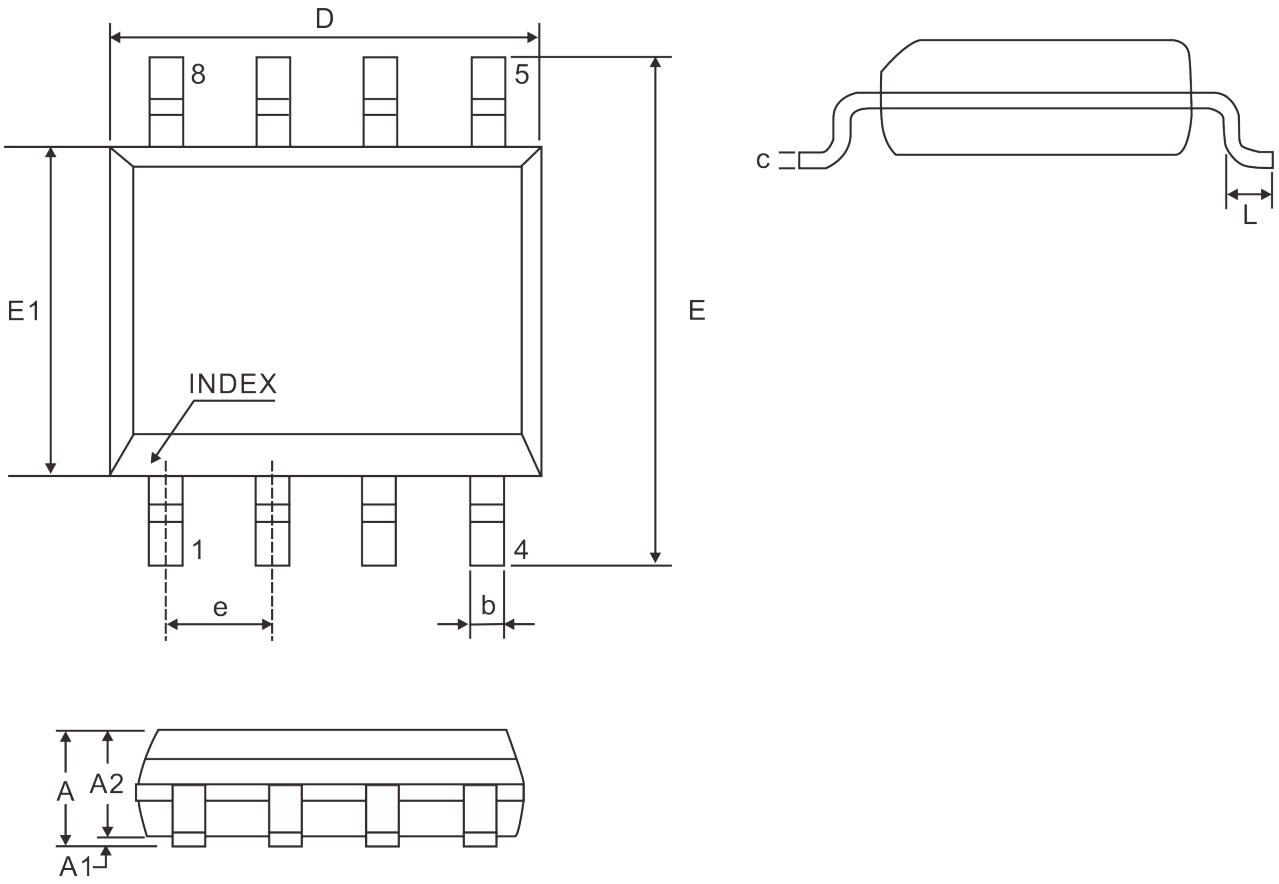
16-PIN, SOP, 150 MIL



Symbol	Dimensions (mm)		
	Min.	Nom.	Max.
A	1.35	-	1.75
A1	0.10	-	0.25
b	0.31	-	0.51
c	0.10	-	0.25
e	1.27 BSC		
D	9.90 BSC		
H	6.00 BSC		
E	3.90BSC		
L	0.45	-	1.27
θ	0°	-	8°

Note: Refer to JEDEC MS-012 AC

8-PIN, SOP, 150 MIL



Symbol	Dimensions (mm)		
	Min.	Nom.	Max.
A	-	-	1.75
A1	0.10	-	0.25
A2	1.25	-	-
b	0.31	-	0.51
c	0.10	-	0.25
D	4.90 BSC		
E	6.00 BSC		
E1	3.90 BSC		
e	1.27 BSC		
L	0.40	-	1.27
θ	0°	-	8°

Note: Refer to JEDEC MS-012 AA

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