User Guide

AB1600

Bluetooth 4.0 Single Chip

for Various Applications

Preliminary Specification

VERSION 0.10 21-Apr-2016

Airoha Technology Corp.

TABLE OF CONTENTS

List of Figures

List of Tables

Revision History

Version	Change Summary	Date	Author
0.10	Created	Apr21st, 16	

1 System Overview

1.1 General Description

AB1600 is an optimized single-chip solution which integrates baseband, radio and flash memory for game controller, mobile payment, and wearable device applications. It complies with Bluetooth version 4.2 with LE packet length extension feature. The embedded 4Mbit flash has high flexibility for customer software development. The support of 17 AIOs is used for game controller application.

1.2 Features

- Embedded 32-bit MCU with 16/48MHz clock rate
- Embedded 4Mbit Flash
- 17 AIO support (12bit)
- 30 GPIO support
- Integrate Mic ADC for voice search applications
- Integrate 1.8V switching regulator and 1.8V LDO regulator
- Ultra low power consumption for battery enabled applications

1.3 Applications

- Remote Controller
- Keyboard
- Mobile Payment
- Smart Home
- Remote Sensor

1.4 Block Diagram

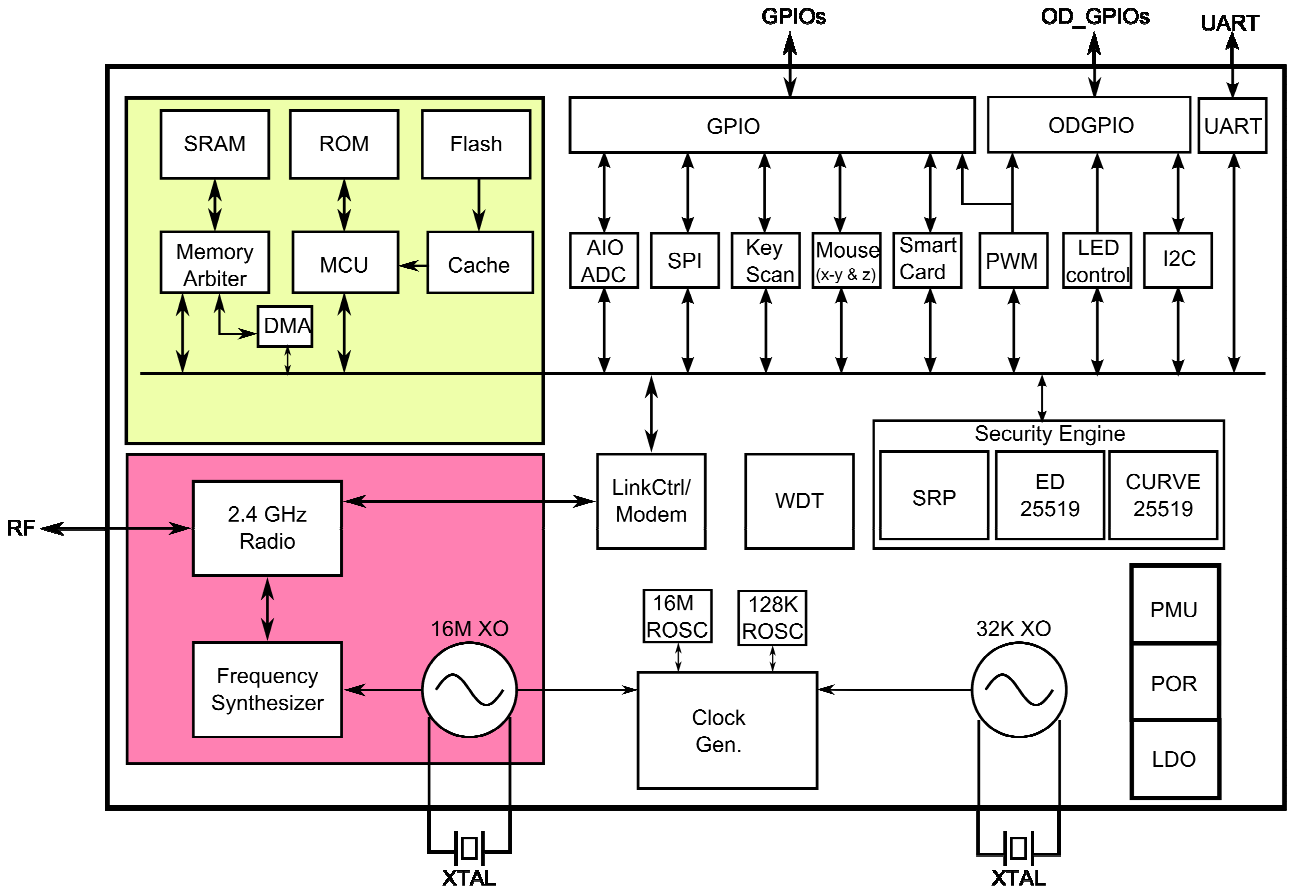


Figure 1-1 Functional Block Diagram

2 Memory Control Unit (MCU)

A low power Andes N705S MCU is embedded in AB1600 series devices. The Andes V3m has a 32 bit instruction set that delivers high density code with a small memory footprint. By using a single-cycle 32 bit multiplier, a 2-stage pipeline, and an Interrupt Controller, the Andes N705S makes program execution simple and highly efficient. The data alignment in Andes N705S implementation is Little Endian. For further information on the embedded Andes N705S MCU, see *AndesCore N705-S Data Sheet*.

3 Memory Maps

All memory blocks and registers are placed in a common memory map. There are three main categories of memory space:

- Instruction space
- Data space
- Control register space

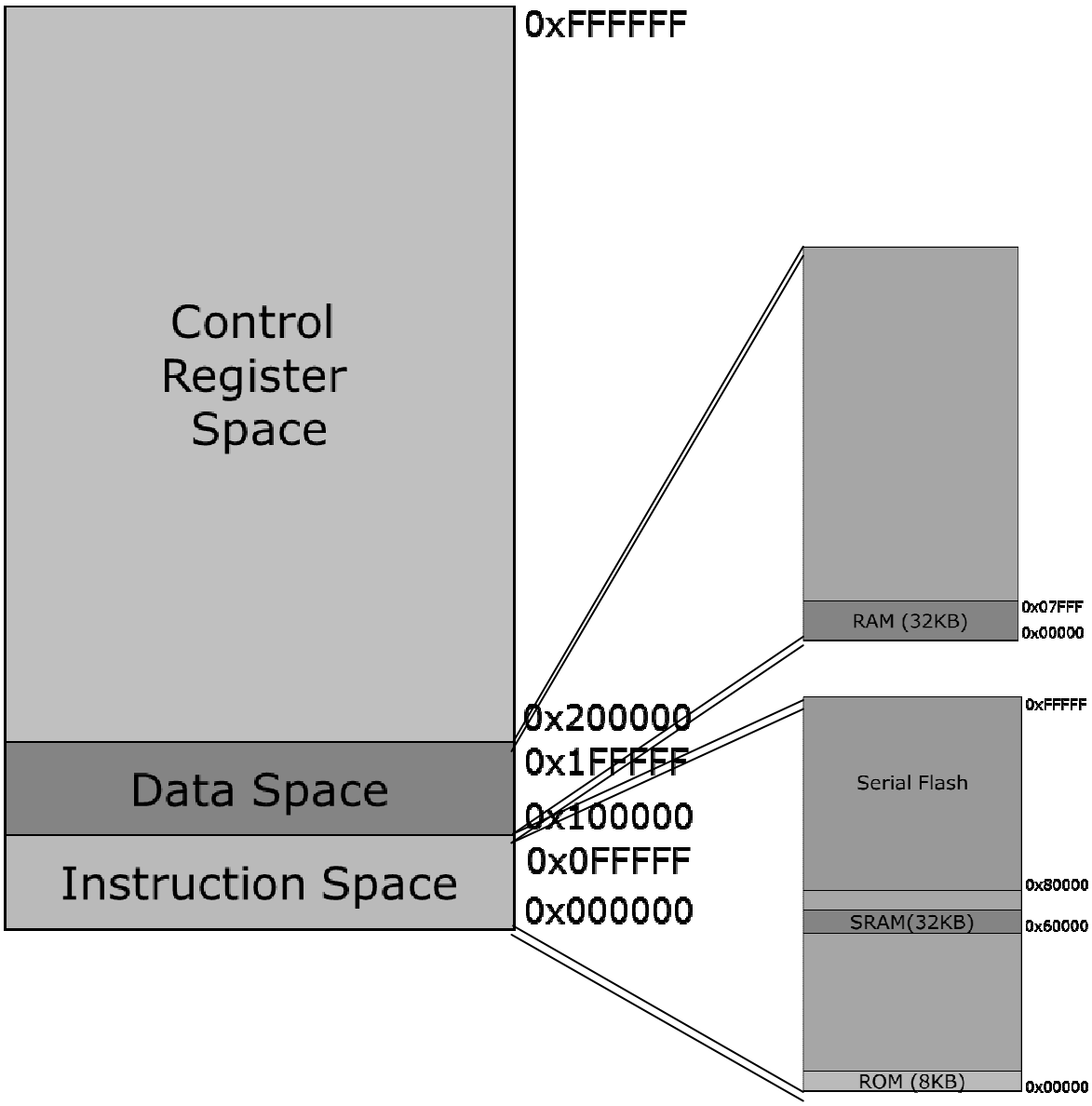


Figure 3-1 Common Memory Maps

3.1 Instruction space

The instruction space is normally used for storing the program executed by MCU, but can also be used for storing data constants that are retained when chip loses power.

The various memory categories can have one of the following memory types:

• Volatile memory (VM)

• Non-volatile memory (NVM)

Volatile memory is a type of memory that will lose its contents when the chip loses power. This memory type can be read/written an unlimited number of times by the MCU.

Non-volatile memory is a type of memory that can retain stored information even when the chip loses power. This memory type can be read for an unlimited number of times by the MCU, but have the restrictions on the number of times it can be written and erased and also on how it can be written. Writing to non-volatile memory is managed by the Non-volatile Memory Controller (NVMC).

3.1.1 ROM space

The system offers ROM space contains the system boot sequence.

3.1.2 RAM space

The system offers 32KB RAM space for executing program. When using this region, loading program from peripheral or flash first to execute program. The 32KB region is shared with data space 32KB

3.1.3 Flash space

The system offers 512KB flash space to store the program and data when the system loses power.

3.2 Data space

The Data RAM 32KB region is located in the SRAM segment of the System Address Map. It is shared with RAM space so it is possible to execute code from this region.

3.3 Control register space

The control register space is peripheral registers used for interfacing to peripheral units such as the timers, the SPI, the UART, the ADC, and so on.

Most peripherals feature an ENABLE register. Unless other specified in the relevant chapter, the peripheral registers shall be configured prior to enabling the peripheral.

4 Platform Introduction

4.1 Power domain

The PMU is designed in AB1600 for the power management tasks. The PMU controls the Buck and LDO Regulator power in sequence. During general operations, MCU may get into sleep mode for power saving. During power saving, the PMU monitors the keys and wakes up the MCU if one of the keys is pressed. PMU also monitors the battery voltage and reports it to MCU.

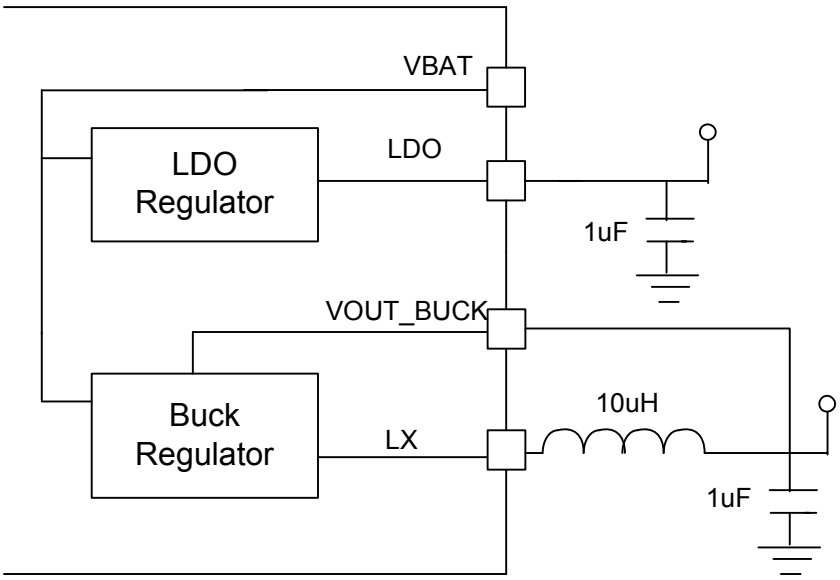


Figure 4-1 Buck_LDO Regulator Circuit

The Buck Regulators are embedded to convert VBAT to 1.8V voltage to supply AB1600. The block shows the buck circuit with LC component. The LDO Regulators are embedded to convert VBAT to 1.8V voltage to supply AB1600.

4.2 System behaviors

According to different power and clock on/off states, AB1600 offers four power states for different application, including normal mode, sleep mode, deep sleep mode and shutdown mode.

4.2.1 Normal mode

When chip power on, it enters in normal mode. At this mode, the MCU, peripheral platform ‘s power and clock are on. MCU can execute program from instruction space and interconnect with outside by peripherals or GPIO.

4.2.2 Sleep mode

AB1600 offers a low power mode: sleep mode. At this mode, the MCU and peripheral platform’s clock are off. The system retains PMU clock to obtain low standby current. System can be woken up from any interrupt, internal events or external events. When the system is woken up, the system returns to normal mode.

4.2.3 Deep sleep mode

There is another low power mode called deep sleep mode. At this mode, the MCU and peripheral platform’s power are off. The power of PMU is turning on to get lower standby current. At deep sleep mode, the standby current is lower, but the startup sequence will be slower. The system can be woken up from deep sleep mode by external events such as GPIOs or MMI timer, etc.. When the system returns to normal mode, the flag set in deep sleep mode will let the software programmers know the systems are woken up from boot or deep sleep mode.

4.2.4 Shutdown mode

The system can enter shutdown mode for the lowest power. The system offers LPO timer for waking up from shutdown mode.

4.2.5 Soft reset

A soft reset is generated when software programmers need to reset the whole chip by controlling the soft _rst registers.

4.2.6 Control registers

The PMU control register is defined as following:

PMU control register: 0x205000

31 9	8	7	6	5	4	3 1	0
	soft_rst		sleep	deepsleep	pwr_off		mcu_rdy

Name	Bit	Write/Read	Reset value	Description
soft_rst	[8]	W	N/A	Writing 1 to this bit will reset the chip. Automatically clear.
sleep	[6]	W	N/A	PMU sleep enable. Writing 1 to this bit will let PMU enter sleep mode. Automatically clear.
deepsleep	[5]	W	N/A	PMU deep sleep enable. Writing 1 to this bit will let PMU enter deep sleep mode. Automatically clear. Note:Set 3-wire register 0x3c "deepsleep_flag" to 1 before entering deepsleep.
pwroff	[4]	W	N/A	Writing 1 to this bit will power off the chip. Automatically clear.
reserve	[3:1]	RW	0	reserve
mcu_rdy	[0]	RW	0	Need to write 1 to this bit, or PMU will enter deep sleep mode/power off mode, which depends on the previous state of PMU

4.3 System Frequency

AB1600 offers different clock combinational for system frequency including 16MHz provided by RCO16M or xtal 16M, 1M/4M/8M divided by previous 16MHz, or 24MHz/48MHz from PLL. Customers can choose different system clocks depend on different applications

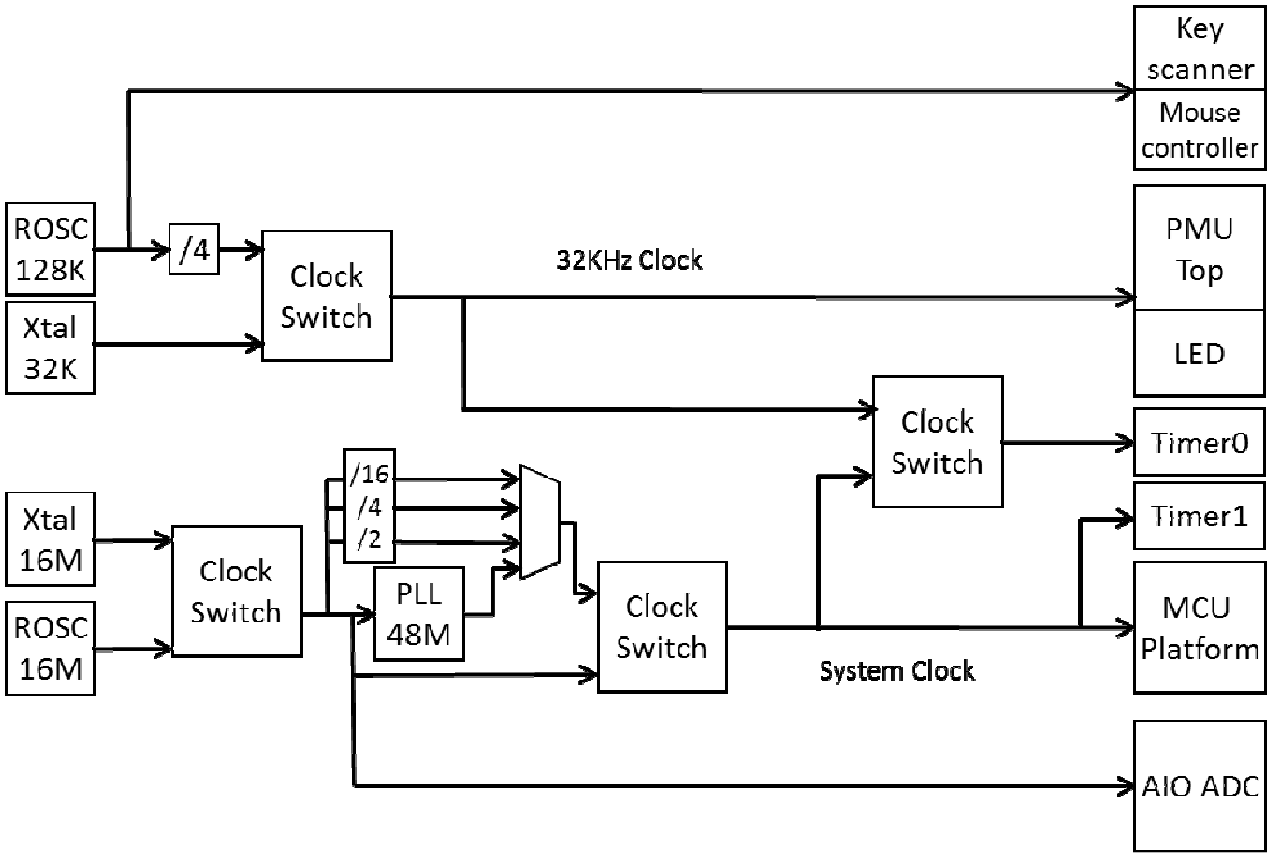


Figure 4-2 Clock Diagram

4.4 Non-volatile Memory Controller (Flash control)

There is a non-volatile memory controller to operate embedded flash for erasing or programming the flash contents. The function includes block-erasing, whole flash erasing, byte-programming, page programming, byte-reading, and page-reading, etc.

(A page is 256 bytes)

(1) Flash command register: 0x202000

31 5	4 0
	command

Name	Bit	Write/Read	Reset value	Description
command	[4:0]	WR	5'b000000	Choose the flash controller command [4:0]: 5'b00000: byte READ (io mode selected by configure register[2:1] (SFR address: 20h)) 5'b01000: READ VERIFY (io mode selected by configure register[2:1] (SFR address: 20h)) 5'b11000: page READ (io mode selected by configure register[2:1] (SFR address: 20h)) 5'b00100: SE (Sector erase) 5'b00101: BE32 (block erase with 32K byte each) 5'b00110: BE64 (block erase with 64K byte each) 5'b00111: CE (chip erase) 5'b01100: PP (byte program) (io mode selected by configure register[2:1] (SFR address: 20h)) 5'b01101: PP (page program) (io mode selected by configure register[2:1] (SFR address: 20h))

(2) Flash address register: 0x202004

31	30 24	23 0
address_sel		flash address

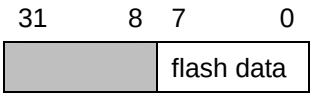
Name	Bit	Write/Read	Reset value	Description
address sel	[31]	WR	1'b0	choose address[23:0] is flash absolute address or MCU space address 1'b1: mcu space address (={4'b0000,instruction space[19:0]}) 1'b0: flash absolute address
address	[23:0]	WR	24'b0	flash address [23:0]

(3) Flash start operating register: 0x202008

31 2	0
	start

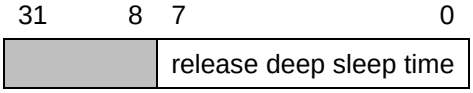
Name	Bit	Write/Read	Reset value	Description
start / busy	[0]	W1AC/R	0	wrte 1 to start pulse [0]: after write 1 to start pulse, read [0] until 0 to know start pulse finished

(4) Flash data register: 0x20200C



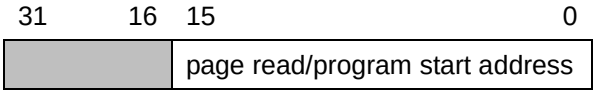
Name	Bit	Write/Read	Reset value	Description
flash data	[7:0]	WR	8'b0	write in byte program data / read out byte read data

(5) Flash release deepsleep time register: 0x202014



Name	Bit	Write/Read	Reset value	Description
release deep sleep time	[7:0]	WR	8'd40	the register storing the number N by calculating from t(us) (release from deep sleep time without electronic signature read) and system clk N= system clk(MHz) x tres1(us) /16 (for example: t=10us, system clk =64MHz, then N=64x10/16=40=7'b00101000)

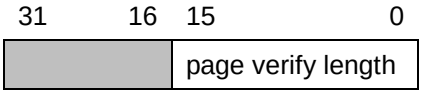
(6) Page read or program start address register: 0x202018



Name	Bit	Write/Read	Reset value	Description
	[15:2]	WR		

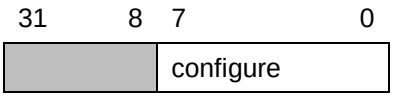
	[1:0]	R		

(7) Page verify length register: 0x20201C



Name	Bit	Write/Read	Reset value	Description
page verify length	[15:0]	WR	16'b0	page verify length

(8) Flash configure register: 0x202020



Name	Bit	Write/Read	Reset value	Description
configure register	[7:0]	WR	8'h11111001	[0]: reserve (need set 1) [2:1]: choosing the io mode 2'b00: 1-io mode (program-1/read-1) 2'b01: 2-io mode (program-1/read-2) 2'b10: 4-io mode (program-1/read-4) 2'b11: 4-io mode (program-4/read-4) [6:3]: reserve (need set 1) number of active pins, x:opcode, y:address, z:data (ex: MXIC=1'b1, WINBOND=1'b0) [7]: flash deep sleep enable (soc in sleep and deep sleep, serial flash will enter deep sleep) 1'b0: disable 1'b1: enable

(9) Flash page read or program CRC register: 0x202028

31 8 7 0

	CRC value

Name	Bit	Write/Read	Reset value	Description
CRC result	[7:0]	R	8'b0	page read/program data CRC result

(10)Flash operating pattern match register: 0x202040

3 1 0

pattern match

Name	Bit	Write/Read	Reset value	Description
pattern match	[31:0]	WR	32'h5a5aa5a5	"pattern match for start pulse (key = 32'hdb0061ba)

4.5 Software interrupt

The Andes N705S offers software reset instruction for use as software interrupts.

4.6 Debugger

AB1600 devices support the three wire serial debug (TWSD) interface from Andes N705S. The interface has three lines; SCLK, SDIO and nRESET. The SCLK, SDIO and nRESET pin all have an internal pull up resistor. Debug interface mode is initiated by mode strapping.

4.7 General-Purpose Input/ Output (GPIO)

The AB1600 offers 30 (GPIO0~29) general-purpose input/output (GPIOs) for use. All the peripheral interfaces are shared with these 30 GPIOs.

4.7.1 Digital IO

Every GPIO is a digital IO. By setting control registers, GPIOs can be configured to peripheral interfaces such as smart card, timers, SPI, UART, etc. GPIOs also can be controlled by MCU setting control registers.

(1) GPIO Group selection register: 0x204000

15	14 12	11	10 8	7	6 4	3	2 0
	grp_sel_11		grp_sel_10		grp_sel_01		grp_sel_00
31	30 28	27	26 24	23	22 20	19	18 16
	grp_sel_41		grp_sel_40		grp_sel_21		grp_sel_20

Name	Bit	Write/Read	Reset value	Description
grp_sel_00[2:0]	[2:0]	WR	3'b000	Control GPIO MUX for GPIO 26,27,28,29 3'b000: Determined by sig_sel_xx[3:0], where xx is 00_29 for each GPIO 3'b001: SPI Master/Slave 3'b010: timer, determined by timer_sel[1:0] 3'b011: N/A 3'b100: mouse X-Y 3'b101: smart_card
grp_sel_01[2:0]	[6:4]	WR	3'b000	Control GPIO MUX for GPIO 24,25 3'b000: Determined by sig_sel_xx[3:0], where xx is 00_29 for each GPIO 3'b001: N/A 3'b010: timer, determinec by timer_sel[1:0] 3'b011: UART_2 3'b100: mouse Z 3'b101: N/A
grp_sel_10[2:0]	[10:8]	WR	3'b000	Control GPIO MUX for GPIO 20,21,22,23 3'b000: Determined by sig_sel_xx[3:0], where xx is 00_29 for each GPIO 3'b001: SPI Master/Slave 3'b010: timer, determined by timer_sel[1:0] 3'b011: N/A 3'b100: mouse X-Y 3'b101: smart_card

grp_sel_11[2:0]	[14:12]	WR	3'b000	Control GPIO MUX for GPIO 18,19 3'b000: Determined by sig_sel_xx[3:0], where xx is 00_29 for each GPIO 3'b001: N/A 3'b010: timer, determined by timer_sel[1:0] 3'b011: UART_2 3'b100: mouse Z 3'b101: N/A
grp_sel_20[2:0]	[18:16]	WR	3'b000	Control GPIO MUX for GPIO 14,15,16,17 3'b000: Determined by sig_sel_xx[3:0], where xx is 00_29 for each GPIO 3'b001: SPI Master/Slave 3'b010: timer, determined by timer_sel[1:0] 3'b011: N/A 3'b100: mouse X-Y 3'b101: smart_card
grp_sel_21[2:0]	[22:20]	WR	3'b000	Control GPIO MUX for GPIO 12,13 3'b000: Determined by sig_sel_xx[3:0], where xx is 00_29 for each GPIO 3'b001: N/A 3'b010: timer, determined by timer_sel[1:0] 3'b011: UART_2 3'b100: mouse Z 3'b101: N/A
grp_sel_40[2:0]	[26:24]	WR	3'b000	Control GPIO MUX for GPIO 0,1,2,3 3'b000: Determined by sig_sel_xx[3:0], where xx is 00_29 for each GPIO 3'b001: SPI Master/Slave 3'b010: timer, determined by timer_sel[1:0] 3'b011: N/A 3'b100: mouse X-Y 3'b101: smart_card

grp_sel_41[2:0]	[30:28]	WR	3'b000	Control GPIO MUX for GPIO 4,5 3'b000: Determined by sig_sel_xx[3:0], where xx is 00_29 for each GPIO 3'b001: N/A 3'b010: timer, determined by timer_sel[1:0] 3'b011: UART_2 3'b100: mouse Z 3'b101: N/A

(2) GPIO0~7 signal selection register: 0x204004

15 12	11 8	7 4	3 0
sig_sel_03	sig_sel_02	sig_sel_01	sig_sel_00
31 28	27 24	23 20	19 16
sig_sel_07	sig_sel_06	sig_sel_05	sig_sel_04

Name	Bit	Write/Read	Reset value	Description
sig_sel_00	[3:0]	WR	4'b0000	sig_sel_XX: Select output for GPIOXX if the related grp_sel is 0. 4'd0: GPIO controlled by MCU 4'd1 : PWM1 4'd2 : PWM2 4'd3 : PWM3 4'd4 : PWM4 4'd5 : xo16m 4'd6 :UART_NRTS 4'd7 : IR32k 4'd8 : xo32k
sig_sel_01	[7:4]	WR	4'b0000	
sig_sel_02	[11:8]	WR	4'b0000	
sig_sel_03	[15:12]	WR	4'b0000	
sig_sel_04	[19:16]	WR	4'b0000	
sig_sel_05	[23:20]	WR	4'b0000	
sig_sel_06	[27:24]	WR	4'b0000	
sig_sel_07	[31:28]	WR	4'b0000	

(3) GPIO8~15 signal selection register: 0x204004

15 12	11 8	7 4	3 0
sig_sel_11	sig_sel_10	sig_sel_09	sig_sel_08
31 28	27 24	23 20	19 16
sig_sel_15	sig_sel_14	sig_sel_13	sig_sel_12

Name	Bit	Write/Read	Reset value	Description
sig_sel_08	[3:0]	WR	4'b0000	sig_sel_XX: Select output for GPIOXX if the related grp_sel is 0. 4'd0: GPIO controlled by MCU 4'd1 : PWM1 4'd2 : PWM2 4'd3 : PWM3 4'd4 : PWM4 4'd5 : xo16m 4'd6 :UART_NRTS 4'd7 : IR32k 4'd8 : xo32k
sig_sel_09	[7:4]	WR	4'b0000	
sig_sel_10	[11:8]	WR	4'b0000	
sig_sel_11	[15:12]	WR	4'b0000	
sig_sel_12	[19:16]	WR	4'b0000	
sig_sel_13	[23:20]	WR	4'b0000	
sig_sel_14	[27:24]	WR	4'b0000	
sig_sel_15	[31:28]	WR	4'b0000	

(4) GPIO16~23 signal selection register: 0x204008

15 12	11 8	7 4	3 0
sig_sel_19	sig_sel_18	sig_sel_17	sig_sel_16
31 28	27 24	23 20	19 16
sig_sel_23	sig_sel_22	sig_sel_21	sig_sel_20

Name	Bit	Write/Read	Reset value	Description
sig_sel_16	[3:0]	WR	4'b0000	sig_sel_XX: Select output for GPIOXX if the related grp_sel is 0. 4'd0: GPIO controlled by MCU 4'd1 : PWM1 4'd2 : PWM2 4'd3 : PWM3 4'd4 : PWM4 4'd5 : xo16m 4'd6 :UART_NRTS 4'd7 : IR32k 4'd8 : xo32k
sig_sel_17	[7:4]	WR	4'b0000	
sig_sel_18	[11:8]	WR	4'b0000	
sig_sel_19	[15:12]	WR	4'b0000	
sig_sel_20	[19:16]	WR	4'b0000	
sig_sel_21	[23:20]	WR	4'b0000	
sig_sel_22	[27:24]	WR	4'b0000	
sig_sel_23	[31:28]	WR	4'b0000	

(5) GPIO24~29 signal selection register: 0x20400C

15 12	11 8	7 4	3 0
sig_sel_27	sig_sel_26	sig_sel_25	sig_sel_24
31 28		23 20	19 16
		sig_sel_29	sig_sel_28

Name	Bit	Write/Read	Reset value	Description
sig_sel_24	[3:0]	WR	4'b0000	sig_sel_XX: Select output for GPIOXX if the related grp_sel is 0. 4'd0: GPIO controlled by MCU 4'd1 : PWM1 4'd2 : PWM2 4'd3 : PWM3 4'd4 : PWM4 4'd5 : xo16m 4'd6 :UART_NRTS 4'd7 : IR32k 4'd8 : xo32k
sig_sel_25	[7:4]	WR	4'b0000	
sig_sel_26	[11:8]	WR	4'b0000	
sig_sel_27	[15:12]	WR	4'b0000	
sig_sel_28	[19:16]	WR	4'b0000	
sig_sel_29	[23:20]	WR	4'b0000	

(6) GPIO control register: 0x204040

31 30	29 … … … . 0	
	GPIO29	…….. GPIO0

Name	Bit	Write/Read	Reset value	Description
GPIOX	[29:0]	WR	30'b0	Read from GPIO inputs or write to GPIO outputs if the corresponding gpio_oe ((7)GPIO oe register) is enabled

(7) GPIO output enable register: 0x20404C

31 30	29 … … … . 0		
	GPIO29_oe	……..	GPIO0_oe

Name	Bit	Write/Read	Reset value	Description
GPIOX_oe	[29:0]	WR	30'b0	GPIO[29:0] output enable to change the GPIO[29:0] as output ports

(8) GPIO read invert register:0x204050

31	30	29	… … … .	0
	GPIO29_inv_en	……..		GPIO0_inv_en

Name	Bit	Write/Read	Reset value	Description
GPIOX_inv_en	[29:0]	WR	30'b0	read from GPIOX input for inv or not

(9) Uart1 ncts selection register: 0x204018

31	13	12	8	7	0
		uart1_ncts_sel			

Name	Bit	Write/Read	Reset value	Description
uart1_ncts_sel[4:0]	[12:8]	WR	5'b00000	Select UART_NCTS from the selected GPIO. 5'd0: GPIO0 5'd1: GPIO1 … and so on. The valid value is from 0 to 29.

(10) Timer selection register: 0x204020

31	2	1	0
		timer1_sel	timer0_sel

Name	Bit	Write/Read	Reset value	Description
timer0_sel	[0]	WR	1'b0	select timer0 from grp_sel_xx if grp_sel_xx = 2 1'b0: grp_sel_0x 1'b1: grp_sel_1x
timer1_sel	[1]	WR	1'b0	select timer1 from grp_sel_xx if grp_sel_xx = 2 1'b0: grp_sel_4x 1'b1: grp_sel_2x

4.7.2 Analog IO

The part of GPIOs can be used for analog IOs. When users set GPIOs to analog IOs, the digital function will be disable.

4.7.3 Open drain IO

There are 3 open drain IOs (ODGPIO) in the AB1600. These pins can be used for LED or I2C applications.

(1) OD_GPIO Group selection and signal selection register: 0x204014

31 16	15 12	11 8	7 4	3 1	0
	od_sig_sel_2	od_sig_sel_1	od_sig_sel_0		od_grp_sel_0

Name	Bit	Write/Read	Reset value	Description
od_grp_sel	[0]	WR	1'b0	Control OD GPIO MUX for OD_GPIO 0,1 1'b0: Determined by od_sig_sel_X[3:0], where X is 0~2 for each OD GPIO 1'b1:I2C
od_sig_sel_0	[7:4]	WR	4'b0000	od_sig_sel_**X**: Select output for OD GPIO **X** if the od_grp_sel is 0. 4'd1 : PWM1 4'd2 : PWM2 4'd3 : PWM3 4'd4 : PWM4 4'd5 : LED04'd6 : LED1 Others: OD GPIO controlled by MCU
od_sig_sel_1	[11:8]	WR	4'b0000	
od_sig_sel_2	[15:12]	WR	4'b0000	

(2) ODGPIO control register: 0x204048

31 3	2	1	0
	ODGPIO2	ODGPIO1	ODGPIO0

Name	Bit	Write/Read	Reset value	Description
ODGPIOX	[2:0]	WR	3'b000	Read from od_gpio inputs or write to od_gpio outputs if the corresponding od_gpio_oe((3)ODGPIO oe) is enabled

(3) ODGPIO output enable register: 0x20404C

31	3	2	1	0
	ODGPIO2_oe	ODGPIO1_oe	ODGPIO0_oe	

Name	Bit	Write/Read	Reset value	Description
ODGPIOX_oe	[2:0]	WR	5'b000	OD_GPIO[2:0] output enable to change the OD_GPIO[2:0] as output port

5 Universal Asynchronous Receiver/ Transmitter (UART)

5.1 Overview

The UART provides asynchronous serial interfaces. The UART uses two-wire or four-wire interface, which consists of uart_tx , uart_rx, and optionally uses uart_cts_n and uart_rts_n. There are FIFOs of depth 8 in both Tx and Rx. The architecture is shown in .

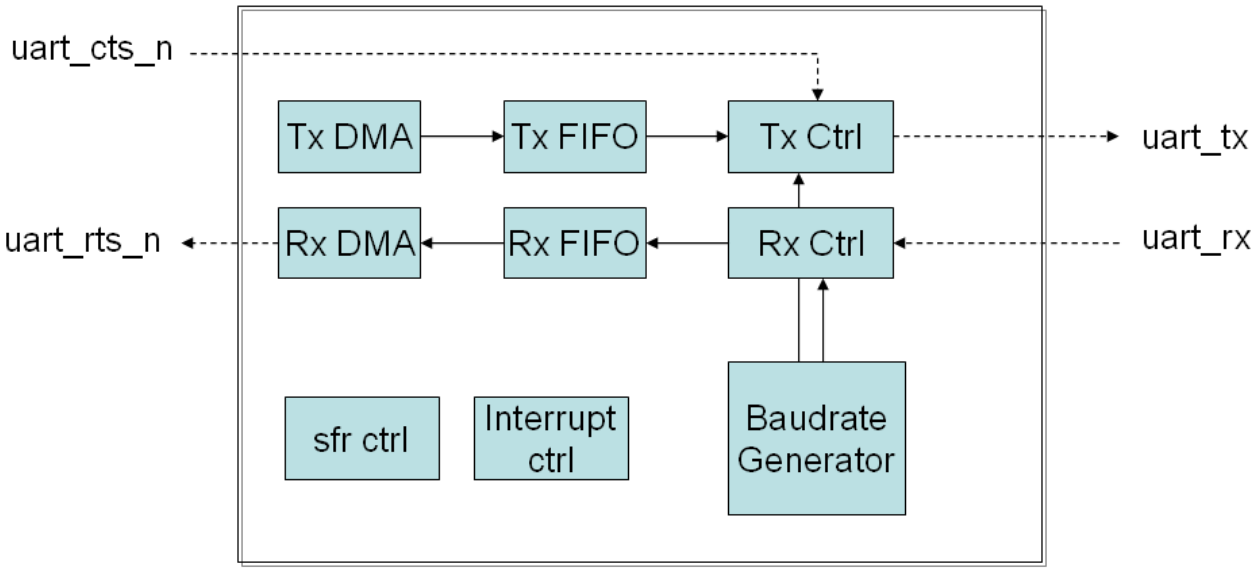


Figure 5-1 The architecture of UART

5.2 Function Descriptions

5.2.1 Interface Modes

The interface is selected by Flow Ctrl Enable(FCE) in FCR. When FCE is 1'b1, four-wire interface is selected, otherwise, two-wire interface is selected. The four-wire interface supports hardware flow control and support in FIFO mode or DMA mode only (see section [3.Operation Modes]). The hardware flow control can detect uart_cts_n to stop/continue transmitting and control uart_rst_n to allow/disallow the connected device to transmit. The two-wire interface uses only uart_tx and uart_rx to support normal Tx and Rx without hardware flow control.

5.2.2 Operation Modes

The UART supports 3-modes: 1.single-byte mode 2.FIFO mode 3.DMA mode. In single-byte mode, software accesses RBR and THR directly. Once the RBR is ready, software need to read it before next character is received. When the THR is written by MCU, software need to wait for the transmission done before writing next one. In FIFO mode, it's similar to single-byte mode, except for 8 bytes buffered registers both in Tx and Rx, which software can read RBR or write THR continuously. In DMA mode, software DO NOT access RBR and THR directly, on the contrary, there is memory interface for software to control. The memory interface uses the same FIFO path in FIFO mode. The mode is determined by FIFO enable and DMA enable in FIFO Control Register(FCR), shown in .

fifo_en	dma_en	Mode
0	0	single-byte mode
1	0	FIFO mode
1	1	DMA mode

Table 5-1 The modes of UART

Note that when software writes dma_en by 1'b1, the fifo_en is also written by 1'b1.

5.2.3 Baudrate

There are two registers to configure the baudrate of UART, integer part (DLI) and fractional part (DLF). The baudrate is calculated as fclk / (DLI + DLF/4) / 4, where fclk is the frequency fo clk of UART. The fclk is either 16M or 48M, refer to section 4.3. The supported baudrates are listed in Table 5-2 :

16M				48M			
Baudrate	DLI	DLF	error(%)	Baudrate	DLI	DLF	error(%)
115200	34	3	-0.08%	115200	104	1	-0.08%
230400	17	1	0.64%	230400	52	0	0.16%
460800	8	3	-0.79%	460800	26	0	0.16%
614400	6	2	0.16%	614400	19	2	0.16%
921600	4	1	2.12%	921600	13	0	0.16%
1228800	3	1	0.16%	1228800	9	3	0.16%
				2457600	5	0	-2.34%
				3000000	4	0	0.00%

Table 5-2 The baudrates of AURT

5.2.4 Format of character

The format of character is controlled by Line Control Register (LCR). Refer to the section 5.3.

5.2.5 Timeout

The timeout function is supported only in FIFO mode and DMA mode. When there is data in FIFO (FIFO mode) or memory (DMA mode) and no access of data (hardware doesn't receive new data and software doesn't read the old data), the timer counts. If the value of the timer reaches certain period, UART will assert interrupt (if the interrupt is enabled). In FIFO mode, the timeout period is fixed as 64 bits (bauds). In DMA mode, the timeout period is defined as DMA_Character_Timeout * 16 bits(bauds).

5.2.6 Data Trigger

The data trigger function is supported only in FFIO mode and DMA mode.

When the data in FIFO **reaches** the trigger level, UART will assert interrupt RDAI in FIFO mode (if the interrupt is enabled). The trigger level is determined by RTL in FIFO Control Register(FCR).

When the valid data in memory **exceeds** the trigger level, UART will assert interrupt RDAI in DMA mode (if the interrupt is enabled). The trigger level is determined by DMA_TriggerLevel.

5.2.7 DMA

The DMA is supported in DMA mode only. The DMA moves the data (characters) between memory and FIFO by UART.

The DMA Tx is a one-shot process. To use DMA Tx, software needs to prepare the data in memory setting the base address(DMA_TxAddr) and length(DMA_TxLength). The DMA Tx will automatically start after writing DMA_TxLength, that is, the DMA_TxAddr must be set in advance. The UART TX DMA will stop when DMA_TxLength bytes of data are transferred to FIFO.

To use DMA Rx, software needs defines a memory block with base address(DMA_RxAddr) and length(DMA_RxLength). Then hardware moves the received data from FIFO to the given memory. Unlike DMA Tx, the memory of DMA Rx is regarded as a ring buffer, that is, software needs to maintain the memory block by monitoring DMA_RxCurentWAddr (wptr in) and updating DMA_RxCurentRAddr(rptr in).

DMA_RxCurentWAddr is the current value of DMA Rx write pointer. DMA_RxCurrentRAddr is the current value of RX DMA memory read pointer. If DMA_RxCurentWAddr equals to DMA_RxCurentRAddr, memory is empty. Otherwise, memory data are ready from DMA_RxCurrentRAddr to (DMA_RxCurrentWAddr - 1) and not valid from DMA_RxCurrentWAddr to (DMA_RxCurrentRAddr -1). There is an example in .

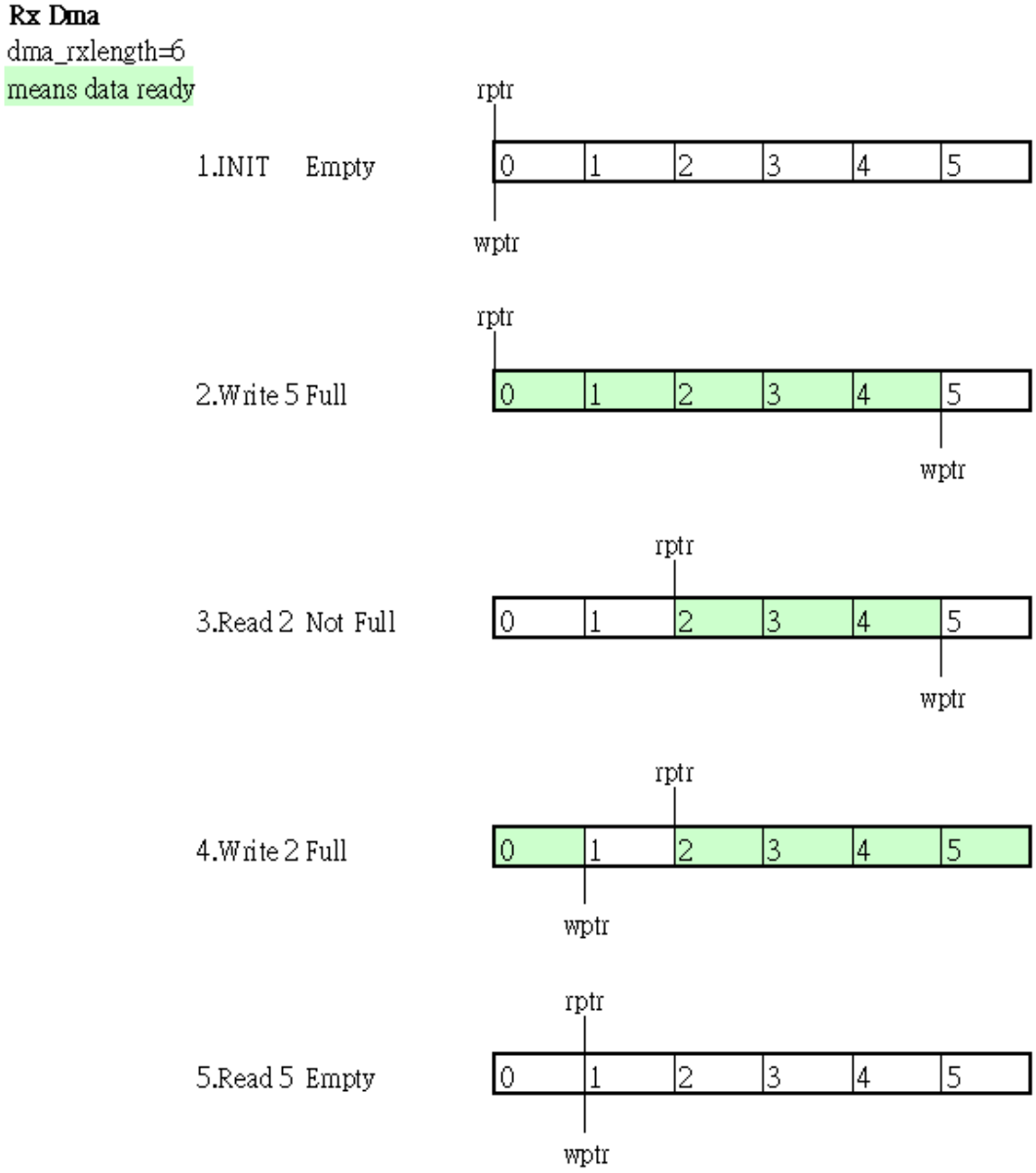


Figure 5-2 An example of an operation of RX DMA.

5.2.8 Interrupts

The UART prioritizes interrupts into four levels and records these in the Interrupt Identification Register. The priority of four levels interrupt conditions are Receiver Line Status: Received Data Ready, Transmitter Holding Register Empty, and MODEM Status. When the MCU accesses the IIR, the UART freezes all interrupts and indicates the highest priority pending interrupt to the MCU. While this MCU access is occurring, the UART records the new interrupt but doesn't change its current indication until the access complete. The interrupts are listed in Table 5-3 .

IIR Register							
B3	B2	B1	B0	Priority	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	0	0	1	NA	None	None	None
0	1	1	0	1st	Receiver Line Status (LSI)	OE or PE or FE or BI	Write 1'b1 to corresponding bit of Line Status Register.
0	1	0	0	2nd	Received Data Available (RDAI)	In DMA mode: Unread data in Rx memory **exceed** the trigger level of DMA In FIFO mode: trigger level reached In single-byte mode: receiver data available	In DMA mode: Write IIR or update the DMA_RxCurentRAddr such that the unread data are below of equal to trigger level In FIFO mode: the data in FIFO drops below the trigger level In single-byte mode: Reading the RBR
1	1	0	0	2nd	Character Timeout Indication (RDAI)	In DMA mode: There is at least 1 byte unread data in memory for a given time (by DMA_ Character_Timeout). In FIFO mode: No characters have been removed from or written to the Rx FIFO during the last 64 bits(bauds) times and there is at least 1 character in it during this time. In One-byte mode: N/A	In DMA mode: Update the DMA_RxCurentRAddr. In FIFO mode: Reading the RBR
0	0	1	0	3rd	In DMA T/R mode: TX DMA completed In other mode: Transmitter Holding Register Empty (TBEI)	In DMA T/R mode: TX DMA completed. All data is moved in Tx fifo. In other mode: Transmitter Holding Register Empty	In DMA T/R mode: Write IIR or restart TX DMA(write DMA_TxLength) In other mode Write IIR or writing into the Transmitter Holding Register
0	0	0	0	4th	MODEM Status (DSSI)	CTS changed	Reading the MODEM Status Register

Table 5-3 The interrupt table of UART

5.3 SFR table

Transmit Holding Register / Receive Buffer Register (THR/RBR 0x210000)

Name	Bit	Write/Read	Reset value	Description
THR/RBR	[7:0]	R/W	0	Used in single-byte mode or FIFO mode only. W: Write data to be sent to FIFO (fifo has depth 8 in FIFO mode and depth 1 in single-byte mode) R: Read received data from FIFO (FIFO has depth 8 in FIFO mode and depth 1 in single-byte mode)

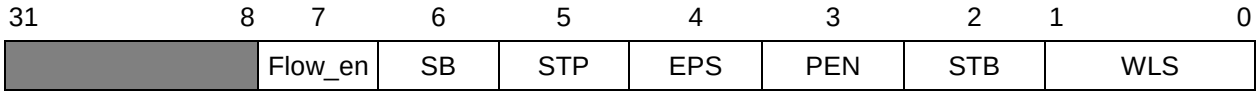
Divisor Latch Register (DLX 0x210004)

The DLX is used to control the baudrate of uart. The baudrate is fclk / (DLI + DLF/4) / 4.

Name	Bit	Write/Read	Reset value	Description
DLF	[1:0]	R/W	0	The fractional part of divisor
DLI	[31:16]	R/W	1	The integral part of divisor

Line Control Register (LCR - 0x210008)

The LCR determines the format of character.



Name	Bit	Write/Read	Reset value	Description
WLS	[1:0]	R/W	0	The data length of character 2'b00: 5 bits 2'b01: 6 bits 2'b10: 7 bits 2'b11: 8 bits
STB	[2]	R/W	0	The number of stop bits 1'b0: 1 bit 1'b1: 2 bits (1.5 bits when WLS is 2'b00)
PEN	[3]	R/W	0	The parity enable. 1'b0: no parity bit

				1’b1: with parity bit
EPS	[4]	R/W	0	The EPS and STP determines the parity bit. {STP,EPS} = 2’b00: odd parity 2’b01: even parity 2’b10: parity 0 2’b11: parity 2
STP	[5]	R/W	0	
SB	[6]	R/W	0	Set Break. When 1’b1, the uart_tx is 1’b0 until the SB is 1’b0.
Flow_en	[7]	R/W	0	Flow Ctrl Enable. Used in FIFO mode or dma mode only. When 1’b1, hardware can controls the uart_ncts and uart_nrts.

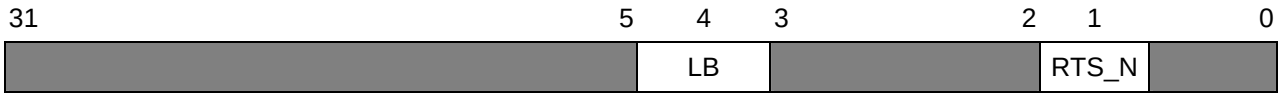
Interrupt Enable Register (IER - 0x21000C)

Interrupt enable.

31 ... 5	3	2	1	0
	EDSSI	ELSI	ETBEI	ERBFI

Name	Bit	Write/Read	Reset value	Description
ERDAI	[0]	R/W	0	Interrupt enable of RDAI
ETBEI	[1]	R/W	0	Interrupt enable of TBEI
ELSI	[2]	R/W	0	Interrupt enable of LSI
EDSSI	[3]	R/W	0	Interrupt enable of DSSI

Modem Control Register (MCR - 0x210010)



Name	Bit	Write/Read	Reset value	Description
RTS_N	[1]	R/W	0	Used for FIFO mode only. This bit is valid when Flow_en is 1’b0. The RTS_N controls the pin uart_nrts.
LB	[4]	R/W	0	Loop Back. The uart_rx is connected to uart_tx internally.

FIFO Control Register (FCR - 0x210014)

The FCR controls the FIFO and DMA enable.

Name	Bit	Write/Read	Reset value	Description
fifo_en	[0]	R/W	1	FIFO enable. Set 1’b1 to enter FIFO mode.
rxf_rst	[1]	Write 1 only & Auto clear	0	Rx FIFO reset. It can only be set 1’b1 and the rx FIFO is reset. After reset is done, it's automatically cleared. It's automatically set 1’b1 when uart_enable is set, as well. Software need to check it's 1'b0 before normal use.

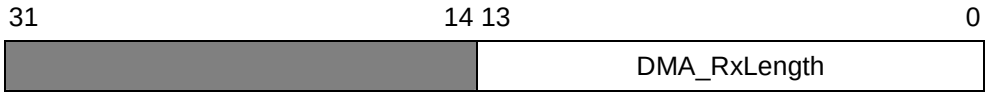
txf_rst	[2]	Write 1 only & Auto clear	0	Tx FIFO reset. It can only be set 1'b1 and the tx FIFO is reset. After reset is done, it's automatically cleared. It's automatically set 1'b1 when uart_enable is set, as well. Software need to check it's 1'b0 before normal use.
dma_en	[3]	R/W	1	DMA enable. Set 1'b1 to enter DMA mode. Note that when software writes dma_en by 1'b1, the fifo_en is also written by 1'b1.
RTL	[7:6]	R/W	0	Rx FIFO Trigger level. Only used in FIFO mode. The interrupt RDAI asserts when the number of data in rx fifo reaches the corresponding value: 2'b00: 1 byte 2'b01: 2 bytes 2'b10: 4 bytes 2'b11: 6 bytes

DMA_RxAddr (0x210018)

31 20	19 0
rsv0	DMA_RxAddr

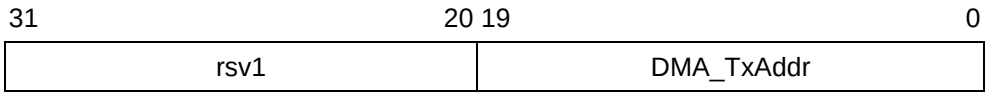
Name	**Bit**	**Write/Read**	**Reset value**	**Description**
DMA_RxAddr	[19:0]	R/W	0	DMA Rx Address of memory. Write DMA_RxAddr will reset RX DMA. **DO NOT** access the bits of DMA_RxAddr separately.
rsv0	[31:20]	R/W	0	Should be 0.

DMA_RxLength (0x21001C)



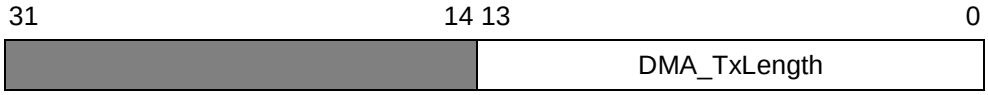
Name	Bit	Write/Read	Reset value	Description
DMA_RxLength	[13:0]	R/W	0	UART DMA Rx Length of memory. Write DMA_RxLength will reset RX DMA. **DO NOT** access the bits of DMA_RxLength separately.

DMA_TxAddr (0x210020)



Name	Bit	Write/Read	Reset value	Description
DMA_TxAddr	[19:0]	R/W	0	DMA Tx Address of memory. **DO NOT** access the bits of DMA_TxAddr separately.
rsv1	[31:20]	R/W	0	Should be 0.

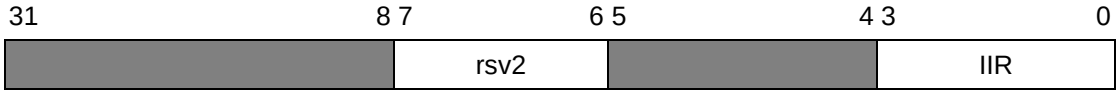
DMA_TxLength (0x210024)



Name	Bit	Write/Read	Reset value	Description
DMA_TxLength	[13:0]	R/W	0	UART DMA Tx Length of memory. Write DMA_TxLength will reset and start TX DMA. **DO NOT** access the bits of DMA_TxLength separately.

Interrupt Identify Register (IIR - 0x210028)

The IIR reflects the current interrupt status.



Name	Bit	Write/Read	Reset value	Description
IIR	[3:0]	see Description	4'h1	The value indicates different interrupt status. 4'h1: no interrupt 4'h6: LSI 4'h4: RDAI (data ready) 4'hc: RDAI (data timeout) 4'h0: DSSI Write any value to 0x210028 to trigger the "read IIR" event.
rsv2	[7:6]	N/A	N/A	The value is not defined.

Line Status Register (LSR - 0x21002C)

The LSR reflects the status of tx/rx.

31 8	7	6	5	4	3	2	1	0
	FERR	TEMT	THRE	BI	FE	PE	OE	DR

Name	Bit	Write/Read	Reset value	Description
DR	[0]	RO	0	The DR is 1'b1 whenever 1. single-byte mode: there is data in RBR 2. FIFO mode or DMA mode: there is any data in FIFO. When all data are read by software, DR is cleared automatically.
OE	[1]	W1C	0	The OE is 1'b1 in the following case: 1. single-byte mode: there is data in RBR and one more character is received. 2. FIFO mode or DMA mode: the FIFO is full and one more character is received. Write 1'b1 to clear it.
PE	[2]	W1C	0	The PE is 1'b1 in the following case: 1. single-byte mode: the data in RBR has wrong parity. 2. fifo mode or dma mode: the particular data in fifo with wrong parity is going to be read (by software or DMA). Write 1'b1 to clear it.
FE	[3]	W1C	0	The FE is 1'b1 in the following case: 1. single-byte mode: the data in RBR doesn't have a valid stop bit.

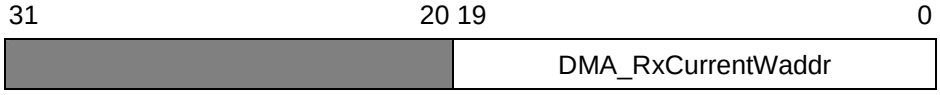
				2. fifo mode or dma mode: the particular data in fifo which doesn't have a valid stop bit is going to be read (by software or DMA). Write 1'b1 to clear it.
BI	[4]	W1C	0	The BI is 1'b1 in the following case: 1. single-byte mode: the whole character in RBR is logic 0. 2. fifo mode or dma mode: the particular character with whole character being logic 0 in fifo is going to be read (by software or DMA). Only one character is written into fifo when break occurs. Write 1'b1 to clear it.
THRE	[5]	RO	1	The THRE is 1'b1 in the following case: 1. single-byte mode: the THR is ready for being written new character. 2. fifo mode: the fifo is empty. 3. dma mode: all data in memory of DMA Tx are written to fifo. When new data is written, THRE is cleared automatically.
TEMT	[6]	RO	1	The TEMP is 1'b1 when Tx is not transmitting and THR(in single-byte mode) / Tx Fifo(other modes) is empty.
FERR	[7]	RO	0	The FERR is 1'b1 when there is any error occurred in fifo (OE,PE,FE or BI).

Modem Status Register (MSR - 0x210030)

The MSR reflects the status related to Modem function.

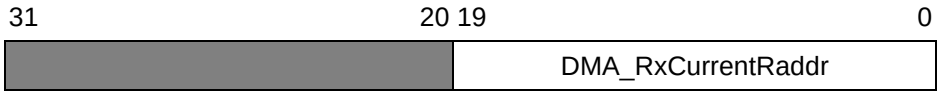
Name	Bit	Write/Read	Reset value	Description
DCTS	[0]	W1C	0	The DCTS is 1'b1 when the uart_cts_n changes. Write 1'b1 to clear it.
CTS_N	[4]	RO	N/A	The CTS_N reflects the value of pin uart_cts_n.

DMA_RxCurrentWAddr (0x210034)



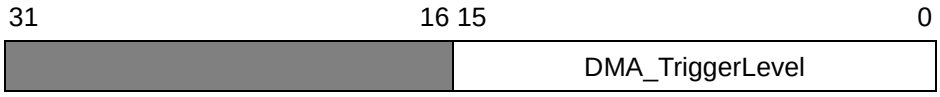
Name	Bit	Write/Read	Reset value	Description
DMA_RxCurrentWaddr	[19:0]	RO	0	DMA_RxCurentWAddr is the current written position of RX DMA pointer by hardware. The initial value is DMA_RxAddr, and will increase by 1 whenever one byte is transferred into the memory throuth RX DMA. It will be wrapped to DMA_RxAddr when DMA_RxLength bytes are received. **DO NOT** access the bits of DMA_RxCurrentWaddr separately.

DMA_RxCurrentWAddr (0x210038)



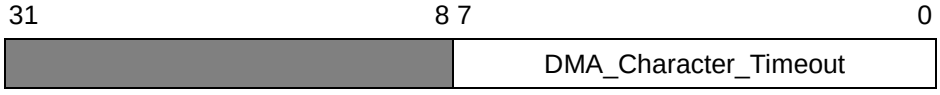
Name	Bit	Write/Read	Reset value	Description
DMA_RxCurrentRaddr	[19:0]	R/W	0	UART_DMA_RxCurentRAddr is the current read position by software. Software has the responsibility to update this pointer to maintain the correct operations of the UART Rx DMA. **DO NOT** access the bits of DMA_RxCurrentRaddr separately.

DMA_TriggerLevel (0x21003C)



Name	Bit	Write/Read	Reset value	Description
DMA_TriggerLevel	[15:0]	R/W	16'h0400	When RX data in memory exceeds the trigger level, UART will issue the interrupt RDAI. The value should be less than 16'h3FFF, or the RDAI will not assert.

DMA_Character_Timeout (0x210040)



Name	Bit	Write/Read	Reset value	Description
DMA_ Character_Timeout	[7:0]	R/W	0	When there is data in memory and no access of data (hardware doesn't receive new data and software doesn't read the old data), the

				timer counts. If the value of the timer reaches DMA_Character_Timeout * 16 bits(bauds), UART will assert interrupt RDAI.

Uart_controls (0x210044)

The Uart_controls controls the uart global settings.

Name	Bit	Write/Read	Reset value	Description
uart_enable	[0]	R/W	0	UART enable. 1'b1: Enable UART 1'b0: Disable UART

DMA_status (0x21004C)

The DMA_status reflects some status of DMA.

Name	Bit	Write/Read	Reset value	Description
dma_tx_busy	[0]	RO	0	Dma Tx busy. To indicate whether tx dma is working 1'b1 : some data in memory block haven't been moved into fifo 1'b0 : all DMA_TxLength data in memory block are moved into fifo

6 Inter-Integrated Circuit (I2C)

6.1 Overview

The I2C module provides the interface between the device and I2C-compatible devices connected by the two-wire I2C serial bus.

Features:

- Role - Master only
- Bit Rate - 100k, 400k and 800k
- Slave Address – 7 bits only
- Support standard and smart slave devices
- Support arbitration (contention) with other masters

6.2 Bus Connection

I2C data is communicated by using the serial data (SDA) pin and the serial clock (SCL) pin. Both SDA and SCL are bidirectional and must be connected to a positive supply voltage by using a pull-high resistor as shown in 1.

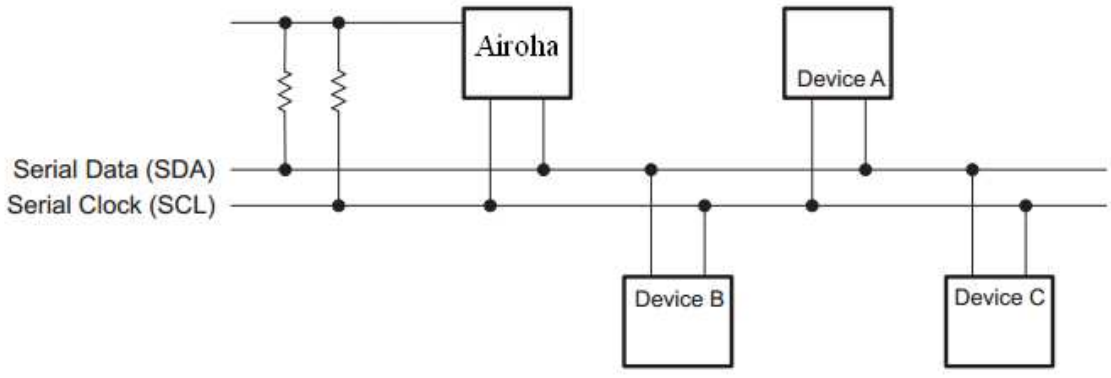


Figure 6-1 The connection of I2C bus

6.3 I2C specification

The bus transmission format is shown in 2. The transmission begin with a START. Then the address/data bytes (8 bits, MSB fist) are followed. The STOP is at the end. If the transmission is not going to finish, the STOP can be replaced by START and the STOP is at the very end of the transmission. In this case, it is called combined format.

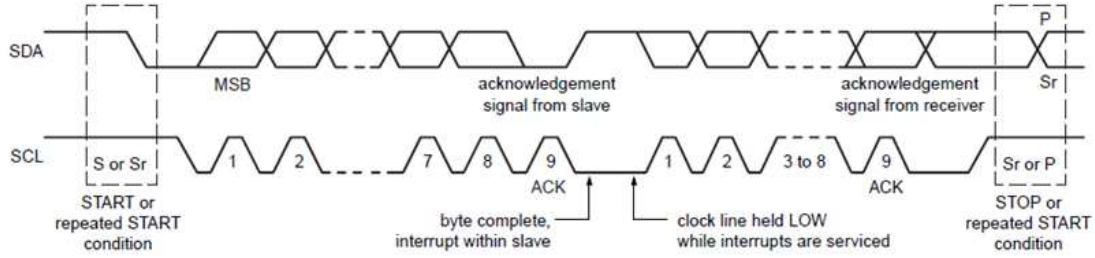


Figure 6-2 The format of I2C bus transmission

The conditions of START and STOP are shown in 3.

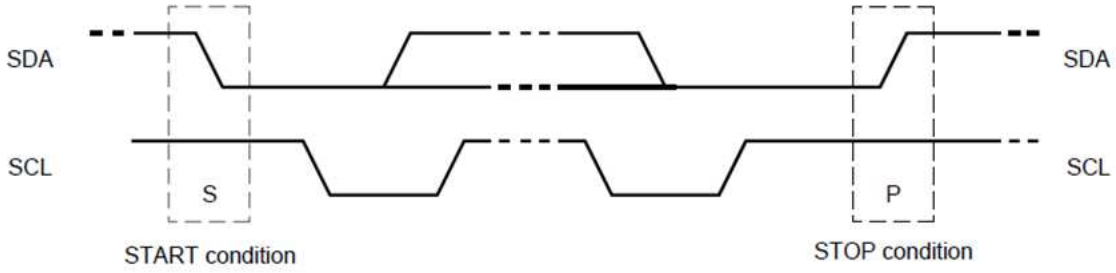


Figure 6-3 The START and STOP condition

The data transition is allowed when SCL is low, shown in 4.

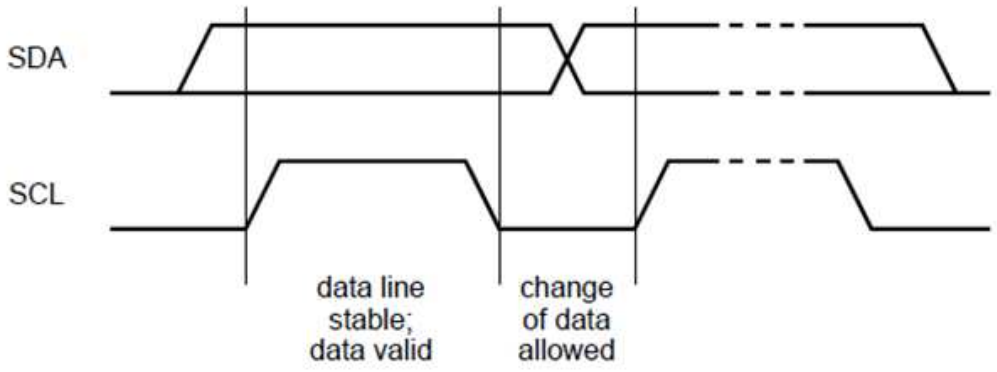


Figure 6-4 The valid timing of data change

The arbitration (contention) is supported. When the I2C loses the arbitraion, the contention_fail status asserts. 5 shows the arbitration. The DATA1 belongs SDA of Master 1 and DATA2 Master 2. The Master 1 and Master 2 generate the START and after that, the DATA1 return to 1 before DATA2, so Master 1 loses the arbitration.

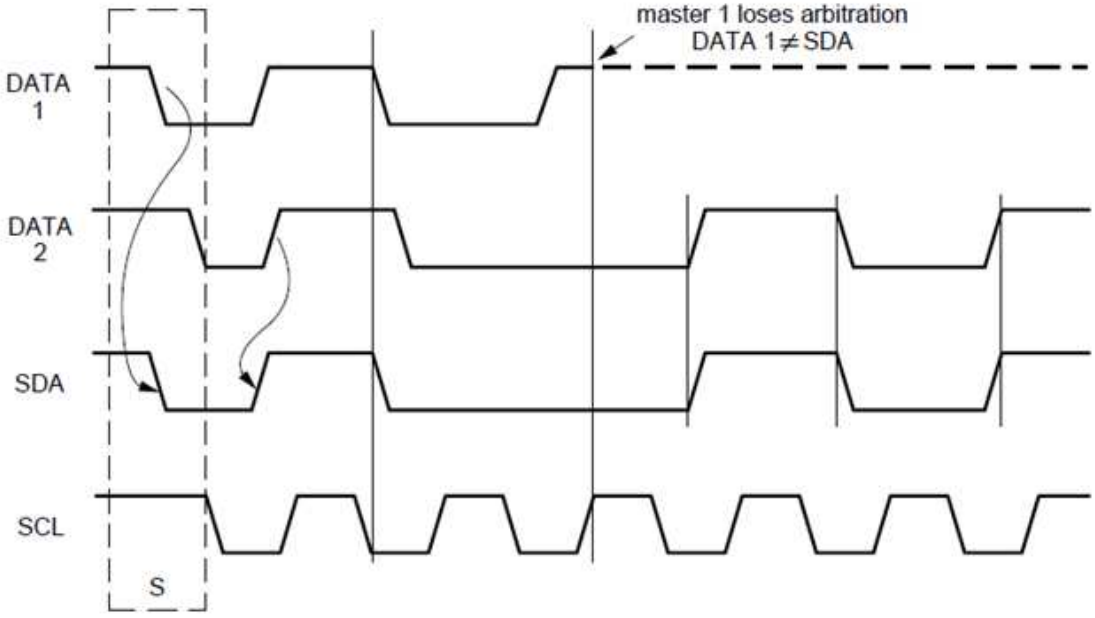


Figure 6-5 The contention of I2C bus

6.4 Function Descriptions

6.4.1 Devices and Operations

There are two kinds of slave devices, standard and smart ones. For standard devices, the length of the

address of registers is 1 byte and the smart devices is 2 bytes. In additon, there are three different commands in I2C spec. The six scenarios are shown as below

(a) Standard device

a.1 Write

START	Control Byte 1 (write)	ACK	Address Byte	ACK	Data Bytes (byte + ack + byte + ack + … + byte +ack)	STOP

a.2 Random Read

START	Control Byte 1 (write)	ACK	Address Byte	ACK	START	Control Byte 2 (read)	ACK	Data Bytes (byte + ack + byte + ack + … + byte + no ack)	STOP

a.3 Current Read

START	Control Byte 2 (read)	ACK	Data Bytes (byte + ack + byte + ack + … + byte + no ack)	STOP

(b) Smart device

b.1 Write

START	Control Byte 1 (write)	ACK	Address High Byte	ACK	Address Low Byte	ACK	Data Bytes (byte + ack + byte + ack + … + byte +ack)	STOP

b.2 Random Read

START	Control Byte 1 (write)	ACK	Address High Byte	ACK	Address Low Byte	ACK	START	Control Byte 2 (read)	ACK	Data Bytes (byte + ack + byte + ack + … + byte + no ack)	STOP

b.3 Current Read

START	Control Byte 2 (read)	ACK	Data Bytes (byte + ack + byte + ack + … + byte + no ack)	STOP

6.4.2 Memory Blocks

Before starting a Write or Read operation, software need to prepare a block of memory which include the information for I2C module. According to different commands and devices, the formats are shown as below. After the memory block is ready, software then set I2C_EN as 1'b1 to start the I2C operation.

(a) Memory format for Standard device with Write or Random Read command

Settings	Length High Byte	Length Low Byte	Control Byte 1	Address Low Byte	Write: Data Bytes or Read: Control Byte 2

(b) Memory format for Smart device with Write or Random Read command

Settings	Length High Byte	Length Low Byte	Control Byte 1	Address High Byte	Address Low Byte	Write: Data Bytes or Read: Control Byte 2

(c) Memory format for both devices with Current Read command

Settings	Length	Length	Control Byte 2

	High Byte	Low Byte	

6.4.3 Field Explanation

The meaning of each field in previous section is explained below:

Control Byte 1
8 bits. Bit [7:1] is the slave address and bit[0] is 1'b0 (for writing)

Control Byte 2
8 bits. Bit [7:1] is the slave address and bit[0] is 1'b1 (for reading)

Address Byte
8 bits. The desired address of register of (standard) slave device

Address High Byte	Address Low Byte
8+8=16 bits. The desired address of register of (smart) slave device	

Data Bytes
Series of 8 bits. The written/read data between I2C module and slave device

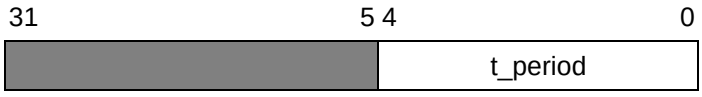
Length High Byte	Length Low Byte
8+8=16 bits. For Write: Length of data bytes For Read: Length of data bytes (not including the Control Byte 2)	

Settings
Bit 0: STD_SMT_SWITCH. 0 for standard I2C, and 1 for smart I2C Bit 1: RW_SW. 0 for Write, and 1 for Read. Bit 2: CURR_RD_EN. 0 for random read, and 1 for current read. Bit 3: SCL_SYN_EN. Enable of sync of SCL. Need to set 1'b0 in 800k mode Bit [5:4] SCL_CHK_SEL. SCL sync check point. Adjust the timing for SCL sync function, set 2'b10 as default value. Note that the value 2'b11 is reserved.

Bit [6] SPIKE_DIS. 1’b1: Disable digital spike suppression. 1’b0: Enable.

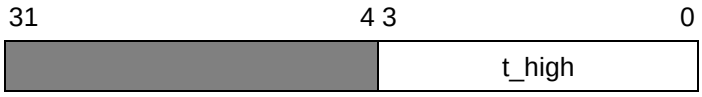
6.5 SFR Tables

t_period (0x214000)



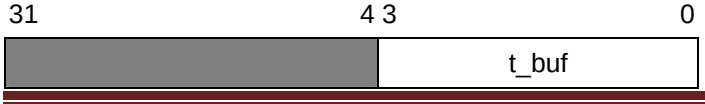
Name	Bit	Write/Read	Reset value	Description
t_period	[4:0]	R/W	0	Period of 1 bit. The unit is fclk/(divisor+1)

t_high (0x214004)



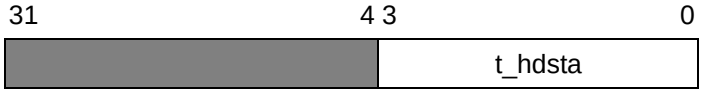
Name	Bit	Write/Read	Reset value	Description
t_high	[3:0]	R/W	0	SCL is high during one period. The worse effective time is t_high - t_syncheck + t_busdelay. The unit is fclk/(divisor+1)

t_buf (0x214008)



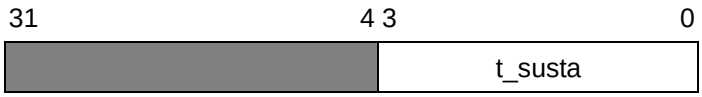
Name	Bit	Write/Read	Reset value	Description
t_buf	[3:0]	R/W	0	Bus free time between stop and start. The unit is fclk/(divisor+1)

t_hdsta (0x21400C)



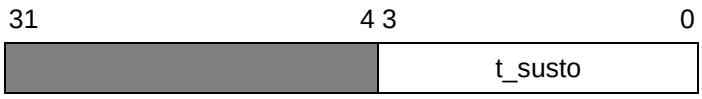
Name	Bit	Write/Read	Reset value	Description
t_hdsta	[3:0]	R/W	0	Hold time for start. The unit is fclk/(divisor+1)

t_susta (0x214010)



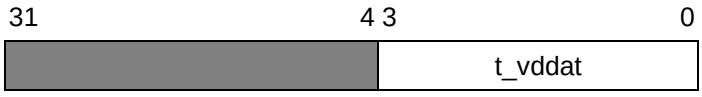
Name	Bit	Write/Read	Reset value	Description
t_susta	[3:0]	R/W	0	Set-up time for start. The unit is fclk/(divisor+1)

t_susto (0x214014)



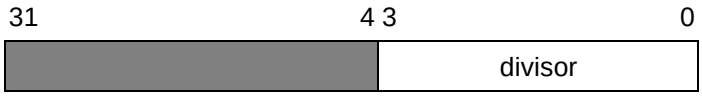
Name	Bit	Write/Read	Reset value	Description
t_susto	[3:0]	R/W	0	Set-up time for stop. The unit is fclk/(divisor+1)

t_vddat (0x214018)



Name	Bit	Write/Read	Reset value	Description
t_vddat	[3:0]	R/W	0	Data valid time after SCL turns to low. The unit is fclk/(divisor+1)

divisor (0x21401C)



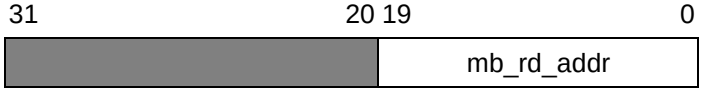
Name	Bit	Write/Read	Reset value	Description
divisor	[3:0]	R/W	0	Generate tick with freq = fclk/(divisor+1). Used for the above sfrs.

1 gives the recommended values of above SFRs in different scenarios.

Name	Recommended Value @ 48M clk 100k / 400k / 800k mode			Recommended Value @ 16M clk 100k / 400k / 800k mode		
t_period	30	24	15	20	20	20
t_high	15	11	9	10	9	10
t_buf	15	13	7	10	11	9
t_hdsta	13	8	9	9	8	10
t_susta	15	8	4	10	6	5
t_susto	13	8	8	9	8	8
t_vddat	2	2	2	2	2	2
divisor	15	4	3	7	1	0

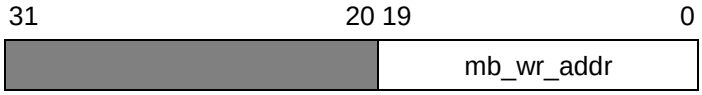
Table 6-1 The recommended value of parameters of I2C

mb_rd_addr (0x214020)



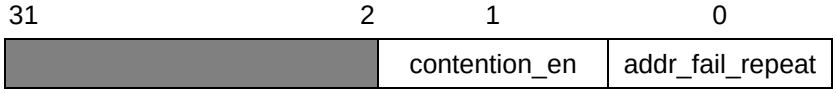
Name	Bit	Write/Read	Reset value	Description
mb_rd_addr	[19:0]	R/W	0	Memory block address for I2C to read the memory block of command, i.e., the address of "Settings" field in section 6.4.2.

mb_wr_addr (0x214024)



Name	Bit	Write/Read	Reset value	Description
mb_wr_addr	[19:0]	R/W	0	Memory block address for I2C to write data received from slave.

i2c_ctrl (0x214028)



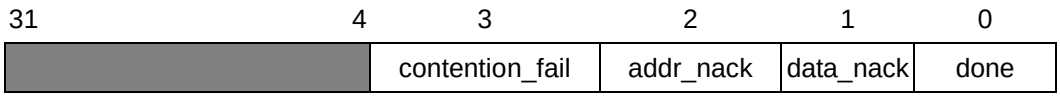
Name	Bit	Write/Read	Reset value	Description
contention_en	[0]	R/W	0	Contention enabled. Set 1'b1 to detect contention from the bus.
addr_fail_repeat	[1]	R/W	0	Set 1'b1 to auto repeat the command when there is no ack received from slave for address bits.

i2c_en (0x21402C)

Name	Bit	Write/Read	Reset value	Description
i2c_en	[0]	R/W	0	I2C enable. Set 1'b1 to start I2C. Set 1'b0 when the transmission is done (after reading the status).

i2c_status (0x214030)

Indication of I2C transmission status.



Name	Bit	Write/Read	Reset value	Description
done	[0]	W1C	0	I2C is done.
data_nack	[1]	W1C	0	The ack of data bits is not received.
addr_nack	[2]	W1C	0	The ack of address bits is not received.
contention_fail	[3]	W1C	0	There is contention on the bus.

i2c_busy (0x214034)

Name	Bit	Write/Read	Reset value	Description
i2c_busy	[0]	RO	0	Indicates I2C is working or idle, can be monitored anytime. 1'b1: working 1'b0: idle

i2c_int (0x214038)

Name	Bit	Write/Read	Reset value	Description
i2c_int	[0]	W1C	0	I2C interrupt status. It's set to 1'b1 when I2C transfer is finished.

i2c_int_mask (0x21403C)

Name	Bit	Write/Read	Reset value	Description
i2c_int_mask	[0]	R/W	0	Enable of i2c_int. If it's 1'b1, the i2c_int can assert interrupt to mcu.

7 Serial Peripheral Interface (SPI)

7.1 Overview

SPI is a serial-parallel interface. The SPI can be either configured as a master or a slave. In the master configuration, data transferring can be throughDMA, DMA FIFO, Register mode, and 3-wire,4-wire I/O interfaces are provided to meet different requirements. In the slave configuration, data transferring can be through DMA FIFO, Register mode, and 4-wire I/O interface is provided.

To achieve high-throughput communications, TX and RX are implemented with internal 8-byte FIFO, respectively. The SPI block diagram is shown in

In this section, TX means the SPI transmits data; RX means the SPI receives data; a transaction means a completion of a SPI TX/RX session.

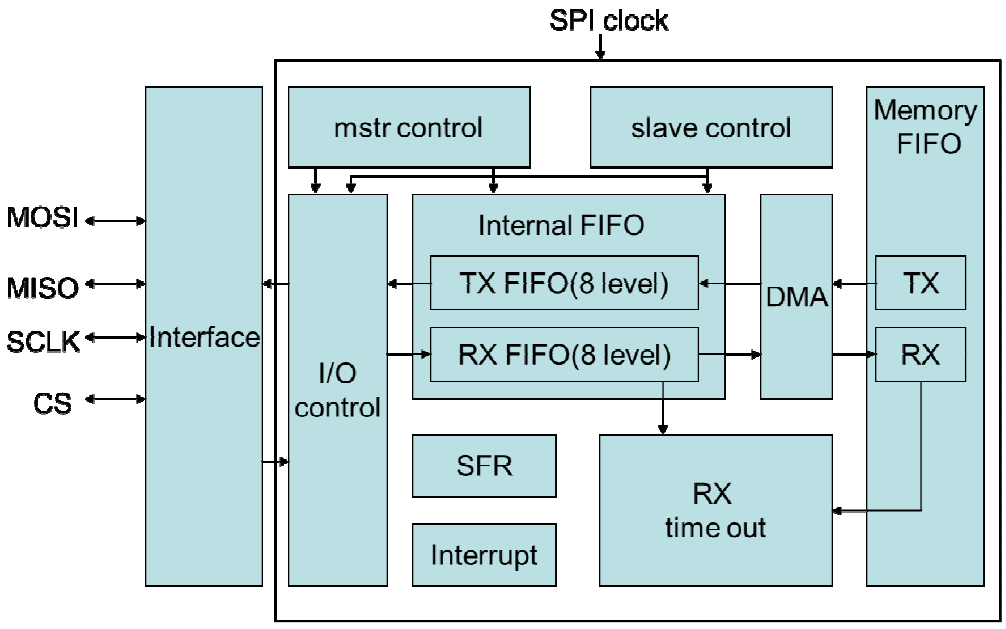


Figure 7-1 SPI Block Diagram

The SPI module's clock is the clock used in the SPI module, and the frequency is 16MHz/48Mhz.

The SPI can be configured as a master or a slave. The pin topology is illustrated in section 7.2.

7.2 Pin Topology

In the master configuration, 3-wire,4-wire I/O interfaces are provided, which are illustrated in and . In the slave configuration, only 4-wire I/O interface is provided, which is illustrated in .

I/O Name	Direction	Description
CS	O	Chip selection to slave. Can be active high or active low.
SCLK	O	Serial clock to slave. Maximum frequency is 8M/24MHz(0.5*SPI module's clock).
MOSI	I/O	Master's data input/output.
MISO	I	No use in this configuration.

Table 7-1 3-wire mode in the master configuration

I/O Name	Direction	Description
CS	O	Chip selection to slave. Can be active high or active low.
SCLK	O	Serial clock to slave. Maximum frequency is 8M/24MHz(0.5*SPI module's clock).
MOSI	O	Master's Data input.
MISO	I	Mater's Data output. Data is received by the order of LSByte first, MSBit first in each data.

Table 7-2 4-wire mode in the master configuration

I/O Name	Direction	Description
CS	I	Chip selection from master. Can be active high or active low.
SCLK	I	Serial clock from master. Maximum frequency is 16Mz/24MHz(SPI module's clock)
MOSI	I	Slave's Data input.
MISO	O	Slave's Data input. Data is transmitted in the order of LSByte first, MSBit first in each byte. Note:This pin is always output no matter slave transmits data.

Table 7-3 4-wire mode in the slave configuration

7.3 Data Transferring Mode

There are 3 data transferring modes in SPI, which are DMA mode, DMA FIFO mode, and Register mode. DMA(Direct Memory Access) mode and DMA FIFO mode make use of a dedicated DMA in SPI, and hence DMA related parameters(src_addr, dst_addr, fifo_size, ptr) are needed to be set. Register mode makes use of the internal FIFO, and hence no DMA parameters need to be set. Master supports all 3 modes, but slave only supports DMA FIFO mode and Register modes. The operation of different data transferring mode is illustrated in Table 7-4 , Table 7-4 , .

7.3.1 Memory FIFO

To use the DMA in DMA/DMA FIFO mode, host should maintain the pointer of the memory FIFO. The memory FIFO is a ring FIFO consisting of base address(tx_src_addr/rx_dst_addr), offset pointer(tx_wptr,tx_rptr/rx_wptr,rx_rptr), and size(tx_fifo_size/rx_fifo_size). These parameters are under some constraints depending on the size of data word(see dw_width in SFR SPI_CONFIG). The base address/offset pointer/size has to be multiple of unit, where unit=1 if a data word is 2 bit~8 bit, unit=2 if a data word is 9 bit~16 bit, unit=4 if a data word is 17 bit~32 bit. The offset pointer starts from 0 and wrap to 0 at “size-unit”(i.e. the maximum value is “size-unit”). The memory FIFO is empty when wptr=rptr. The memory FIFO is full when wptr=rptr-unit if wptr<rptr and wptr=rptr+size-unit if wptr>rptr where there is “(size-unit)/unit” data word in the memory FIFO at that time.

In TX, tx_wptr has to be updated once there is data written to the TX memory FIFO by host, and DMA will automatically update the tx_rptr when data is transferred to the TX internal FIFO from the TX memory FIFO (see). For example, if host writes 4 bytes into the TX memory FIFO, it should also increase tx_wptr by 4; the tx_rptr is updated by the DMA as soon as data is transferred to the TX internal FIFO from the TX memory FIFO. Writing tx_src_addr in SFR SPI_TX_SRC_ADDR resets both tx_wptr and tx_rptr to 0.

In RX, rx_rptr has to be updated once there is data read from the RX memory FIFO by host, and DMA will automatically update the rx_wptr when data is received and transferred from the RX internal FIFO to the RX memory FIFO (see). For example, if host reads 4 byte from the RX memory FIFO, then it should also increase rx_rptr by 4; the rx_wptr is updated by the DMA as soon as data is transferred from the RX internal FIFO to the RX memory FIFO. Writing rx_dst_addr in SFR SPI_RX_DST_ADDR resets both rx_wptr and rx_rptr to 0.

7.3.2 Internal FIFO

The internal FIFO size is 8 byte and can be manipulated by writing SFR SPI_TX_DATA or reading SFR SPI_RX_DATA in Register mode. Host does not have to maintain offset pointer in Register mode.

The internal FIFO has some constraints depending on the unit, where unit=1 if a data word is 2 bit~8 bit, unit=2 if a data word is 9 bit~16 bit, unit=4 if a data word is 17 bit~32 bit. First constraint is it contains at most 8 data word if uint=1, 4 data word if unit=2, 2 data word if unit=4. Second constraint is each time when writing/reading SPI_TX_DATA, SPI_RX_DATA, host will write/read unit byte into/from the internal FIFO.

7.3.3 DMA Mode

In DMA mode, TX/RX data number is determined by tbc/rbc. A transaction starts when writing 1 to the start bit in SPI_CTRL SFR, and the transaction ends when both tbc/rbc number of data is transmitted/received.

Mode	3-wire,fdm=0	3-wire,fdm=1	4-wire,fdm=0	4-wire=1,fdm=1
Set src/dst address	O	invalid	O	O
Set FIFO size	O	invalid	O	O
Maintain tx_wptr/rx_rptr	O	invalid	O	O
Set tbc/rbc	O	invalid	O	O
Start condition	Write 1 to “start” bit	Invalid	Write 1 to “start” bit	Write 1 to “start” bit

End condition	TX cnt="tbc" RX cnt="rbc"	invalid	TX cnt="tbc" RX cnt="rbc"	TX cnt="tbc" RX cnt="rbc"
TX/RX	TX/RX/ TX then RX	invalid	TX/RX/ TX then RX	TX and RX simultaneously
TX data length	tbc	invalid	tbc	tbc
RX data length	rbc	invalid	rbc	rbc

Table 7-4 DMA mode(only for master)

7.3.4 DMA FIFO Mode

In DMA FIFO mode, TX/RX data number is the data number written to the TX memory FIFO. A transaction starts as soon as the TX memory FIFO is not empty, and the transaction ends when both TX memory FIFO and internal FIFO are empty at last SCLK edge.

Mode	3-wire, fifo_rx_en=0	3-wire, fifo_rx_en=1	4-wire, fifo_rx_en=0	4-wire, fifo_rx_en=1
Set src/dst address	O	O	O	O
Set FIFO size	O	O	O	O
Maintain tx_wptr/rx_rptr	O	O	O	O

Set tbc/rbc	X	X	X	X
Start condition (master only)	TX memory FIFO is not empty.	TX memory FIFO is not empty.	TX memory FIFO is not empty.	TX memory FIFO is not empty.
End condition (master only)	TX memory FIFO and internal FIFO are empty.	TX memory FIFO and internal FIFO are empty.	TX memory FIFO and internal FIFO are empty.	TX memory FIFO and internal FIFO are empty.
TX/RX	TX	RX	TX	TX and RX simultaneously
TX data length	Data number in TX FIFO	Data number in TX FIFO	Data number in TX FIFO	Data number in TX FIFO
RX data length	0	Data number in TX FIFO	0	Data number in TX FIFO

Table 7-4 DMA FIFO mode

7.3.5 Register Mode

In Register mode, TX/RX data number is the data number written to the TX internal FIFO. A transaction starts as soon as the TX internal FIFO is not empty, and the transaction ends when TX internal FIFO is empty at last SCLK edge.

Mode	3-wire, fifo_rx_en=0	3-wire, fifo_rx_en=1	4-wire, fifo_rx_en=0	4-wire, fifo_rx_en=1
Set src/dst address	O	O	O	O
Set FIFO size	O	O	O	O
Maintain tx_wptr/rx_rptr	O	O	O	O
Set tbc/rbc	X	X	X	X
Start condition (master only)	TX internal FIFO is not empty.	TX internal FIFO is not empty.	TX internal FIFO is not empty.	TX internal FIFO is not empty.
End condition (master only)	TX internal FIFO is empty.	TX internal FIFO is empty.	TX internal FIFO is empty.	TX internal FIFO is empty.
TX/RX	TX	RX	TX	TX and RX simultaneously
TX data length	Data number in TX FIFO	Data number in TX FIFO	Data number in TX FIFO	Data number in TX FIFO
RX data length	0	Data number in TX FIFO	0	Data number in TX FIFO

Table 7-5 Register mode

7.4 Data Transferring Format

Bit width of a data word ranges from 2 bit to 32 bit. In DMA/DMA FIFO mode, all parameters related to the memory FIFO need to be 1/2/4 byte aligned when data bit number is 2bit ~8bit, 9 bit ~16bit, 17bit ~32bit, respectively.

When MSB in SFR SPI_CONFIG is set to 1, MSB of each data word is sent first. When MSB in SFR SPI_CONFIG is set to 0, LSB of each data word is sent first.

In the master configuration, the SPI can limit the data number in one cs-active session by setting cs_data in SFR SPI_CTRL to value other than 0. If cs_data!=0, the SPI divides the transaction to several cs-active sessions with each session contains “cs_data” data word. If cs_data=0, all the data in the transaction is in one cs-active session. This is shown in . In the slave configuration, it is not supported.

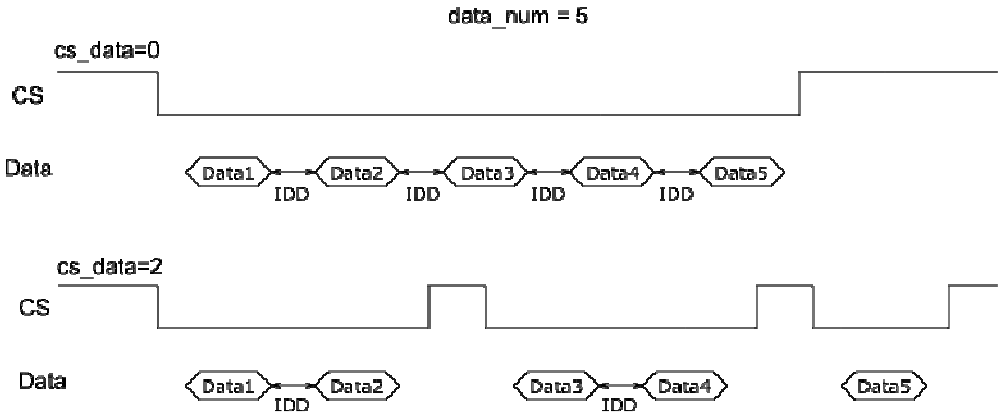


Figure 7-2 Data number in one cs and inter-byte delay

7.5 SPI Timing

In the master configution, an IDD(inter-data delay) exists in 2 consecutive data word. There are 3 types of IDDs, WR_WR, RD_RD, WR_RD, which can be set individually. The IDD chosen during a transaction is determined by the rule listed in Table 7-6 6

Data mode	Configuration	This data	Next data	Delay name
DMA FIFO/Register	3-wire,fifo_rx_en=0	Don’t care	Don’t care	WR_WR
DMA FIFO/Register	3-wire,fifo_rx_en=1	Don’t care	Don’t care	RD_RD
DMA FIFO/Register	4-wire,fifo_rx_en=0	Don’t care	Don’t care	WR_WR
DMA FIFO/Register	4-wire,fifo_rx_en=1	Don’t care	Don’t care	WR_WR

DMA	3-wire,fdm=0	TX	TX	WR_WR
DMA	3-wire,fdm=0	RX	RX	RD_RD
DMA	3-wire,fdm=0	TX	RX	WR_RD
DMA	4-wire,fdm=0	TX	TX	WR_WR
DMA	4-wire,fdm=0	RX	RX	RD_RD
DMA	4-wire,fdm=0	TX	RX	WR_RD
DMA	4-wire,fdm=1	Don't care	Don't care	WR_WR

Table 7-6 Rules to Determine Inter-data Delay in master

In the slave configuration, IDD depends on SPI module's clock. If SPI module's clock is 48MHz, then the IDD has to be greater than 90ns. If SPI module's clock is 16MHz, then the IDD has to be greater than 260ns.

CS to SCLK(CS2SCLK) delay is the delay between CS asserting(CS become active) and the first SCLK. SCLK to CS(SCLK2CS) delay is the delay between the last SCLK to CS deasserting(CS become inactive). The delay after CS deassert(CSAFT) is the delay between CS deasserting to next transaction starts. These timing relations are shown in Figure 7-3 and Figure 7-4 .

In the master configuration, CS2SCLK, SCLK2CS, CSAFT can be set separately. In the slave configuration, these delays depend on SPI module's clock. If SPI module's clock is 48MHz, then CS2SCLK, SCLK2CS has to be greater than 90ns and CSAFT has to be greater than 45ns. If SPI module's clock is 16MHz, then CS2SCLK, SCLK2CS has to be greater than 260ns and CSAFT has to be greater than 130ns.

7.6 Clock Polarity(pol) and Phase(pha)

The SPI supports all SCLK polarity and phase configuration.

7.6.1 Clock Polarity(pol)

Clock polarity determines the SCLK polarity. If pol=0 the SCLK remains low when it is not active. If pol=1 the SCLK remains high when it is not active.

7.6.2 Clock Phase(pha)

Clock phase determines the data sampling and shifting phase. If pha=0, data samples at odd edges of the SCLK and shifts at even edges. If pha=1, data samples at even edges of the SCLK and shifts at odd edges.

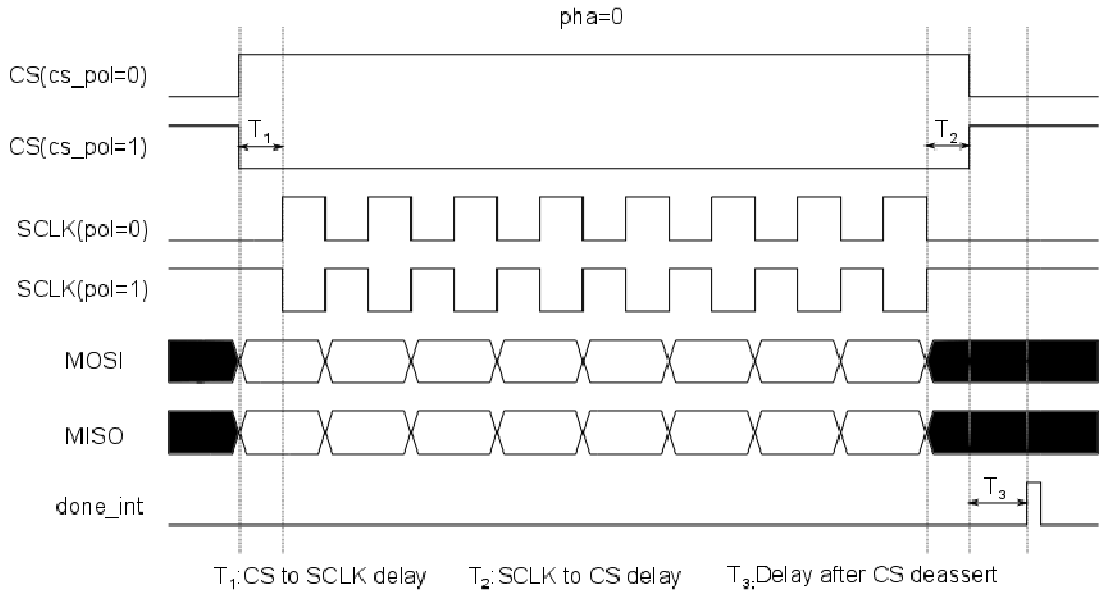


Figure 7-3 Timing Diagram of SPI when pha=0

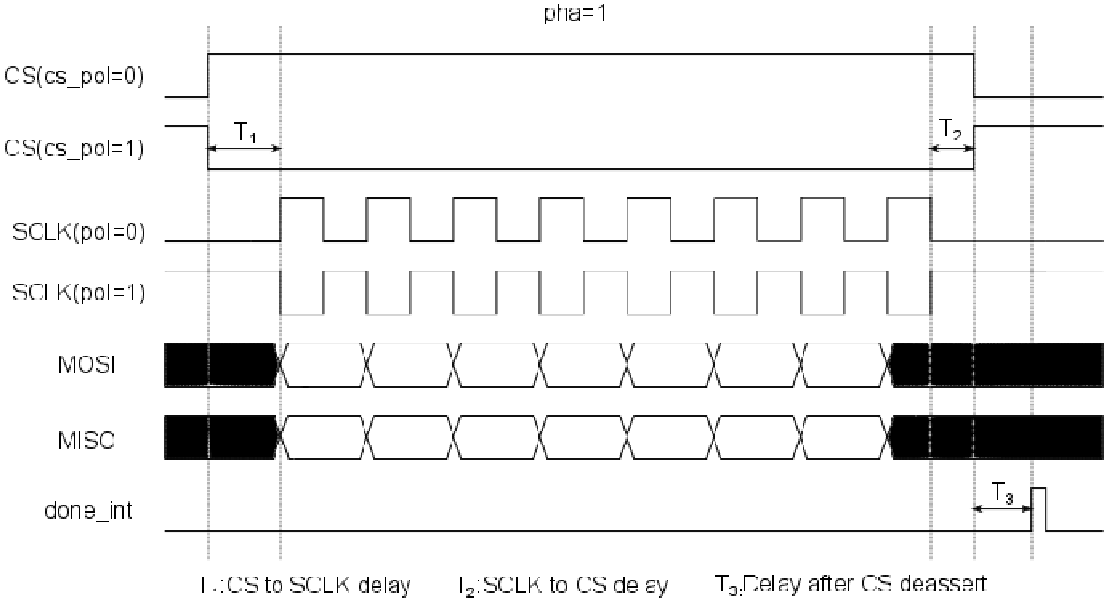


Figure 7-4 Timing Diagram of SPI when pha=1

7.7 Interrupt

Each SPI interrupt can be enabled/disabled individually and be cleared by writing 1 to the corresponding bit in the SFR SPI_INT_FLAG.

7.7.1 Done Pnterrupt(done_int)

A done interrupt indicates the SPI finishes a transaction.

In the master configuration, it indicates a transaction was done.

In the slave configuration, it indicates CS becomes inactive from active.

7.7.2 TX FIFO Threshold Interrupt(tx_th_rch_int)

Indicating the data count in TX FIFO is less or equal than the threshold. In DMA, DMA FIFO mode, TX FIFO is the memory FIFO. In Register mode, TX FIFO is the internal FIFO.

7.7.3 TX Empty Interrupt(tx_empty_int)

Indicating the TX FIFO is empty. In DMA, DMA FIFO mode, TX FIFO is the memory FIFO. In Register mode, TX FIFO is the internal FIFO.

7.7.4 TX Underflow Interrupt(tx_uf_int)

Only valid in the SPI slave. It Indicates that the SPI TX internal FIFO does not load successfully. The TX data to master is not valid data in this case.

7.7.5 RX FIFO Threshold Interrupt(rx_th_rch_int)

It Indicates the data count in RX FIFO is greater than the threshold. In DMA, DMA FIFO mode, RX FIFO is the memory FIFO. In Register mode, RX FIFO is the internal FIFO.

7.7.6 RX Full Interrupt(rx_full_int)

It indicates the RX FIFO is full. In DMA, DMA FIFO mode, RX FIFO is the memory FIFO. In Register mode, RX FIFO is the internal FIFO.

7.7.7 RX Overflow Interrupt(rx_of_int)

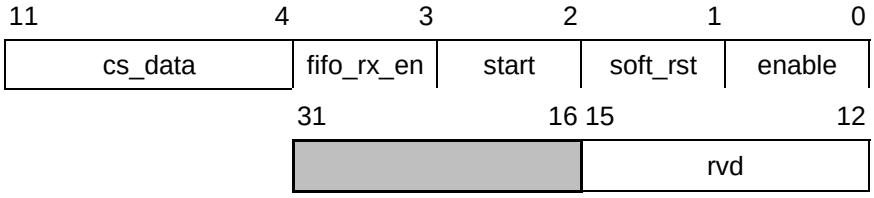
Only valid in the SPI slave, and indicates the SPI RX internal FIFO does not store successfully. The RX data is discarded in this case.

7.7.8 RX Time-out Interrupt(rx_to_int)

A time-out counter to notify there is data remains in the RX FIFO. The time-out counter starts to count when RX FIFO is not empty and reset when new RX data is received or RX FIFO is read. In DMA/DMA FIFO mode RX FIFO is read by updating RX read pointer. In Register mode RX FIFO is ready by read internal RX FIFO.

7.8 SFR Table

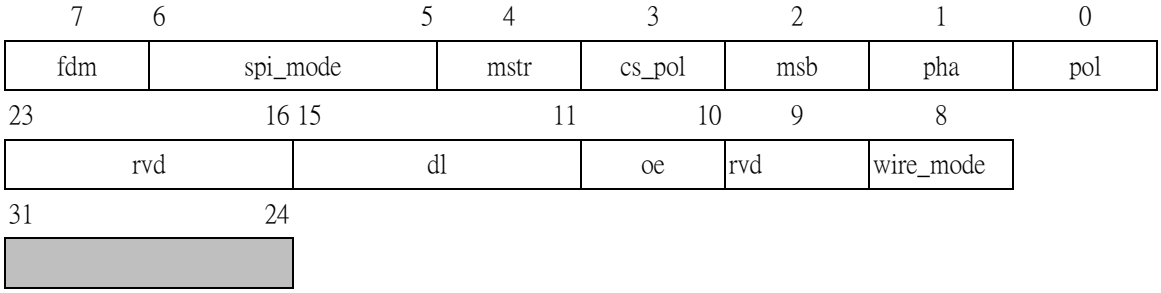
SPI_CTRL register: 0x218000



Name	Bit	Write/Read	Reset value	Description
enable	[0]	RW	1'h0	The enable of the SPI controller. 1'b0:Disable 1'b1:Enable
soft_rst	[1]	R/WC	1'h0	SPI soft reset. Write 1 to soft reset SPI controller and FIFO. This bit should be polled until it becomes 0(See note 1). 1'b0: soft reset is not busy. 1'b1: soft reset is busy. Note: 1. It takes 1 clock to let this bit become 1, be sure to add 1 software NOP before start to poll.

				2. Interrupt flag is not reset by this bit. If interrupt flags are desired to be cleared, writing 1 to interrupt flag registers SPI_INT_FLAG.
start	[2]	WC	1'h0	SPI transaction start trigger. Used in master's DMA mode only. Write 1 to this bit to trigger a SPI transaction. Clear to 0 by hardware automatically. Note: Enable has to be set before writing 1 to this bit.
fifo_rx_en	[3]	RW	1'h0	RX enable. Used in master's Register/DMA FIFO mode only. 1'b0:SPI receives data. 1'b1:SPI does not receive data.
cs_data	[11:4]	RW	8'h0	Data word number in one CS-active session. 8'h0: all data is in one CS-active session. others: "cs_data" data word in one CS-active session. Note: It is by the unit of data, not byte.
rvd	[15:12]	RW	4'h5	Reserved bit

SPI_CONFIG register: 0x218004

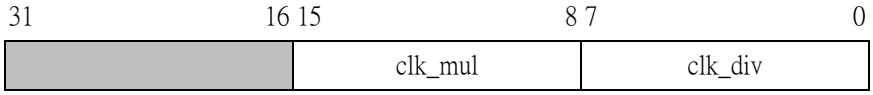


Name	Bit	Write/Read	Reset value	Description
pol	[0]	RW	1'h0	SCLK polarity selection. 1'b0 : Active-high. In idle state, SCLK is low. 1'b1 : Active-low. In idle state, SCLK is high.
pha	[1]	RW	1'h0	Data sampling edge of the SCLK. 1'b0:sampling occurs at ODD edges. 1'b1:sampling occurs at EVEN edges.

msb	[2]	RW	1'h1	MSB/LSB of a data word is transmitted/received first. 1'b0:LSB is transmitted/received first. 1'b1:MSB is transmitted/received first.
cs_pol	[3]	RW	1'h1	CS polarity selection. 1'b0:Active-high. In idle state, CS is low. 1'b1:Active-low. In idle state, CS is high.
mstr	[4]	RW	1'h1	Master/slave switch. 1'b0:Slave 1'b1:Master
spi_mode	[6:5]	RW	2'h0	DMA/Register/DMA FIFO mode selection. 2'b00 : Register mode 2'b01 : DMA mode(master only) 2'b10 : DMA FIFO mode 2'b11 : reserved
fdm	[7]	RW	1'h0	Full/Half duplex selection. For master's DMA mode only. 1'b1 : Full-duplex (not supported in 3-wire mode) 1'b0 : Half-duplex
wire_mode	[8]	RW	2'h1	Wire mode selection. Applied to master only. Slave is always 4-wire. 2'b00:3-wire mode 2'b01:4-wire mode
rvd	[9]	RW	1'h0	Reserved bit. This bit should be set to 0.
oe	[10]	RW	1'h1	Output enable for MOSI pin in 3-wire/3p1-wire mode, when it is in IDLE. Applied to master mode only. 1'b0:MOSI is input. 1'b1:MOSI is output
dw_width	[15:11]	RW	5'h7	Bit width of a data word 5'h0:invalid value. others:(dw_width +1) bit/per data word Note: memory address/fifo size/ptr/ should be 1-byte aligned if dw_width =0~7 2-byte aligned if dw_width =8~15 4-byte aligned if dw_width >15

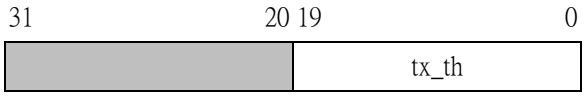
rvd	[23:16]	RW	8'h55	Reserved bit.

SPI_SCLK register: 0x218008



Name	Bit	Write/Read	Reset value	Description
clk_div	[7:0]	RW	8'h0	The SPI clock divider. Master only. SCLK frequency= SPI module's clk frequency/(clk_mul*2(clk_div+1))
clk_mul	[15:8]	RW	8'h1	The multiplier of the divided clock. Master only. 8'b0: Invalid value others: SCLK frequency= SPI module's clk frequency /(clk_mul*2(clk_div+1))

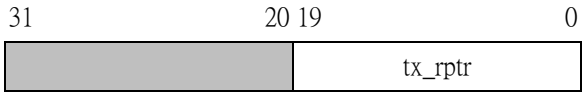
SPI_TX_TH register: 0x21800C



Name	Bit	Write/Read	Reset value	Description
tx_th	[19:0]	RW	20'h0	TX FIFO threshold. Interrupt is issued when "data number(data byte in FIFO /unit) in TX FIFO"<="tx_th". The tx_th has the following limitation. (1) In DMA/DMA FIFO mode: 0<=tx_th<=((tx_fifo_size-unit)/unit) (2) Register mode:0<=tx_th<=7 where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15

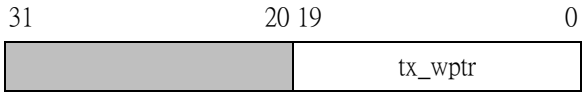
				unit=4(byte) for dw_width>15 Note: TX FIFO is the TX memory FIFO in DMA/DMA FIFO mode, and is the internal TX FIFO in Register mode.

SPI_TX_RPTR register: 0x218010



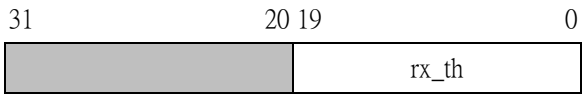
Name	Bit	Write/Read	Reset value	Description
tx_rptr	[19:0]	R	20'h0	The read pointer(byte) of the memory TX FIFO. This pointer is maintained by hardware. The return value is the offset from base address. For DMA/DMA FIFO mode only. The tx_rptr has the following property: (1)0<=tx_rptr<=(tx_fifo_size-unit) (2)tx_rptr is increased by the unit. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15

SPI_TX_WPTR register: 0x218014



Name	Bit	Write/Read	Reset value	Description
tx_wptr	[19:0]	RW	20'h0	The write pointer(byte) of the memory TX FIFO. When software writes data to the memory TX FIFO, it needs to update this pointer. The value is the offset from base address. DMA/DMA FIFO mode only. The tx_rptr has the following limitation: (1)tx_wptr is the next byte to write. Ex: if tx_wptr=0 and there are 2 byte is written, increase tx_wptr from 0 to 2. (2)0<=tx_wptr<=(tx_fifo_size-unit) (3)The memory TX FIFO is full when 1. tx_wptr=tx_rptr-unit if tx_wptr<tx_rptr 2. tx_wptr=tx_rptr+tx_fifo_size-unit if tx_wptr>tx_rptr (4)tx_wptr is increased by the unit. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15

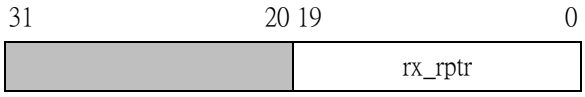
SPI_RX_TH register: 0x218018



Name	Bit	Write/Read	Reset value	Description
rx_th	[19:0]	RW	20'h0	RX FIFO threshold. Interrupt is issued when "data number(data byte in FIFO/unit) in RX FIFO">"rx_th". The rx_th has the following limitation. (1)In DMA/DMA FIFO mode: 0<=rx_th<=((rx_fifo_size-unit)/unit) (2)Register mode:0<=rx_th<=7 where unit=1(byte) for dw_width<8

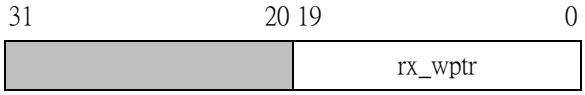
				unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15 Note: RX FIFO is the RX memory FIFO in DMA/DMA FIFO mode, and is the internal RX FIFO in Register mode.

SPI_RX_RPTR register: 0x21801C



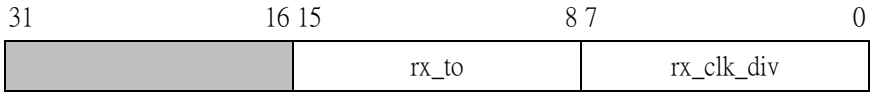
Name	Bit	Write/Read	Reset value	Description
rx_rptr	[19:0]	R	20'h0	The read pointer(byte) of the memory RX FIFO. When software reads data to the memory RX FIFO, it needs to update this pointer. The value is the offset from base address. DMA/DMA FIFO mode only. The rx_rptr has the following limitation: (1)rx_rptr is the next byte to read. Ex: if rx_rptr=0 and there are 2 byte is read, increase rx_rptr from 0 to 2. (2)0<=rx_wptr<=(rx_fifo_size-unit) (3) The memory RX FIFO is full when 1. rx_wptr=rx_rptr-unit if rx_wptr<rx_rptr 2. rx_wptr=rx_rptr+rx_fifo_size-unit if rx_wptr>rx_rptr (4)rx_rptr is increased by the unit. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15

SPI_RX_WPTR register: 0x218020



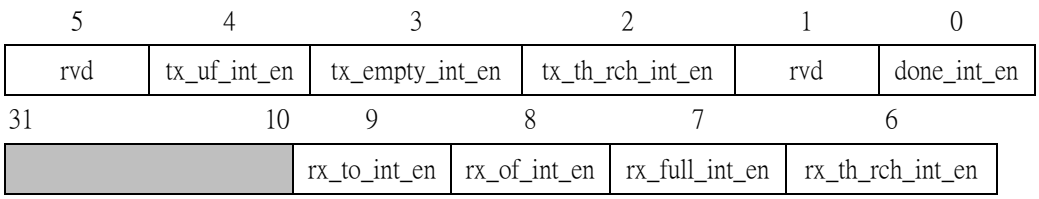
Name	Bit	Write/Read	Reset value	Description
rx_wptr	[19:0]	RW	20'h0	The write pointer(byte) of the memory RX FIFO. This pointer is maintained by hardware. The return value is the offset from base address. For DMA/DMA FIFO mode only. The tx_rptr has the following property: (1)0<=rx_wptr<=(rx_fifo_size-unit) (2)rx_wptr is increased by the unit. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15

SPI_RX_TIMEOUT register: 0x218024



Name	Bit	Write/Read	Reset value	Description
rx_clk_div	[7:0]	RW	8'h0	The divider of the divided clock of rx buffer timeout interrupt. time-out time= 32*rx_to*2* (rx_clk_div+1) * SPI module's clock period
rx_to	[15:8]	RW	8'h1	The multiplier of the divided clock of rx buffer timeout interrupt. 8'b0: invalid value. others: time-out time= time-out time= 32*rx_to*2* (rx_clk_div+1) * SPI module's clock period

SPI_INT_EN register: 0x218028



Name	Bit	Write/Read	Reset value	Description
done_int_en	[0]	RW	1'h0	Done interrupt enable 1'b0: disable 1'b1: enable
rvd	[1]	RW	1'h0	This bit should be set to 0.
tx_th_rch_int_en	[2]	RW	1'h0	TX FIFO threshold-reach interrupt enable 1'b0: disable 1'b1: enable
tx_empty_int_en	[3]	RW	1'h0	TX FIFO empty interrupt enable 1'b0: disable 1'b1: enable
tx_uf_int_en	[4]	RW	1'h0	TX underflow interrupt enable 1'b0: disable 1'b1: enable
rvd	[5]	RW	1'h0	This bit should be set to 0.
rx_th_rch_int_en	[6]	RW	1'h0	RX FIFO threshold-reach interrupt enable 1'b0: disable 1'b1: enable
rx_full_int_en	[7]	RW	1'h0	RX FIFO full interrupt enable 1'b0: disable 1'b1: enable
rx_of_int_en	[8]	RW	1'h0	RX overflow interrupt enable 1'b0: disable 1'b1: enable
rx_to_int_en	[9]	RW	1'h0	RX time out interrupt enable 1'b0: disable 1'b1: enable

SPI_INT_FLAG register: 0x21802C

5	4	3	2	1	0

rvd	tx_uf_int_flag	tx_empty_int_flag	tx_th_rch_int_flag	rvd	done_int_flag
31	10	9	8	7	6

	rx_to_int_ flag	rx_full_int_ flag	rx_full_int_ flag	rx_th_rch_int_ flag

Name	Bit	Write/Read	Reset value	Description
done_int_flag	[0]	W1C	1'h0	Transaction done interrupt. Wirte 1 to clear it. The transaction done is defined as the following: (1)Master: In DMA mode, it is set when a transaction is done. In Register/DMA FIFO mode, it is set when TX FIFO becomes empty and SPI returns to IDLE state. (2)Slave: It is set when CS is becomes inactive.
rvd	[1]	W1C	1'h0	The read out value is not defined.
tx_th_rch_int_flag	[2]	W1C	1'h0	It is set when byte count in TX FIFO reaches threshold, which is "data number(data byte in FIFO/unit) in TX FIFO"<="tx_th". Write 1 to clear it. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15 Note: TX FIFO is the TX memory FIFO in DMA/DMA FIFO mode, and is the internal TX FIFO in Register mode. Note: TX FIFO is the TX memory FIFO in DMA/DMA FIFO mode, and is the internal TX FIFO in Register mode.

tx_empty_int_flag	[3]	W1C	1'h0	It is set when TX FIFO is empty. Write 1 to clear it.
tx_uf_int_flag	[4]	W1C	1'h0	Slave mode only. It is set when data word is not ready but starts to transmit. The last data word will be transmitted again. Write 1 to clear it. Note: TX FIFO is the TX memory FIFO in DMA/DMA FIFO mode, and is the internal TX FIFO in Register mode.
rvd	[5]	W1C	1'h0	The read out value is not defined.
rx_th_rch_int_flag	[6]	W1C	1'h0	It is set when data count in RX FIFO reaches threshold, which is "data number(data byte in FIFO/unit) in RX FIFO">"rx_th". Write 1 to clear it. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15 (2)RX FIFO is the RX memory FIFO in DMA/DMA FIFO mode, and is the internal RX FIFO in Register mode.
rx_full_int_flag	[7]	W1C	1'h0	It is set when RX FIFO is full. Write 1 to clear it. Note: (1)In DMA/DMA FIFO mode the memory RX FIFO contains "rx_fifo_size-unit" byte when it is full. where unit=1(byte) for dw_width<8

				unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15 (2)RX FIFO is the RX memory FIFO in DMA/DMA FIFO mode, and is the internal RX FIFO in Register mode.
rx_of_int_flag	[8]	W1C	1'h0	Slave mode only. It is set when received buffer is full while the next data is received. The newest data will be discarded. Write 1 to clear it.
rx_to_int_flag	[9]	W1C	1'h0	It is set when the RX FIFO is non-empty and the time-out counter reaches the pre-defined time, which is 32*rx_to * 2(rx_clk_div+1) * SPI module's clock period. Write 1 to clear it. The time-out counter is re-start when In DMA/DMA FIFO mode (1)last bit of a data is received. (2)the RX read pointer is updated. In Register mode (1)last bit of a data is received. (2)the RX data register is read.

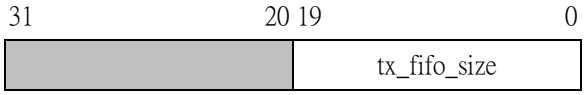
SPI_STATUS register: 0x218030

31 ... 5	4	3	2	1	0
	rx_fifo_full	rx_fifo_empty	tx_fifo_full	tx_fifo_empty	ready

Name	Bit	Write/Read	Reset value	Description
ready	[0]	R	1'h0	It indicates the SPI is ready to use. 1'b0: SPI is not ready 1'b1: SPI is ready. Ready is set by the following rule: (1)Master mode SPI is enabled and it is not transmitting or receiving data. Note: It takes 1 clock to let this bit become 0 after start bit is written 1, be sure to add 1 software nop before start to poll. (2)Slave mode SPI is enabled and CS is inactive.
tx_fifo_empty	[1]	R	1'h1	TX FIFO empty status. 1'b0: TX FIFO is not empty. 1'b1: TX FIFO is empty. Note: TX FIFO is the TX memory FIFO in DMA/DMA FIFO mode, and is the internal TX FIFO in Register mode.
tx_fifo_full	[2]	R	1'h0	TX FIFO full status. 1'b0: TX FIFO is not full. 1'b1: TX FIFO is full. Note: TX FIFO is the TX memory FIFO in DMA/DMA FIFO mode, and is the internal TX FIFO in Register mode.
rx_fifo_empty	[3]	R	1'h1	RX FIFO empty status. 1'b0: RX FIFO is not empty. 1'b1: RX FIFO is empty. Note: RX FIFO is the RX memory FIFO in DMA/DMA FIFO mode, and is the internal RX FIFO in Register mode.

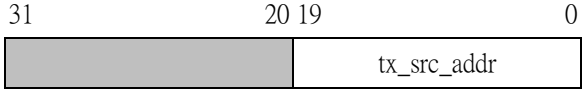
rx_fifo_full	[4]	R	1'h0	RX FIFO full status. 1'b0: RX FIFO is not full. 1'b1: RX FIFO is full. Note: RX FIFO is the RX memory FIFO in DMA/DMA FIFO mode, and is the internal RX FIFO in Register mode.

SPI_TX_FIFO_SIZE register: 0x218048



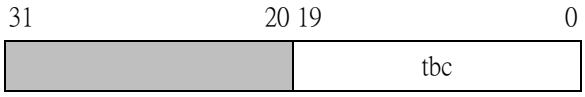
Name	Bit	Write/Read	Reset value	Description
tx_fifo_size	[19:0]	RW	20'h20	The TX memory FIFO size(byte). For DMA/DMA FIFO mode only. 14'b0: invalid value when transmitted data bye is not 0. Other : valid value Note: (1)tx_fifo_size should be multiple of the unit. (2)All the TX pointers should not exceed "tx_fifo_size-unit" . (3)When TX FIFO is full, byte number in the memory TX FIFO is "tx_fifo_size-unit". where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15

SPI_TX_SRC_ADDR register: 0x21804C



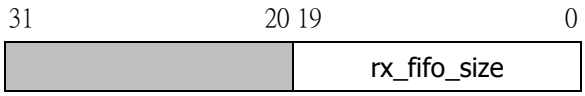
Name	Bit	Write/Read	Reset value	Description
tx_src_addr	[19:0]	RW	20'h0	The base address of the memory TX FIFO. For DMA/DMA FIFO mode only. Note: (1)Writing this register will reset tx_wptr and tx_rptr of the memory TX FIFO. (2)tx_src_addr should be multiple of the unit. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15

SPI_TBC register: 0x218050



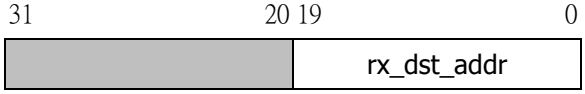
Name	Bit	Write/Read	Reset value	Description
tbc	[19:0]	RW	20'b0	Total transmitted data word count. Master's DMA mode only. 14'b0: invalid value when rbc is also set to 0. others: transmit tbc data word. Note: If there is not enough data word in TX memory FIFO, the SPI still transmits until data word in TX memory FIFO is all transmitted. After the TX memory FIFO is empty, the SPI will wait new data word written into TX memory FIFO. The transaction will end when total transmitted data word number equal tbc. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15

SPI_RX_FIFO_SIZE register: 0x218054



Name	Bit	Write/Read	Reset value	Description
rx_fifo_size	[19:0]	RW	20'h20	The RX memory FIFO size(byte). For DMA/DMA FIFO mode only. 14'b0: invalid value when received data bye is not 0. Other : valid value Note: (1)rx_fifo_size should be multiple of the unit. (2)All the RX pointers should not exceed "rx_fifo_size-unit" . (3)When RX FIFO is full, byte number in the memory RX FIFO is "rx_fifo_size-unit". where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15

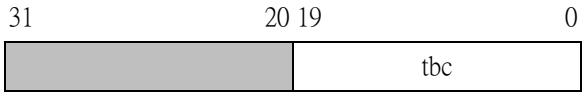
SPI_RX_DST_ADDR register: 0x218058



Name	Bit	Write/Read	Reset value	Description
rx_dst_addr	[19:0]	RW	20'h0	The base address of the memory RX FIFO. For DMA/DMA FIFO mode only. Note: (1)Writing this register will reset rx_wptr and rx_rptr of the memory RX FIFO. (2)rx_dst_addr should be multiple of the unit. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15

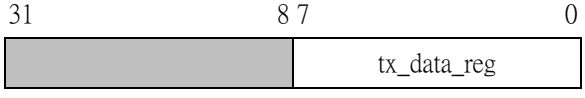
				unit=4(byte) for dw_width>15

SPI_RBC register: 0x21805C



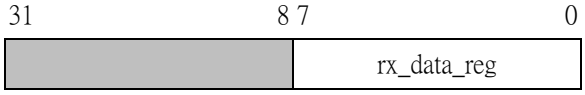
Name	Bit	Write/Read	Reset value	Description
tbc	[19:0]	RW	20'h0	Total transmitted data word count. Master's DMA mode only. 14'b0: invalid value when rbc is also set to 0. others: transmit tbc data word. Note: If there is not enough data word in TX memory FIFO, the SPI still transmits until data word in TX memory FIFO is all transmitted. After the TX memory FIFO is empty, the SPI will wait new data word written into TX memory FIFO. The transaction will end when total transmitted data word number equal tbc. where unit=1(byte) for dw_width<8 unit=2(byte) for dw_width=8~15 unit=4(byte) for dw_width>15

SPI_TX_DATA_REG register: 0x218060



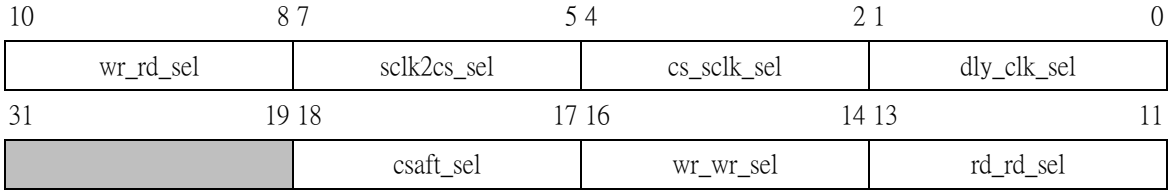
Name	Bit	Write/Read	Reset value	Description
tx_data_reg	[7:0]	W	8'h0	TX data register. Data(in unit of byte) written to this address will write to the internal TX FIFO. For Register mode only

SPI_RX_DATA_REG register: 0x218064



Name	Bit	Write/Read	Reset value	Description
rx_data_reg	[7:0]	R	8'h0	RX data register. Read this address will read the internal RX FIFO. For Register mode only. Note: If the internal RX FIFO is empty, the read out value will be the last received data.

SPI_MSTR_CONFIG register: 0x218068



Name	Bit	Write/Read	Reset value	Description
delay_sclk_sel	[1:0]	RW	2'h0	The divided clk(T) which is used in cs_sclk_sel/ wr_wr_sel/ rd_rd_sel/ wr_rd_sel/ csaft_sel 2'b00 : half SCLK period 2'b01 : SPI module's clk period 2'b10 : 2*SPI module's clk period 2'b11 : 4*SPI module's clk period

cs_sclk_sel	[4:2]	RW	3'h0	Minimum delay from the the last SCLK to CS becomes inactive. For master only. 3'b000 : 1 delay clk period. 3'b001 : 2 delay clk period. 3'b010 : 4 delay clk period. 3'b011 : 60 delay clk period. 3'b100 : 120 delay clk period. 3'b101 : 240 delay clk period. 3'b110 : 360 delay clk period. 3'b111 : 480 delay clk period. Where delay clk unit is set by delay_clk_sel in SPI_MSTR_CONFIG Register.
sclk_cs_sel	[7:5]	RW	3'h0	Minimum delay from the cs becomes active to the first SCLK. For master only. Note: T is the divided clock set by bit[1:0] in MSTRCONFIG Register. 3'b000 : 1 delay clk period. 3'b001 : 2 delay clk period. 3'b010 : 4 delay clk period. 3'b011 : 60 delay clk period. 3'b100 : 120 delay clk period. 3'b101 : 240 delay clk period. 3'b110 : 360 delay clk period. 3'b111 : 480 delay clk period. Where delay clk unit is set by delay_clk_sel in SPI_MSTR_CONFIG Register.

wr_rd_sel	[10:8]	RW	3'h0	write-read inter-data minimum delay(see Table 7-6). For Master only. 3'b000: 1 delay clk period. 3'b001: 2 delay clk period. 3'b010: 4 delay clk period. 3'b011: 60 delay clk period. 3'b100: 120 delay clk period. 3'b101: 240 delay clk period. 3'b110: 360 delay clk period. 3'b111: 480 delay clk period. Where delay clk unit is set by delay_clk_sel in SPI_MSTR_CONFIG Register.
rd_rd_sel	[13:11]	RW	3'h0	read-read inter-data minimum delay(see Table 7-6). For Master only. 3'b000: 1 delay clk period. 3'b001: 2 delay clk period. 3'b010: 4 delay clk period. 3'b011: 60 delay clk period. 3'b100: 120 delay clk period. 3'b101: 240 delay clk period. 3'b110: 360 delay clk period. 3'b111: 480 delay clk period. Where delay clk unit is set by delay_clk_sel in SPI_MSTR_CONFIG Register.

wr_wr_sel	[16:14]	RW	3'h0	Write-write inter-data minimum delay(see Table 7-6). For Master only. 3'b000: 1 delay clk period. 3'b001: 2 delay clk period. 3'b010: 4 delay clk period. 3'b011: 60 delay clk period. 3'b100: 120 delay clk period. 3'b101: 240 delay clk period. 3'b110: 360 delay clk period. 3'b111: 480 delay clk period. Where delay clk unit is set by delay_clk_sel in SPI_MSTR_CONFIG Register.
csaft_sel	[18:17]	RW	2'h0	The delay time after CS is deasserted. For Master only. 2'b00 : 1 delay clk period. 2'b01 : 2 delay clk period. 2'b10 : 4 delay clk period. 2'b11 : 8 delay clk period. Where delay clk unit is set by delay_clk_sel in SPI_MSTR_CONFIG Register.

reserved	[31:19]	RW	13'h0	These bits should be set to 0.

8 Timer and counter

8.1 32-bit timer

8.1.1 Overview

There are two 32-bit timers in AB1600. The timers are designed to count cycles of the peripheral clock (PCLK) or an external signal. It can optionally generate interrupts or perform other actions at specified timer values based on four match registers. For timer instance 0, the PCLK is provided by the core_clk or slow_clk. For timer instance 1, only the core_clk is provided. The block diagram is shown in Figure 8-1 1.

The Supported features:

- One 32-bit timer with a programmable 32-bit prescaler.
- The minimum period is 1 cycle of PCLK and the maximum period is (2^32-1) * 2^32 cycles of PCLK.

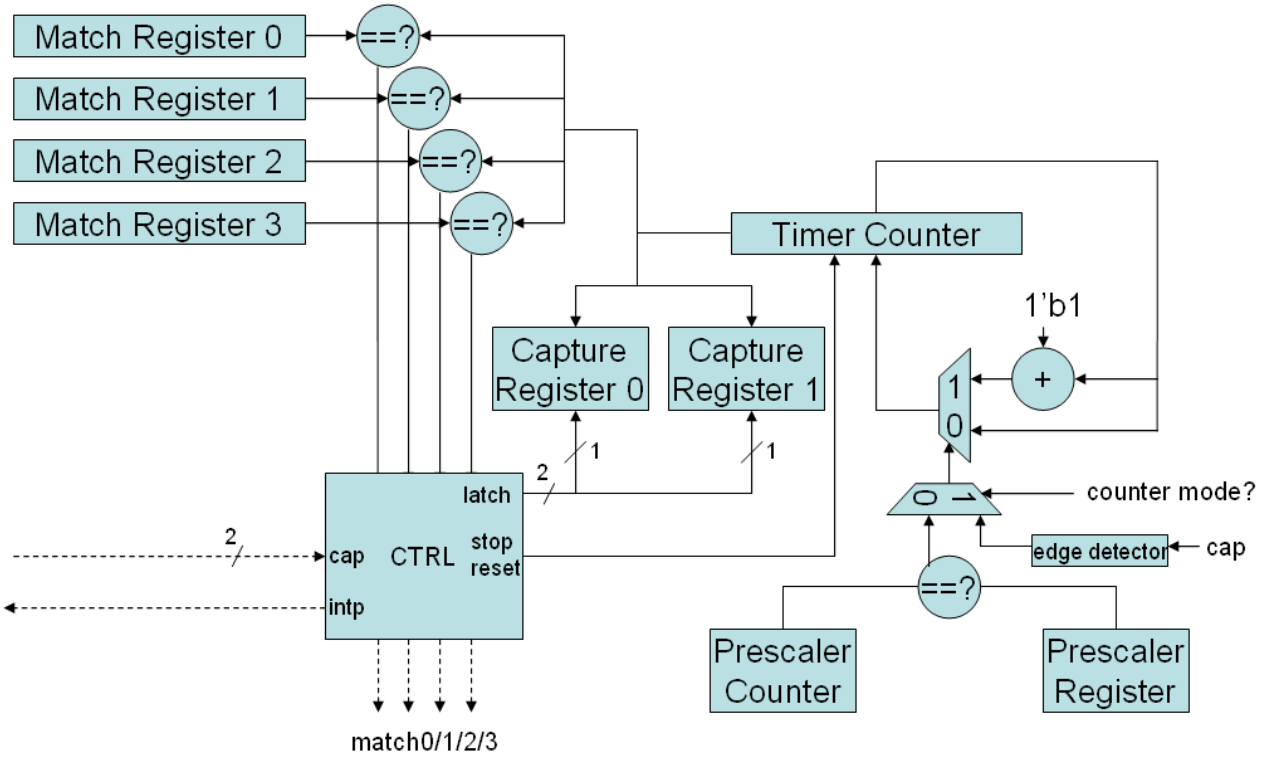


Figure 8-1 The architecture of 32-bit Timer

8.1.2 Function Descriptions

There are two modes can be used, timer mode and counter mode.
In timer mode, it counts the internal clock PCLK.
In counter mode, it counts the input signal edge (rising, falling or both edges based on setting).

There are four match registers. If it is equal to the timer(called “matched”) for each one, it can assert interrupt, and stop the timer or reset the timer. Moreover, there is corresponding output pin to each match register. The output pin can be set to high, set to low, toggle or do nothing when matched.

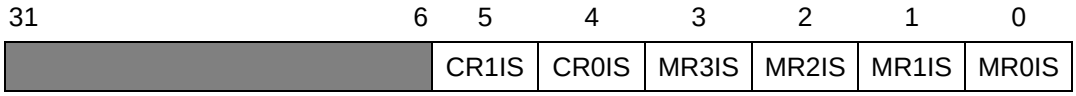
In timer mode, the timer can measure the period of a pulse from the input pin by the reset function(CTCR) and capture function(CCR). Software can configure the timer to reset the timer at the rising edge of the input signal and capture the value of the time at the falling edge. Once the value is captured, the timer can asserts an interrupt to MCU.

For instance 0, if the slow_clk is used, the interrupts of instance 0 can wake up the chip from sleep status.

8.1.3 SFR tables

Interrupt Status Register (IR - 0x220000)

Interrupt status register, store the interrupt status.



Name	Bit	Write/Read	Reset value	Description
MR0IS	[0]	W1C	0	Interrupt status of MR0. It’s set 1’b1 when the timer is equal to MR0. Write 1’b1 to clear it.
MR1IS	[1]	W1C	0	Interrupt status of MR1. It’s set 1’b1 when the timer is equal to MR1. Write 1’b1 to clear it.

MR2IS	[2]	W1C	0	Interrupt status of MR2. It's set 1'b1 when the timer is equal to MR2. Write 1'b1 to clear it.
MR3IS	[3]	W1C	0	Interrupt status of MR3. It's set 1'b1 when the timer is equal to MR3. Write 1'b1 to clear it.
CR0IS	[4]	W1C	0	Interrupt status of CR0. It's set 1'b1 when the value of timer is captured into CR0. Write 1'b1 to clear it.
CR1IS	[5]	W1C	0	Interrupt status of CR1. It's set 1'b1 when the value of timer is captured into CR1. Write 1'b1 to clear it.

Timer Control Register (TCR - 0x220004)

The TCR is used to control the Timer Counter functions. The Timer Counter can be disabled or reset through the TCR.

Name	Bit	Write/Read	Reset value	Description
CEN	[0]	R/W	0	When CEN equal to one, the Timer Counter and Prescale Counter are enabled for counting. When CEN equal to zero, the counters are disabled. Note that the CEN can be set as one by external trigger function (see SFR ETS).
CRST	[1]	R/W	0	When CRST equal to one, the Timer Counter and the Prescale Counter are synchronously reset on the next positive edge of PCLK. The counters remain reset until CRST is returned to zero.

Timer Counter (TC - 0x220008)

The Timer Counter is incremented for every (PR+1) cycles of PCLK. The TC is controlled by TCR.

31	0
TC	

Name	Bit	Write/Read	Reset value	Description
TC	[31:0]	RO	0	Timer Counter.

Prescale Register (PR - 0x22000C)

The PR specifies the maximum value for the PC.

31	0
PR	

Name	Bit	Write/Read	Reset value	Description
PR	[31:0]	R/W	0	Prescale Register.

Prescale Counter (PC - 0x220010)

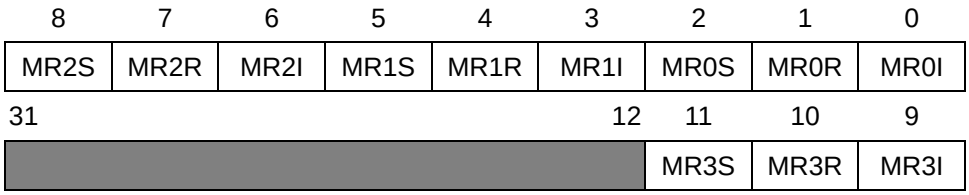
To scale down the TC. When PC reaches PR, it's cleared to 0 and the TC is incremented by 1.

31	0
PC	

Name	Bit	Write/Read	Reset value	Description
PC	[31:0]	RO	0	Prescale Counter.

Match Control Register (MCR - 0x220014)

The MCR controls what the operations performed when the TC matches the Match Registers.



Name	Bit	Write/Read	Reset value	Description
MR0I	[0]	R/W	0	An interrupt is generated when MR0 matches the value in the TC.
MR0R	[1]	R/W	0	the TC will be reset if MR0 matches TC.
MR0S	[2]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR0 matches the TC.
MR1I	[3]	R/W	0	an interrupt is generated when MR1 matches the value in the TC.
MR1R	[4]	R/W	0	the TC will be reset if MR1 matches TC.
MR1S	[5]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR1 matches the TC.
MR2I	[6]	R/W	0	an interrupt is generated when MR2 matches the value in the TC.

MR2R	[7]	R/W	0	the TC will be reset if MR2 matches TC.
MR2S	[8]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR2 matches the TC.
MR3I	[9]	R/W	0	an interrupt is generated when MR3 matches the value in the TC.
MR3R	[10]	R/W	0	the TC will be reset if MR3 matches TC.
MR3S	[11]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR3 matches the TC.

Match Register 0 (MR0 - 0x220018)

The MR0 is used to trigger operations based on MR0I,MR0R or MR0S.

31 0

MR0

Name	Bit	Write/Read	Reset value	Description
MR0	[31:0]	R/W	x	Match Register 0.

Match Register 1 (MR1 - 0x22001C)

The MR1 is used to trigger operations based on MR1I,MR1R or MR1S.

31 0

MR1

Name	Bit	Write/Read	Reset value	Description
MR1	[31:0]	R/W	x	Match Register 1.

Match Register 2 (MR2 - 0x220020)

The MR2 is used to trigger operations based on MR2I,MR2R or MR2S.

31	0
MR2	

Name	Bit	Write/Read	Reset value	Description
MR2	[31:0]	R/W	x	Match Register 2.

Match Register 3 (MR3 - 0x220024)

The MR3 is used to trigger operations based on MR3I,MR3R or MR3S.

31	0
MR3	

Name	Bit	Write/Read	Reset value	Description
MR3	[31:0]	R/W	x	Match Register 3.

Capture Control Register (CCR - 0x220028)

The CCR controls the function by Two Capture pins (cap0, cap1).

31 ~ 8	7	6	5	4	3	2	1	0
	ETS		CAP1I	CAP1FE	CAP1RE	CAP0I	CAP0FE	CAP0RE

Name	Bit	Write/Read	Reset value	Description
CAP0RE	[0]	R/W	0	When 1'b1, a rising edge (0 to 1) on CAP0 will cause CR0 to be loaded with the contents of TC
CAP0FE	[1]	R/W	0	When 1'b1, a falling edge (1 to 0) on CAP0 will cause CR0 to be loaded with the contents of TC
CAP0I	[2]	R/W	0	When 1'b1, the event of CAP0RE or CAP0FE will cause interrupt.
CAP1RE	[3]	R/W	0	When 1'b1, a rising edge (0 to 1) on CAP1 will cause CR1 to be loaded with the contents of TC
CAP1FE	[4]	R/W	0	When 1'b1, a falling edge (1 to 0) on CAP1 will cause CR1 to be loaded with the contents of TC
CAP1I	[5]	R/W	0	When 1'b1, the event of CAP1RE or CAP1FE will cause interrupt.
ETS	[6]	R/W	0	External Trigger Select. Enable CAP0 or CAP1 (based on CIS) to enable timer in timer mode (CTM==2'd0).

Capture Register 0(CR0 - 0x22002C)

The CR0 is loaded with the timer counter when the event of CAP0RE or CAP0FE occurs.

31 ~ 0

CR0

Name	Bit	Write/Read	Reset value	Description
CR0	[31:0]	RO	0	Capture Register 0

Capture Register 1(CR1 - 0x220030)

The CR1 is loaded with the timer counter when the event of CAP1RE or CAP1FE occurs.

31 0

CR1

Name	Bit	Write/Read	Reset value	Description
CR1	[31:0]	RO	0	Capture Register 1

External Match Register (EMR - 0x22003C)

The EMR provides control and status of the external match pins(EM0_PIN,EM1_PIN,EM2_PIN,EM3_PIN).

9 8 7 6 5 4 3 2 1 0

EMC2	EMC1	EMC0	EM3	EM2	EM1	EM0

31 16 15 14 13 12 11 10

	EM3_INV	EM2_INV	EM1_INV	EM0_INV	EMC3

Name	Bit	Write/Read	Reset value	Description
EM0	[0]	RO	0	External Match 0. When TC is matched by MR0, this bit acts according to EMC0.

EM1	[1]	RO	0	External Match 1. When TC is matched by MR1, this bit acts according to EMC1.
EM2	[2]	RO	0	External Match 2. When TC is matched by MR2, this bit acts according to EMC2.
EM3	[3]	RO	0	External Match 3. When TC is matched by MR3, this bit acts according to EMC3.
EMC0	[5:4]	R/W	0	External Match Control 0. Control EM0. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EMC1	[7:6]	R/W	0	External Match Control 1. Control EM1. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EMC2	[9:8]	R/W	0	External Match Control 2. Control EM2. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EMC3	[11:10]	R/W	0	External Match Control 3. Control EM3. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EM0_INV	[12]	R/W	0	Enable of EM0 inversion. 1'b0 : The value of EM0_PIN is EM0 1'b1 : The value of EM0_PIN is invert of EM0, i.e.,~EM0
EM1_INV	[13]	R/W	0	Enable of EM1 inversion. 1'b0 : The value of EM1_PIN is EM1 1'b1 : The value of EM1_PIN is invert of EM1, i.e.,~EM1

EM2_INV	[14]	R/W	0	Enable of EM2 inversion. 1'b0 : The value of EM2_PIN is EM2 1'b1 : The value of EM2_PIN is invert of EM2, i.e.,~EM2
EM3_INV	[15]	R/W	0	Enable of EM3 inversion. 1'b0 : The value of EM3_PIN is EM3 1'b1 : The value of EM3_PIN is invert of EM0, i.e.,~EM0

Counter Control Register (CTCR - 0x220070)

The CTCR is used to select between Timer and Counter mode, and in Counter mode to select the pin (CAP0,CAP1) and edge(s) for counting.

31 ... 8	7 ... 5	4	3 ... 2	1 ... 0
	SELCC	ENCC	CIS	CTM

Name	Bit	Write/Read	Reset value	Description
CTM	[1:0]	R/W	0	Counter/Timer Mode. This field selects which rising PCLK edges can trigger PC and TC. 2'd0: Timer Mode: every rising PCLK edge 2'd1: Counter Mode: rising edges on the CAP input selected by CIS. 2'd2: Counter Mode: falling edges on the CAP input selected by CIS. 3'd3: Counter Mode: both edges on the CAP input selected by CIS.
CIS	[3:2]	R/W	0	Counter Input Select. When CTM is not 2'd0, this selects which CAP pin is used: 2'd0: CAP0 2'd1: CAP1 2'd2: Reserved 2'd3: Reserved

ENCC	[4]	R/W	0	Enable of Clear Counter. When 1'b1, the PC and TC are cleared when capture edge selected in SELCC occurs.
SELCC	[7:5]	R/W	0	Select Clear Counter. When ENCC is 1'b1, the selected capture edge can clear PC and TC. 3'd0 : Rising edge of CAP0 3'd1 : Falling edge of CAP0 3'd2 : Rising edge of CAP1 3'd3 : Falling edge of CAP1 others: reserved

8.2 16-bit timer

8.2.1 Overview

The 16-bit timer is designed to count cycles of the peripheral clock (PCLK) and support PWM signal. It can optionally generate interrupts or perform other actions at specified timer values based on two sets of four match registers. The PCLK is provided by the core_clk. The block diagram is shown in Figure 8-2 2.

The Supported features:

- One 16-bit timer with a programmable 16-bit prescaler.
- The minimum period is 1 cycle of PCLK and the maximum period is (2^16-1) * 2^16 cycles of PCLK

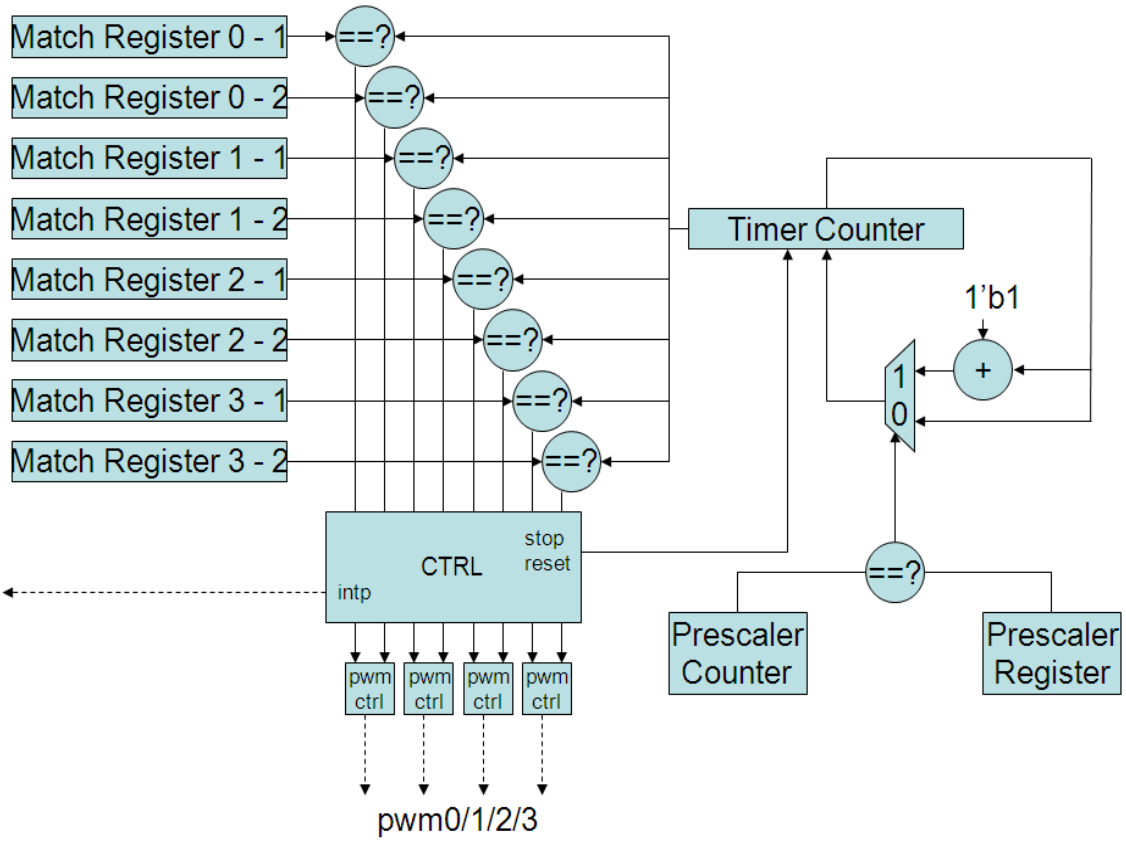


Figure 8-2 The architecture of 16-bit timer

8.2.2 Function Descriptions

Unlike 32-bit timers, there is only timer mode in 16-bit timer. It counts the internal clock PCLK.

There are two sets of four match registers. If it is equal to the timer(called “matched”) for each one, it can assert interrupt, and stop the timer or reset the timer. Moreover, there is corresponding output pin to each match register. The output pin can be set to high, set to low, toggle or do nothing when matched. Use these match registers, four independent PWM signals can be generated with the same period.

Example (shown in Fig. 5 10):

MR0=40 EMC0=2 MR0_2=50 EMC0_2=1

MR1=50 EMC1=2 MR1_2=60 EMC1_2=1

MR2=60 EMC2=2 MR2_2=70 EMC2_2=1

MR3=70 EMC3=2 MR3_2=80 EMC3_2=1 MR3R_2=1 (align last falling edge to reset TC and PC)

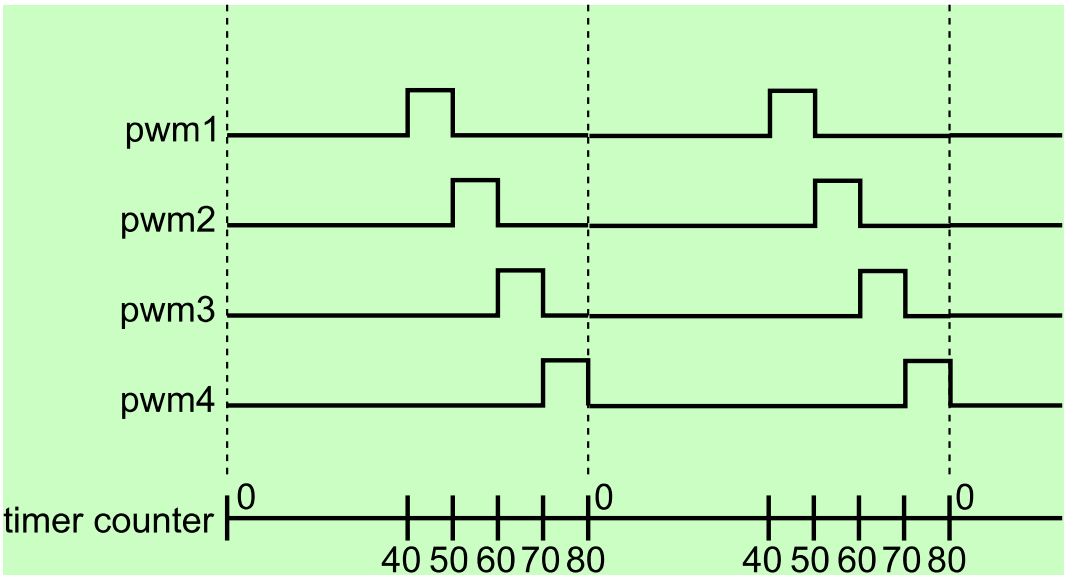


Figure 8-3 An example of 16-bit timer

8.2.3 SFR tables

Interrupt Status Register (IR - 0x222000)

Interrupt status register, store the interrupt status.

31 ... 8	7	6	5	4	3	2	1	0
	MR3IS_2	MR2IS_2	MR1IS_2	MR0IS_2	MR3IS	MR2IS	MR1IS	MR0IS

Name	Bit	Write/Read	Reset value	Description
MR0IS	[0]	W1C	0	Interrupt status of MR0. It's set 1'b1 when the timer is equal to MR0. Write 1'b1 to clear it.
MR1IS	[1]	W1C	0	Interrupt status of MR1. It's set 1'b1 when the timer is equal to MR1. Write 1'b1 to clear it.
MR2IS	[2]	W1C	0	Interrupt status of MR2. It's set 1'b1 when the timer is equal to MR2. Write 1'b1 to clear it.
MR3IS	[3]	W1C	0	Interrupt status of MR3. It's set 1'b1 when the timer is equal to MR3. Write 1'b1 to clear it.

MR0IS_2	[4]	W1C	0	Interrupt status of MR0_2. It's set 1'b1 when the timer is equal to MR0_2. Write 1'b1 to clear it.
MR1IS_2	[5]	W1C	0	Interrupt status of MR1_2. It's set 1'b1 when the timer is equal to MR1_2. Write 1'b1 to clear it.
MR2IS_2	[6]	W1C	0	Interrupt status of MR2_2. It's set 1'b1 when the timer is equal to MR2_2. Write 1'b1 to clear it.
MR3IS_2	[7]	W1C	0	Interrupt status of MR3_2. It's set 1'b1 when the timer is equal to MR3_2. Write 1'b1 to clear it.

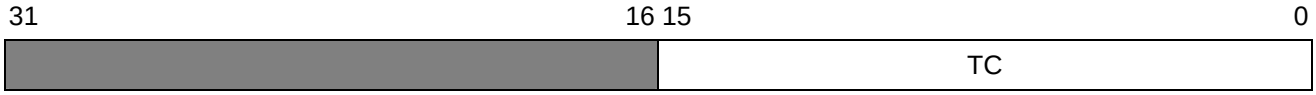
Timer Control Register (TCR - 0x222004)

The TCR is used to control the Timer Counter functions. The Timer Counter can be disabled or reset through the TCR.

Name	Bit	Write/Read	Reset value	Description
CEN	[0]	R/W	0	When CEN equal to one, the Timer Counter and Prescale Counter are enabled for counting. When CEN equal to zero, the counters are disabled. Note that the CEN can be set as one by external trigger function (see SFR ETS).
CRST	[1]	R/W	0	When CRST equal to one, the Timer Counter and the Prescale Counter are synchronously reset on the next positive edge of PCLK. The counters remain reset until CRST is returned to zero.

Timer Counter (TC - 0x222008)

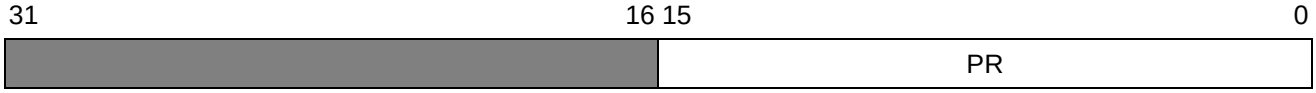
The Timer Counter is incremented for every (PR+1) cycles of PCLK. The TC is controlled by TCR.



Name	Bit	Write/Read	Reset value	Description
TC	[15:0]	RO	0	Timer Counter.

Prescale Register (PR - 0x22200C)

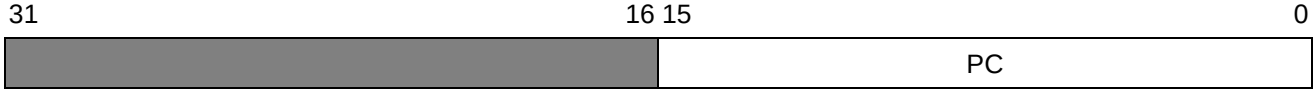
The PR specifies the maximum value for the PC.



Name	Bit	Write/Read	Reset value	Description
PR	[15:0]	R/W	0	Prescale Register.

Prescale Counter (PC - 0x222010)

To scale down the TC. When PC reaches PR, it's cleared to 0 and the TC is incremented by 1.



Name	Bit	Write/Read	Reset value	Description
PC	[15:0]	RO	0	Prescale Counter.

Match Control Register (MCR - 0x222014)

The MCR controls the what operations are performed when the TC matches the Match Registers.

8	7	6	5	4	3	2	1	0
MR2S	MR2R	MR2I	MR1S	MR1R	MR1I	MR0S	MR0R	MR0I

15	12	11	10	9
		MR3S	MR3R	MR3I

24	23	22	21	20	19	18	17	16
MR2S_2	MR2R_2	MR2I_2	MR1S_2	MR1R_2	MR1I_2	MR0S_2	MR0R_2	MR0I_2

31	28	27	26	25
		MR3S_2	MR3R_2	MR3I_2

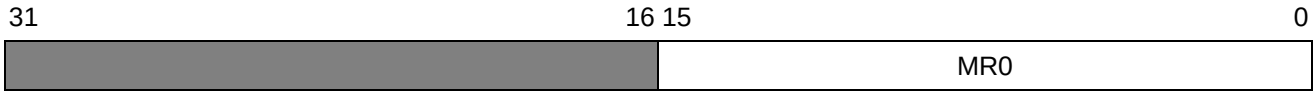
Name	Bit	Write/Read	Reset value	Description
MR0I	[0]	R/W	0	an interrupt is generated when MR0 matches the value in the TC.
MR0R	[1]	R/W	0	the TC will be reset if MR0 matches TC.
MR0S	[2]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR0 matches the TC.
MR1I	[3]	R/W	0	an interrupt is generated when MR1 matches the value in the TC.
MR1R	[4]	R/W	0	the TC will be reset if MR1 matches TC.

MR1S	[5]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR1 matches the TC.
MR2I	[6]	R/W	0	an interrupt is generated when MR2 matches the value in the TC.
MR2R	[7]	R/W	0	the TC will be reset if MR2 matches TC.
MR2S	[8]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR2 matches the TC.
MR3I	[9]	R/W	0	an interrupt is generated when MR3 matches the value in the TC.
MR3R	[10]	R/W	0	the TC will be reset if MR3 matches TC.
MR3S	[11]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR3 matches the TC.
MR0I_2	[16]	R/W	0	an interrupt is generated when MR0_2 matches the value in the TC.
MR0R_2	[17]	R/W	0	the TC will be reset if MR0_2 matches TC.
MR0S_2	[18]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR0_2 matches the TC.
MR1I_2	[19]	R/W	0	an interrupt is generated when MR1_2 matches the value in the TC.
MR1R_2	[20]	R/W	0	the TC will be reset if MR1_2 matches TC.
MR1S_2	[21]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR1_2 matches the TC.
MR2I_2	[22]	R/W	0	an interrupt is generated when MR2_2 matches the value in the TC.

MR2R_2	[23]	R/W	0	the TC will be reset if MR2_2 matches TC.
MR2S_2	[24]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR2_2 matches the TC.
MR3I_2	[25]	R/W	0	an interrupt is generated when MR3_2 matches the value in the TC.
MR3R_2	[26]	R/W	0	the TC will be reset if MR3_2 matches TC.
MR3S_2	[27]	R/W	0	the TC and PC will be stopped and CEN will be reset to 0 if MR3_2 matches the TC.

Match Register 0 (MR0 - 0x222018)

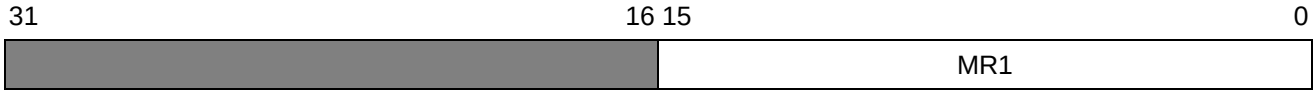
The MR0 is used to trigger operations based on MR0I,MR0R or MR0S.



Name	Bit	Write/Read	Reset value	Description
MR0	[31:0]	R/W	x	Match Register 0.

Match Register 1 (MR1 - 0x22201C)

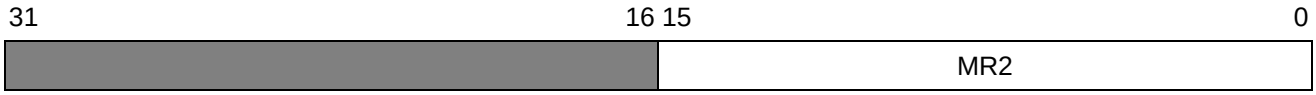
The MR1 is used to trigger operations based on MR1I,MR1R or MR1S.



Name	Bit	Write/Read	Reset value	Description
MR1	[31:0]	R/W	x	Match Register 1.

Match Register 2 (MR2 - 0x222020)

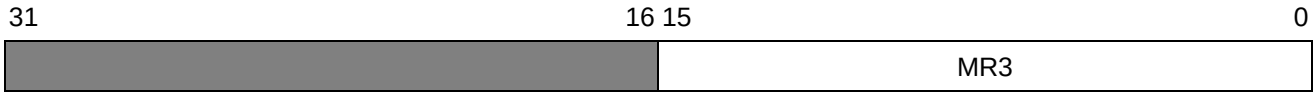
The MR2 is used to trigger operations based on MR2I,MR2R or MR2S.



Name	Bit	Write/Read	Reset value	Description
MR2	[31:0]	R/W	x	Match Register 2.

Match Register 3 (MR3 - 0x222024)

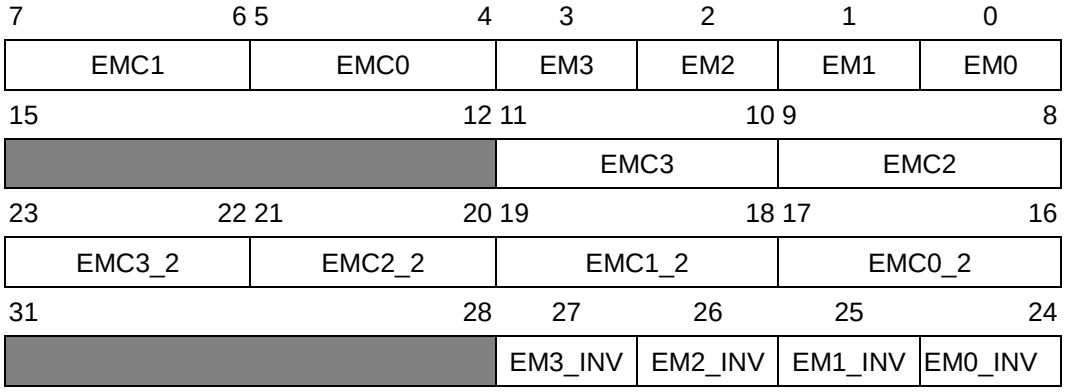
The MR3 is used to trigger operations based on MR3I,MR3R or MR3S.



Name	Bit	Write/Read	Reset value	Description
MR3	[31:0]	R/W	x	Match Register 3.

External Match Register (EMR - 0x22203C)

The EMR provides control and status of the external match pins(EM0_PIN,EM1_PIN,EM2_PIN,EM3_PIN).



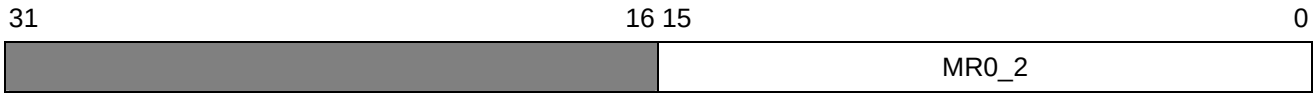
Name	Bit	Write/Read	Reset value	Description
EM0	[0]	RO	0	External Match 0. When TC is matched by MR0, this bit acts according to EMC0.
EM1	[1]	RO	0	External Match 1. When TC is matched by MR1, this bit acts according to EMC1.
EM2	[2]	RO	0	External Match 2. When TC is matched by MR2, this bit acts depends on EMC2.
EM3	[3]	RO	0	External Match 3. When TC is matched by MR3, this bit acts depends on EMC3.
EMC0	[5:4]	R/W	0	External Match Control 0. Control EM0. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EMC1	[7:6]	R/W	0	External Match Control 1. Control EM1. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1.

				2'd3 Toggle.
EMC2	[9:8]	R/W	0	External Match Control 2. Control EM2. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EMC3	[11:10]	R/W	0	External Match Control 3. Control EM3. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EMC0_2	[17:16]	R/W	0	Second External Match Control 0. Control EM0_2. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EMC1_2	[19:18]	R/W	0	Second External Match Control 1. Control EM1_2. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EMC2_2	[21:20]	R/W	0	Second External Match Control 2. Control EM2_2. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1. 2'd3 Toggle.
EMC3_2	[23:22]	R/W	0	Second External Match Control 3. Control EM3_2. 2'd0 Do Nothing. 2'd1 Set to 0. 2'd2 Set to 1.

				2'd3 Toggle.
EM0_INV	[24]	R/W	0	Enable of EM0 inversion. 1'b0 : The value of EM0_PIN is EM0 1'b1 : The value of EM0_PIN is invert of EM0, i.e.,~EM0
EM1_INV	[25]	R/W	0	Enable of EM1 inversion. 1'b0 : The value of EM1_PIN is EM1 1'b1 : The value of EM1_PIN is invert of EM1, i.e.,~EM1
EM2_INV	[26]	R/W	0	Enable of EM2 inversion. 1'b0 : The value of EM2_PIN is EM2 1'b1 : The value of EM2_PIN is invert of EM2, i.e.,~EM2
EM3_INV	[27]	R/W	0	Enable of EM3 inversion. 1'b0 : The value of EM3_PIN is EM3 1'b1 : The value of EM3_PIN is invert of EM0, i.e.,~EM0

Second Match Register 0 (MR0_2 - 0x222040)

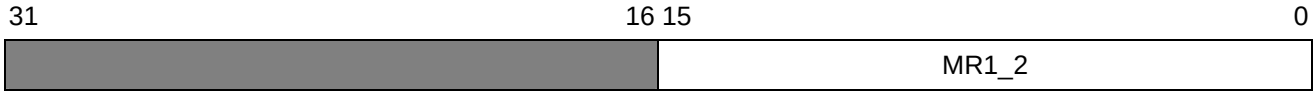
The MR0_2 is used to trigger operations based on MR0I_2,MR0R_2 or MR0S_2.



Name	Bit	Write/Read	Reset value	Description
MR0_2	[31:0]	R/W	x	Second Match Register 0.

Second Match Register 1 (MR1_2 - 0x222044)

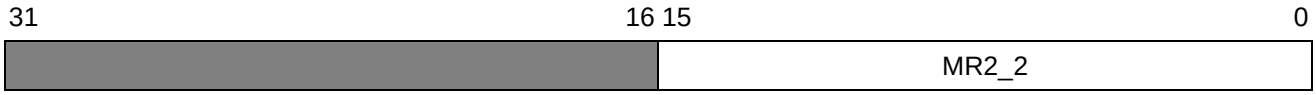
The MR1_2 is used for trigger operations based on MR1I_2,MR1R_2 or MR1S_2.



Name	Bit	Write/Read	Reset value	Description
MR1_2	[31:0]	R/W	x	Second Match Register 1.

Second Match Register 2 (MR2_2 - 0x222048)

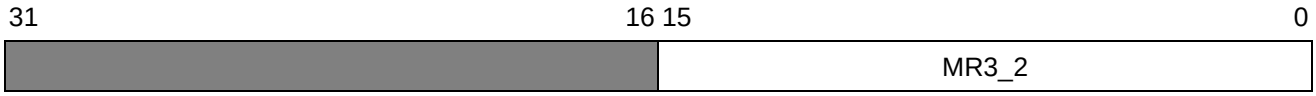
The MR2_2 is used for triggering operations based on MR2I_2,MR2R_2 or MR2S_2.



Name	Bit	Write/Read	Reset value	Description
MR2_2	[31:0]	R/W	x	Second Match Register 2.

Second Match Register 3 (MR3_2 - 0x22204C)

The MR3_2 is used for triggering operations based on MR3I_2,MR3R_2 or MR3S_2.



Name	Bit	Write/Read	Reset value	Description
MR3_2	[31:0]	R/W	x	Seconde Match Register 3.

9 Watch dog

9.1 Overview

The WatchDog Timer (WDT) is used for reseting hardware when there is something wrong happened in software so the chip can restart to work. The WDT counts down a timeout counter (called “timer” simply).When WDT reaches 0 (timeout), the watchdog reset is asserted and the hardware is reset. Software should reload the timer periodically to prevent timeout occurs. The WDT uses core_clk (fclk=16M/48M) as the clock source,and the timeout period is

(WDPR+1) * WDTC / fclk (s).

9.2 Function Descriptions

9.2.1 Feed Mechanism

The WDT uses feed function(WDFEED) to reload the timer instead of writing it directly. It can decrease the possibility that the abnormal software reload the timer coincidently. In addition, the settings WDEN, WDRESET and WDPROTECT are also protected by another feed function(WDFEED2).

9.2.2 Interrupts

There are four interrupts, which can be independently enabled by WDINT_EN.

1. Warnint. This interrupt is asserted when the timer is below or equal to the value in WDWARNINT.
2. Feed Error. This interrupt is asserted when there is an error caused by WDFEED or WDFEED2
3. Timeout. This interrupt is asserted when the timer reaches 0.
4. Protect Error. This interrupt is asserted when software attempts to modify the value of WDTC before the timer is below WDWINDOW if WDPROTECT is 1’b1.

9.3 SFR table

Watchdog Mode register (WDMOD - 0x224000)

The WDMOD controls the mode of Watchdog Timer.

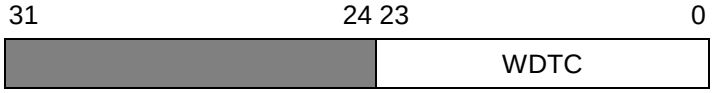
31 4	3	2	1	0
	WDPROTECT	WDTOF	WDRESET	WDEN

Name	Bit	Write/Read	Reset value	Description
WDEN	[0]	R/W	0	Watchdog enable bit. This bit can be reset only by WDFEED2. 1'b0 Disable watchdog timer. 1'b1 Enable watchdog timer. After WDEN=1, software needs to trigger WDFEED once start watchdog timer.
WDRESET	[1]	R/W	0	Watchdog reset enable bit. This bit can be reset only by WDFEED2. Watchdog timeout, feed error, or protect error will 1'b0 : NOT cause a chip reset. 1'b1 : cause a chip reset.
WDTOF	[2]	R/W	0	Watchdog time-out or error flag. It is set by a feed error, or protect error when the timer timeout. Chip will reset if WDRESET = 1. Write 1'b0 to clear this bit. Note: hardware reset it to 1'b0 only when power-on reset.
WDPROTECT	[3]	R/W	0	Watchdog protect enable. This bit can be reset only by WDFEED2. 1'b0 The watchdog reload value (WDTC) can be changed at any time. 1'b1 The watchdog reload value (WDTC)

				can be only changed after the timer is below the value of WDWINDOW. Note: this mode is intended to use only when WDRESET =1.

Watchdog Timer Constant register (WDTC - 0x224004)

The WDTC determines the timeout value (cooperates with WDPR).



Name	Bit	Write/Read	Reset value	Description
WDTC	[23:0]	R/W	0xFF	Watchdog timer constant register. This register determines the timeout period. Whenever the WDFEED is triggered by software, the value of WDTC is reloaded into the Watchdog timer. Note that if the value below 0xFF is written to WDTC, the 0xFF is written instead.

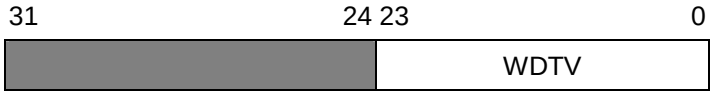
Watchdog Feed register (WDFEED - 0x224008)

Writing 0xAA followed by 0x55 to this register to reload the Watchdog timer with the value of WDTC. Note that if the Watchdog Timer hasn't been enabled, it's necessary to trigger the WDFEED sequence after the WDEN is set to enable the Watchdog Timer. After writing 0xAA, any SFR access to WDT except writing 0x55 to this SFR causes feed error.

Name	Bit	Write/Read	Reset value	Description
WDFEED	[7:0]	WO	N/A	Watchdog feed sequence register.

Watchdog Timer Value register (WDTV - 0x22400C)

This register reflects the current value (with several clocks of delay) of the Watchdog timer.



Name	Bit	Write/Read	Reset value	Description
WDTV	[23:0]	RO	0xFF	Watchdog timer value register.

Watchdog Feed register 2 (WDFEED2 - 0x224010)

Writing 0x5A followed by

0xA0 : reset WDEN,WDRESET,WDPROTECT

0xA1 : reset WDEN

0xA2 : reset WDRESET

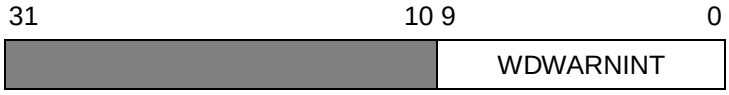
0xA3 : reset WDPROTECT

After writing 0x5A, any SFR access to WDT except writing 0xA0,A1,A2,A3 to this sfr causes feed error.

Name	Bit	Write/Read	Reset value	Description
WDFEED2	[7:0]	WO	N/A	Watchdog feed sequence register 2.

Watchdog Warning Interrupt register (WDWARNINT - 0x224014)

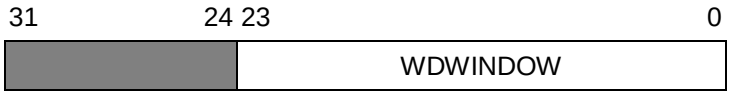
When the watchdog timer counter matches the value of WDWARNINT, an interrupt is asserted (if enabled).



Name	Bit	Write/Read	Reset value	Description
WDWARNINT	[9:0]	R/W	0	Watchdog Warning Interrupt compared value.

Watchdog Timer Window register (WDWINDOW - 0x224018)

If a feed sequence is triggered before WDTV which is smaller than or equal to the value in WDWINDOW, a feed error will occur. Moreover, the value of WDWINDOW is used by the function of WDPROTECT.



Name	Bit	Write/Read	Reset value	Description
WDWINDOW	[23:0]	R/W	0xFFFFFF	Watchdog window compared value.

Watchdog Interrupt Status (WDINT - 0x22401C)

The interrupt status of Watchdog timer. The feed error, timeout and protect error reflect the event of watchdog reset, that is, if WDRESET is 1'b1, the three status causes the watchdog reset.

31 ... 4	3	2	1	0
	Protect Error	Timeout	Feed Error	Warnint

Name	Bit	Write/Read	Reset value	Description
Warnint	[0]	W1C	0	It's set 1'b1 when the timer is below or equal to the value in WDWARNINT.
Feed Error	[1]	W1C	0	It's set 1'b1 when there is an error caused by WDFEED or WDFEED2.
Timeout	[2]	W1C	0	It's set 1'b1 when the timer reaches 0.
Protect Error	[3]	W1C	0	It's set 1'b1 when software attempts to modify the value of WDTC before the timer is below WDWINDOW if WDPROTECT is 1'b1.

Watchdog Interrupt Enable (WDINT_EN - 0x224020)

The enable parameters of interrupts for Watchdog Timer.

31 ... 4	3	2	1	0
	Protect Error Enable	Timeout Enable	Feed Error Enable	Warnint Enable

Name	Bit	Write/Read	Reset value	Description
Warnint Enable	[0]	R/W	0	The enable parameter of interrupt for Warnint. When it's 1'b1, the Warnint can assert interrupt to mcu.

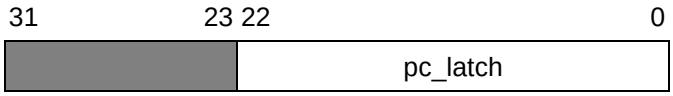
Feed Error Enable	[1]	R/W	0	The enable parameter of interrupt for Feed Error. When it's 1'b1, the Feed Error can assert interrupt to mcu.
Timeout Enable	[2]	R/W	0	The enable parameter of interrupt for Timeout. When it's 1'b1, the Timeout can assert interrupt to mcu.
Protect Error Enable	[3]	R/W	0	The enable parameter of interrupt for Protect Error. When it's 1'b1, the Protect Error can assert interrupt to mcu.

Watchdog Prescaler Register (WDPR - 0x224024)

The WDPR scales down the Watchdog Timer. The Watchdog Timer is decreased by 1 for every (WDPR+1) PCLK clocks.

Name	Bit	Write/Read	Reset value	Description
WDPR	[7:0]	R/W	0xFF	Watchdog Prescale Register. This register determines the timeout period.

pc_latch (0x224028)



Name	Bit	Write/Read	Reset value	Description
pc_latch	[22:0]	RO	0	Program Counter Latch. When WARNINT, Feed Error or Protect Error occurs, the program counter of mcu is latched. This register is reset only by power-on reset

10 LED control

10.1 Overview

The LED controls the output pin with certain PWM waveform. There are two independent LED controllers and the clock source can be selected by slow_clk or core_clk(fclk=16M/48M) (refer to section 4.3).

10.2 Function Descriptions

10.2.1 Complete Period

A complete period of PWM is composed of T1(controlled by ledx_t1), T2(controlled by ledx_t2) and T3(controlled by ledx_t3). T1 is the light-on time, T2 is the light-off time and T3 is the wait time during light-off time and next light-on time (if exist) (assuming ledx_t1t2repeat=0, see SFR table). Note that T0(controlled by ledx_t0) is initial delay time, which is not included in a complete period. See Figure 10-1 .

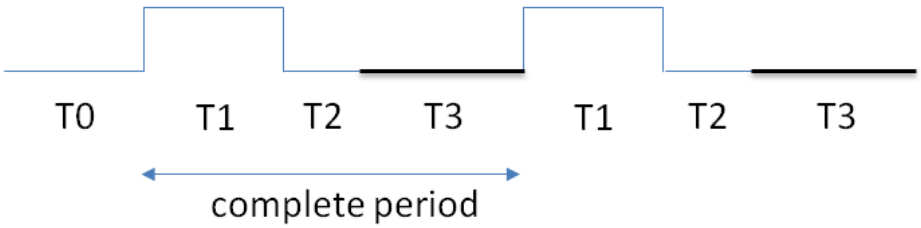


Figure 10-1 Complete period of LED

10.2.2 Time unit

The unit of ledx_t0, ledx_t1, ledx_t2 and ledx_t3 is determined by ledx_unit and ledx_xn. The corresponding real time is (2^ledx_xn * ledx_unit / fclk).

10.2.3 PWM modes

The PWM can be configured as fixed-duty or ramping-duty mode. The duty is defined during every 16 clocks and is controlled by ledx_pwm_duration_set, which is called final duty. The mode is determined by ledx_pwm_onstep and ledx_pwm_offstep. When ledx_pwm_onstep and ledx_pwm_offstep are both 0, it's in

fixed-duty mode. The LED lights with final duty when entering T1 state and lights with duty 0 when entering T2 state. When ledx_pwm_onstop or ledx_pwm_offstep is not 0, the duty of PWM gradually increases to final duty. The duty of PWM entering T1 state or gradually decrease to duty 0 when entering T2 state. Figure 10-2 shows the mechanism.

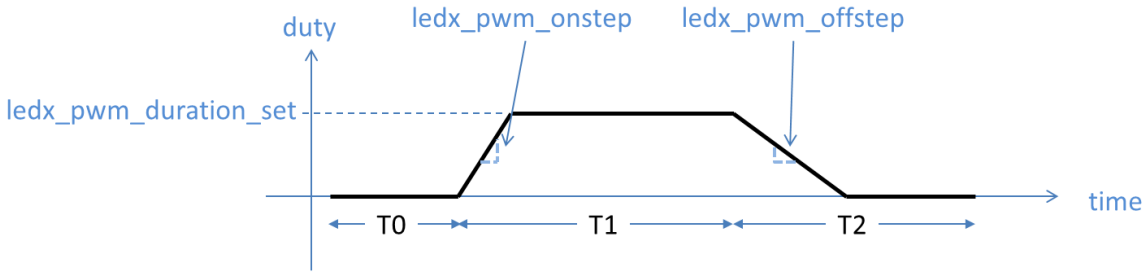


Figure 10-2 The mechanism of duty in PWM mode.

10.2.4 Follow function

The led support the following function. The function is enabled by set ledx_fw_sel. The following time is determined by ledx_fw_t1t2. The ledx_fw_retrig determines whether to re-follow even when the current waveform is not completed, where the definition of “completed” is different in fixd-duty mode and ramping-duty mode. Figure 10-3 shows LED2 follows LED1.

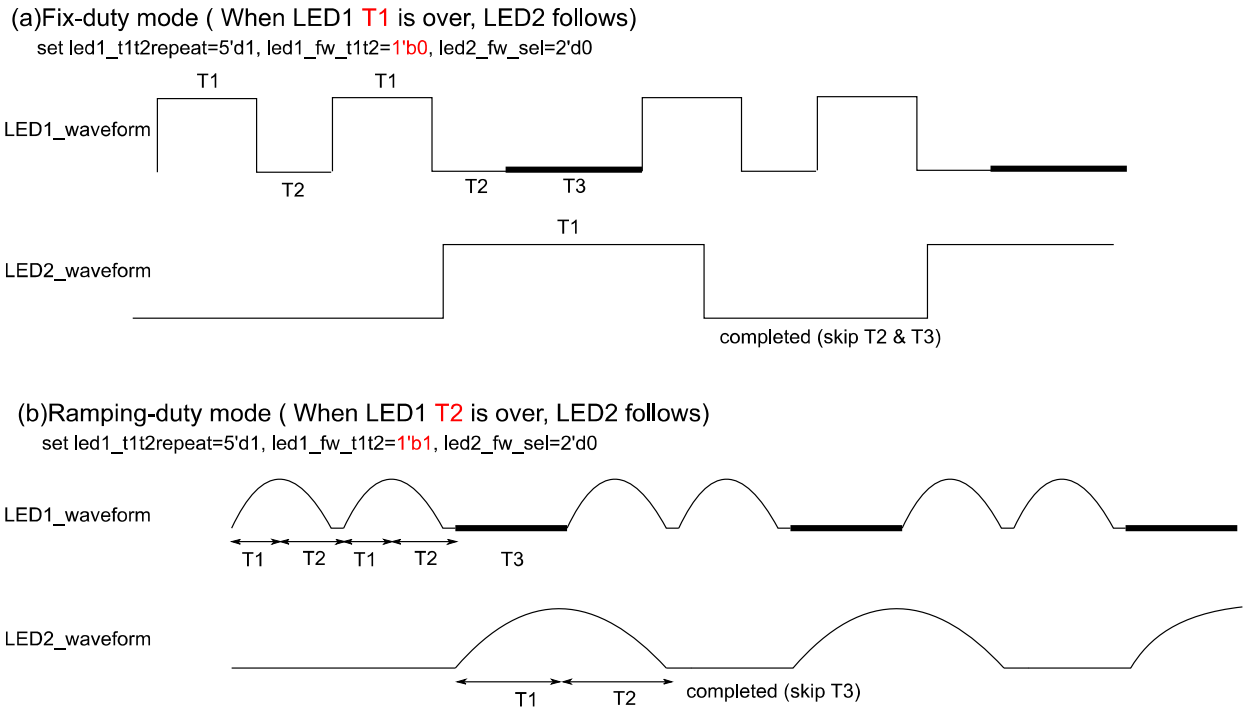


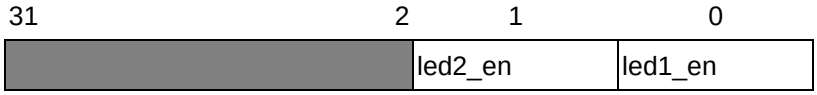
Figure 10-3 The follow function of LED

10.3 I/O configure

Refer to section 4.7.

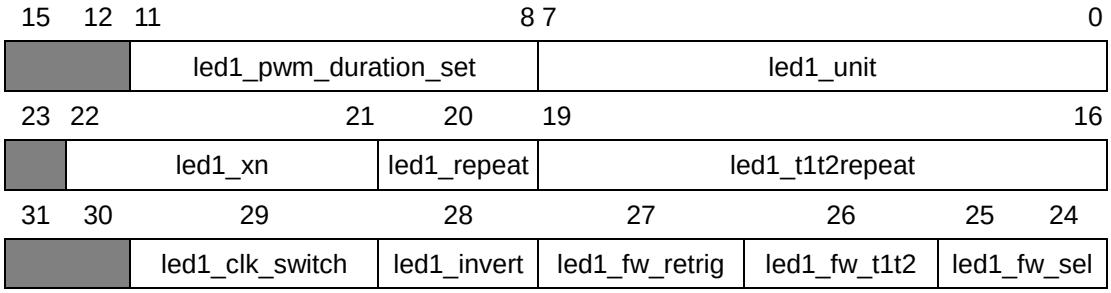
10.4 SFR table

Led Enable (0x225000)



Name	Bit	Write/Read	Reset value	Description
led2_en	[1]	WR	0	Enable of led2
led1_en	[0]	WR	0	Enable of led1

Led1 ctrl (0x225010)



Name	Bit	Write/Read	Reset value	Description
led1_clk_switch	[29]	WR	0	Select clock of led1 between mcu clk or pmu clk 1'b0 : slow_clk (32k) 1'b1 : core_clk (16/48M, refer to section 4.3)
led1_invert	[28]	WR	0	to invert the output waveform 1'b0 : none 1'b1 : invert
led1_fw_retrig	[27]	WR	0	led follow retrigger. If the current waveform hasn't completed and new interrupt comes, 1'b0 : the waveform continues until it's done (ignore this interrupt) 1'b1 : the waveform restarts
led1_fw_t1t2	[26]	WR	0	Follow timing. 1'b0 : other led follow led1 when T1 is finished 1'b1 : other led follow led1 when T2 is finished
led1_fw_sel	[25:24]	WR	0	led follow select. Determine which led to follow. 2'd0 : disable(default) 2'd1 : follow led2 2'd2 : follow led3 2'd3 : follow led4
reserved	[23]	WR	0	
led1_xn	[22:21]	WR	0	led multiplier, to amplify the led1_unit additionally. 2'd0 : 1x 2'd1 : 2x 2'd2 : 4x 2'd3 : 8x
led1_repeat	[20]	WR	0	led repeat the configured complete period waveform forever 1'b0 : no repeat 1'b1 : repeat

led1_t1t2repeat	[19:16]	WR	0	the repeat times of T1 & T2 during one complete period. For example, if this value is 2, then one complete period is T0,T1,T2,T1,T2,T3.
reserved	[15:12]	WR	0	
led1_pwm_duration_set	[11:8]	WR	4'hF	Final duty of led1_waveform during every 16 clks. If led1_pwm_onstep is not zero, the duty increases smoothly; otherwise, duty goes from 0 to this value directly. If led1_pwm_offstep is not zero, the duty decreases smoothly; otherwise, duty goes from this value to 0 directly. 4'hF : 100% 4'hE : 14/16 ... 4'h2 : 2/16 4'h1 : 1/16 4'h0 : invalid (led never on)
led1_unit	[7:0]	WR	8'd16	The time unit of led1_t0, led1_t1, led1_t2 and led1_t3. The corresponding real time is (2^led1_xn * led1_unit / fclk) second. The fclk is the frequency of clk and the clk is determined by led1_clk_switch

Led1 PWM ramping step (0x225014)

31 16	15 8	7 0
	led1_pwm_offstep	led1_pwm_onstep

Name	Bit	Write/Read	Reset value	Description
led1_pwm_offstep	[15:8]	WR	0	Duty-decreased step. Determine how fast duty decrease to 0. 8'd0 : disable breathe (final duty->0 directly) others : every (64*led_pwm_offstep/fclk) seconds to decrease 1/16 duty
led1_pwm_onstep	[7:0]	WR	0	Duty-increased step. Determine how fast duty reaches led1_pwm_duration_set. 8'd0 : disable breathe (0->final duty directly) others : every (64*led_pwm_onstep/fclk) seconds to increase 1/16 duty

Led1 period (0x225018)

31 24	23 16	15 8	7 0
led1_t0	led1_t1	led1_t2	led1_t3

Name	Bit	Write/Read	Reset value	Description
led1_t0	[31:24]	WR	x	period of T0 (initial wait period)
led1_t1	[23:16]	WR	x	period of T1 (led on period)
led1_t2	[15:8]	WR	x	period of T2 (led off period)
led1_t3	[7:0]	WR	x	period of T3 (final wait period)

Led2 ctrl (0x225020)

15 12	11 8	7 0

		led2_pwm_duration_set										led2_unit											
23	22									21	20	19											16
		led2_xn								led2_repeat		led2_t1t2repeat											
31	30							29				28	27					26			25	24	
		led2_clk_switch								led2_invert		led2_fw_retrig						led2_fw_t1t2			led2_fw_sel		

Name	Bit	Write/Read	Reset value	Description
led2_clk_switch	[29]	WR	0	Select clock of led2 between mcu clk or pmu clk 1'b0 : slow_clk (32k) 1'b1 : core_clk (16/48M, refer to section 4.3)
led2_invert	[28]	WR	0	to invert the output waveform 1'b0 : none 1'b1 : invert
led2_fw_retrig	[27]	WR	0	led follow retrigger. If the current waveform hasn't completed and new interrupt comes, 1'b0 : the waveform continues until it's done (ignore this interrupt) 1'b1 : the waveform restarts
led2_fw_t1t2	[26]	WR	0	Follow timing. 1'b0 : other led follow led2 when T1 is finished 1'b1 : other led follow led2 when T2 is finished
led2_fw_sel	[25:24]	WR	1	led follow select. Determine which led to follow. 2'd0 : follow led1 2'd1 : disable(default) 2'd2 : follow led3 2'd3 : follow led4
reserved	[23]	WR	0	
led2_xn	[22:21]	WR	0	led multiplier, to amplify the led2_unit additionally. 2'd0 : 1x 2'd1 : 2x 2'd2 : 4x 2'd3 : 8x

led2_repeat	[20]	WR	0	led repeat the configured complete period waveform forever 1'b0 : no repeat 1'b1 : repeat
led2_t1t2repeat	[19:16]	WR	0	the repeat times of T1 & T2 during one complete period. For example, if this value is 2, then one complete period is T0,T1,T2,T1,T2,T3.
reserved	[15:12]	WR	0	
led2_pwm_duration_set	[11:8]	WR	4'hF	Final duty of led2_waveform during every 16 clks. If led2_pwm_onstep is not zero, the duty increases smoothly; otherwise, duty goes from 0 to this value directly. If led2_pwm_offstep is not zero, the duty decreases smoothly; otherwise, duty goes from this value to 0 directly. 4'hF : 100% 4'hE : 14/16 ... 4'h2 : 2/16 4'h1 : 1/16 4'h0 : invalid (led never on)
led2_unit	[7:0]	WR	8'd16	The time unit of led2_t0, led2_t1, led2_t2 and led2_t3. The corresponding real time is (2^led2_xn * led2_unit / fclk) second. The fclk is the frequency of clk and the clk is determined by led2_clk_switch

Led2 pwm ramping step (0x225024)

31 16	15 8	7 0
	led2_pwm_offstep	led2_pwm_onstep

Name	Bit	Write/Read	Reset value	Description
led2_pwm_offstep	[15:8]	WR	0	Duty-decreased step. Determine how fast duty decrease to 0. 8'd0 : disable breathe (final duty->0 directly) others : every (64*led_pwm_offstep/fclk) seconds to decrease 1/16 duty
led2_pwm_onstep	[7:0]	WR	0	Duty-increated step. Determine how fast duty reaches led1_pwm_duration_set. 8'd0 : disable breathe (0->final duty directly) others : every (64*led_pwm_onstep/fclk) seconds to increase 1/16 duty

Led2 period (0x225028)

31 24	23 16	15 8	7 0
led2_t0	led2_t1	led2_t2	led2_t3

Name	Bit	Write/Read	Reset value	Description
led2_t0	[31:24]	WR	x	period of T0 (initial wait period)
led2_t1	[23:16]	WR	x	period of T1 (led on period)
led2_t2	[15:8]	WR	x	period of T2 (led off period)
led2_t3	[7:0]	WR	x	period of T3 (final wait period)

11 Analog to Digital Converter (ADC)

11.1 Analog to Digital Converter (ADC)

The type of ADC is a sigma-delta ADC. It offers two modes for different applications, including the MIC mode and the AIO mode.

11.1.1 MIC mode

The MIC mode offers two differential input signals, named MIC_P and MIC_N, and one output signal for MIC bias. The ADC's output through the filter and then the data into the memory for software developers use.

11.1.2 AIO mode

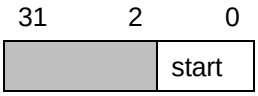
The AIO mode offers 32 channels for analog measurements, 26 signals from analog IOs and 6 signals for internal signals, such as VBAT etc.

(1) ADC value register: 0x21D000~0x21D07C

address	31 13	1 2 0
0x21D000		ADC channel 0 value
0x21D004		ADC channel 1 value
0x21D008		ADC channel 2 value
0x21D00C		ADC channel 3 value
0x21D010		ADC channel 4 value
0x21D014		ADC channel 5 value
0x21D018		ADC channel 6 value
0x21D01C		ADC channel 7 value
0x21D020		ADC channel 8 value
0x21D024		ADC channel 9 value
0x21D028		ADC channel 10 value
0x21D02C		ADC channel 11 value
0x21D030		ADC channel 12 value
0x21D034		ADC channel 13 value
0x21D038		ADC channel 14 value
0x21D03C		ADC channel 15 value
0x21D040		ADC channel 16 value
0x21D044		ADC channel 17 value
0x21D048		ADC channel 18 value
0x21D04C		ADC channel 19 value
0x21D050		ADC channel 20 value
0x21D054		ADC channel 21 value
0x21D058		ADC channel 22 value
0x21D05C		ADC channel 23 value
0x21D060		ADC channel 24 value
0x21D064		ADC channel 25 value
0x21D068		ADC channel 26 value
0x21D06C		ADC channel 27 value
0x21D070		ADC channel 28 value
0x21D074		ADC channel 29 value
0x21D078		ADC channel 30 value
0x21D07C		ADC channel 31 value

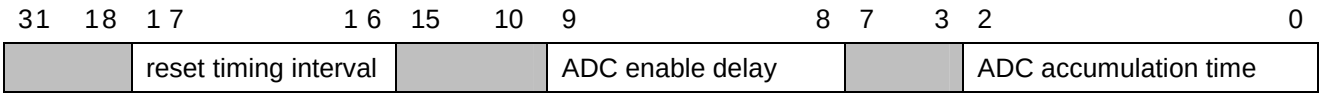
Name	Bit	Write/Read	Reset value	Description
ADC channel X value	[12:0]	R	13'b0	ADC channel X data

(2) ADC start register: 0x21D080



Name	Bit	Write/Read	Reset value	Description
start / busy	[0]	W1AC/R	0	write 1 to start pulse [0]: after write 1 to start pulse, read [0] until 0 to know start pulse finished

(3) ADC setting register: 0x21D084



Name	Bit	Write/Read	Reset value	Description
ADC accumulation time	[2:0]	WR	3'b000	Accumulation time 3'b000: 16 3'b001: 64 3'b010: 256 3'b100: 1024 3'b101: 4096
ADC enable delay	[9:8]	WR	2'b01	ADC enable delay 2'b00: 200T 2'b01: 250T 2'b10: 300T 2'b11: 350T
reset timing interval	[17:16]	WR	2'b00	Reset rising to start sample adc1 timing interval 2'b00: 20T 2'b01: 30T

				2'b10: 40T 2'b11: 50T

1T = one system clock cycle

(4) ADC channel selection register: 0x21D088

31 0

ADC channel selection

Name	Bit	Write/Read	Reset value	Description
ADC channel selection	[31:0]	WR	32'b0	ADC channel enable

(5) ADC interrupt enable register: 0x21D08C

31 1 0

	ADC interrupt enable

Name	Bit	Write/Read	Reset value	Description
ADC interrupt enable	[0]	WR	1'b0	ADC interrupt enable

(6) ADC interrupt clear register: 0x21D090

31 1 0

	ADC interrupt clear

Name	Bit	Write/Read	Reset value	Description
ADC interrupt clear	[31:0]	W1C	NA	ADC interrupt clear

(7) ADC rck done register: 0x21D094

31 1 0

	ADC rck done

Name	Bit	Write/Read	Reset value	Description
ADC rck done	[0]	R	1'b0	if this equals one, rck calibration done and can use ADC

12 Key scan

12.1 Overview

Keyscan supports a maximum 8 row*18 column key matrix and provides debounced results. Key matrix is connected to the keyscan controller by GPIO. By setting the I/O configuration, GPIOs can be assigned to be specific row/column. At each key event , up to 8 keys can be stored, and an interrupt is issued. A 32 level internal key FIFO is designed to buffer these key events. The block diagram of Keyscan is shown in Figure 12-1 .

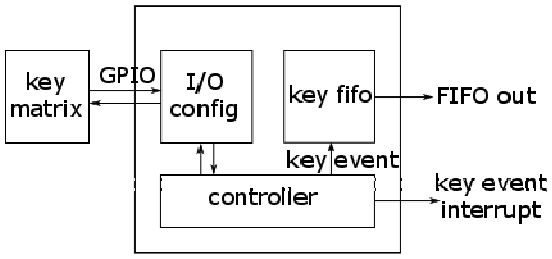


Figure 12-1 The Block Diagram of Keyscan

12.2 Operation

When the keyscan function is used, a key matrix is connected to the keyscan controller through GPIO. A key matrix consists of key buttons which act as switches. In each key, one terminal connects to a row and the other connects to a column, which is shown in Figure 12-2 and Figure 12-3 .

Columns are the input to the keyscan controller and are pulled to VDD by pull-up resisters. Rows are controlled by the keyscan controller. If a row is scanned, the keyscan controller sink the row to 0V; the remaining rows are pulled to VDD by pull-up resisters. If a key in the scanned row is pressed, the column is sunk to 0V. On the other hand, if a key in the scanned row is not pressed, the column remains VDD. By detecting the column voltage, the keyscan controller can detect which key is pressed.

The fast scan rate of the controller is row_num*(col_num+4)*128k clock period(it is 1.38ms in 8*18 case).

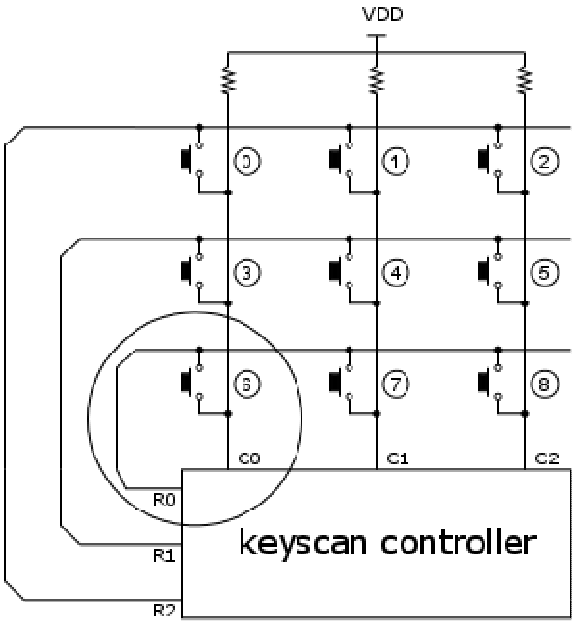


Figure 12-2 Keyscan controller topology

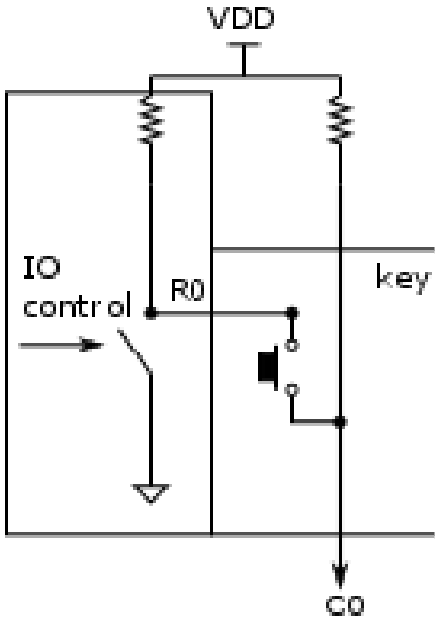


Figure 12-3 Detail of a key

12.3 Ghost Key

If there are more than 3 keys pressed simultaneously, ghosts keys may happen. Consider the case that key R0C0,R2C0,R2C2 are pressed, and R0 is scanned and sunk to 0V, which is shown in Figure 12-4 . In this case, C2 is sunk to 0V due to the shorted path from R0C0->R2C0->R2C2. Keyscan controller may consider R0C2 is pressed although it is not. The extra detected key is called a ghost key, and the key event is called

ghost key events. In this ghost key example, key R0C0, R0C2, R2C0, R2C2 and a ghost key indicator will be stored to the keyscan FIFO.

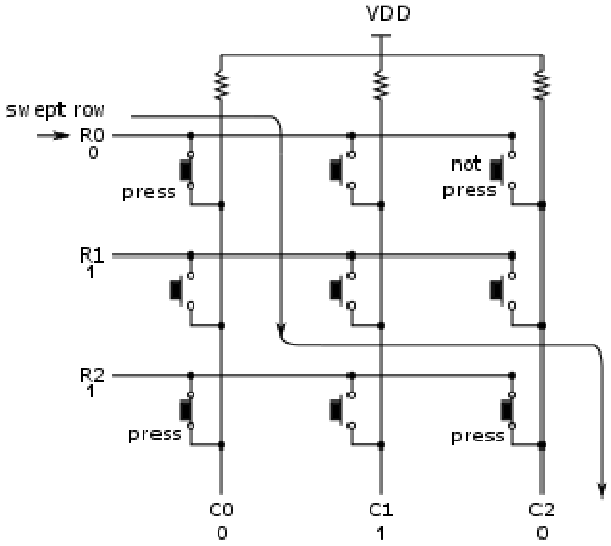


Figure 12-4 Ghost keys

12.4 Key Code

Key code is an 8-bit data representing either a specific key or an end of data indicator, which ranges from 0 to 255. Key codes which are greater or equal to 247 are used as end of data indicator to separate different key events and indicate the event type. Other key codes are used to represent a specified key.

The key code of a specific key in a key matrix increases from the smallest column to the largest column and from the smallest row to the largest row, which is shown in Figure 12-5 .

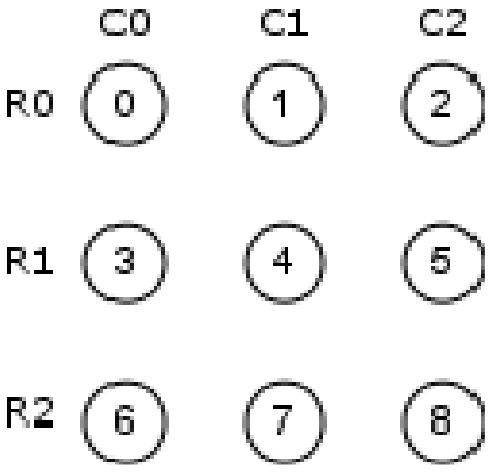


Figure 12-5 Diagram of a key matrix with it key code

The end of data(EOD) indicator is used to separate different key events and as an indicator to indicates the event type. The EODs are encoded by key codes greater or equal to 247. There are 8 types of EOD which is listed in Table 12-1 .

EOD name	Key code	Data stored in the key event	Description
Key release	255	EOD255	Key released event
FIFO overflow	254	EOD254	FIFO overflow event. It means there are keys missing due to key FIFO overflows
FIFO Full due to key more than 8 event	253	8 Key+EOD253	Key FIFO is full, and the event to let the FIFO full is a key more than 8 event. No key is discarded in this case but the keyscan stop scanning if the FIFO remains full.
FIFO Full due to normal key event	252	Key+EOD252	Key FIFO is full, and the event to let the FIFO full is a normal key event. No key is discarded in this case but the keyscan stop scanning if the FIFO remains full.
FIFO Full due to a released event	251	EOD251	Key FIFO is full, and the event to let the FIFO full is a key released event. No key is discarded in this case but the keyscan stop scanning if the FIFO remains full.
FIFO Full due to a ghost key event	250	Key+EOD250	Key FIFO is full, and the event to let the FIFO full is a ghost key event. No key is discarded in this case but the keyscan stop scanning if the FIFO remains full.
Ghost key	249	Key+EOD249	Ghost key.
Key more than 8	248	8 Key+EOD248	The scanned key numbers in that event is greater than 8. Only 8 key is stored and the rest is discarded.

Normal	247	Key+EOD247	The scanned key numbers in that event is less or equal than 8. It is the normal case.

Table 12-1 EOD table

12.5 Key FIFO

There is a 32-level key FIFO in the keyscan controller. The scanned keys and an EOD are stored to the key FIFO serially in every key event. For example, if key1,key2 are pressed, key1,key2 and EOD247 will be stored in the Key FIFO. This is shown in Figure 12-6 . Host can process these keys depending on different type of EOD.

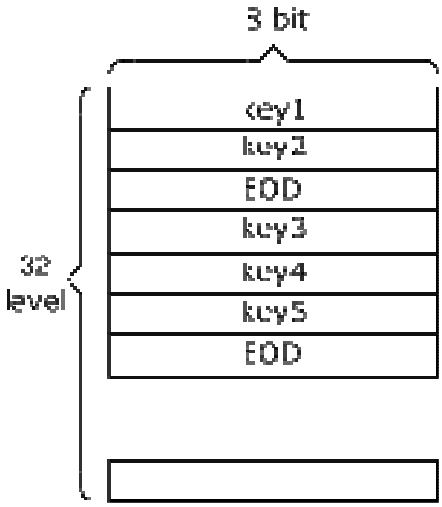


Figure 12-6 Key FIFO

12.6 Debounce Time

Due to the mechanical structure of keys, whan a key is pressed or released there is a transient bouncing. To filter the scanned value during the bouncing period, a technic called debounce is applied to the scanned result. The keyscan controller will not report key events until the same scanned results obtain in the consecutive pre-defined times. The debounce times of pressing key and releasing key can be set individually by the SDK??. Figure 12-7 shows an example of debounce.

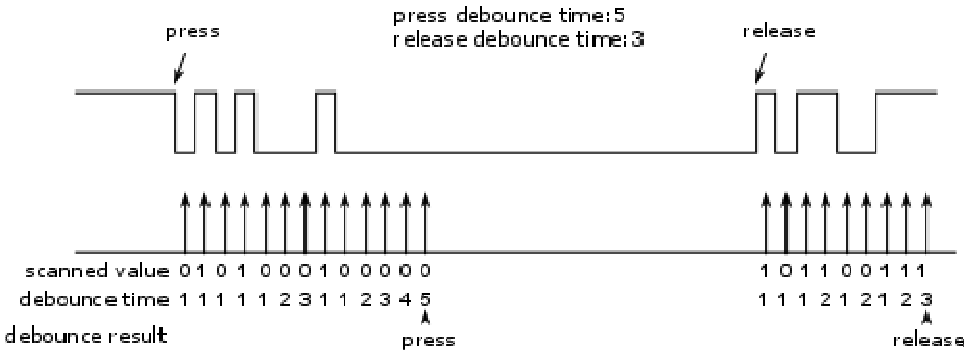


Figure 12-7 Debouncer

12.7 Interrupt

The keyscan controller provides an interrupt, keyscan_int, to indicate a key event happens.

12.8 SFR Table

KS_INT_EN register: 0x226008

31	1	0
		keyscan_int_en

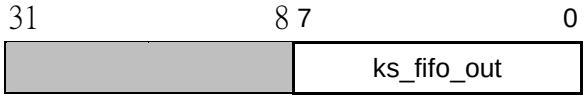
Name	Bit	Write/Read	Reset value	Description
keyscan_int_en	[0]	WR	1'h0	Keyscan interrupt enable. 1'b0:disable 1'b1:enable

KS_INT_FLAG register: 0x22600C

31	1	0
		keyscan_int_flag

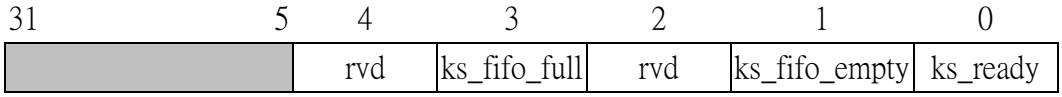
Name	Bit	Write/Read	Reset value	Description
keyscan_int_flag	[0]	W1C	1'h0	Keyscan interrupt flag. Write 1 to clear it.

KS_FIFO_OUT register: 0x226010



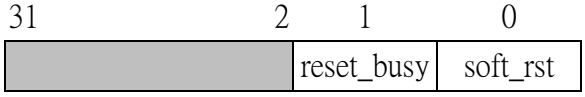
Name	Bit	Write/Read	Reset value	Description
keyscan_fifo_out	[7:0]	R	8'h0	Keyscan fifo output. Each byte represents either a key or a EOD. This register is valid when ks_fifo_empty in KS_STATUS is not 0.

KS_STATUS register: 0x226014



Name	Bit	Write/Read	Reset value	Description
ks_ready	[0]	R	1'h1	keyscan ready. 1'b1:keyscan is not scanning. 1'b0:keyscan is scanning.
ks_fifo_empty	[1]	R	1'h1	keyscan fifo empty. 1'b1:fifo is empty. 1'b0:fifo is not empty.
ks_fifo_full	[2]	R	1'h0	keyscan fifo full. 1'b1:fifo is full. 1'b0:fifo is not full.
rvd	[3]	R	1'h0	this bit is undefined.

KS_RESET register: 0x226018



Name	Bit	Write/Read	Reset value	Description
soft_rst	[0]	WC	1'h0	wirte 1 to reset keyscan and keyscan fifo. Note: (1) Interrupt flag is not reset.

				(2) Turn off keyscan controller before soft_rst.
reset_busy	[1]	R	1'h0	keyscan reset busy. Polling this bit until 0 to ensure soft_rst is done. 1'b1:keyscan is being reset. 1'b0:keyscan has been reset.

13 Mouse

The Mouse supports 3-axis detection. The xy-axis detects the movement of mouse and the z-axis detect the rotation of wheel.

13.1 XY-axis

13.1.1 Overview

The x-axis and y-axis use two input pins respectively. Because the behavior of x-axis and y-axis are the same, only the x-axis is described in the remaining section.

13.1.2 Function Descriptions

The detection of x-axis is to compare the two inputs of x-axis(say x1 and x2). According to the criterion in Fig 13-1 and Table 13-1, hardware stores the counted value in a counter(x_count) and software can determine the movement of the mouse. The +1/-1 event occurs when (1) x1 or x2 change and (2) lasts for 30us without other x2 or x1 change. Figure 13-1 and Table 13-1 shows the details in case of x_dir_sel=0, if x_dir_sel=1, the value 1 and -1 are swapped.

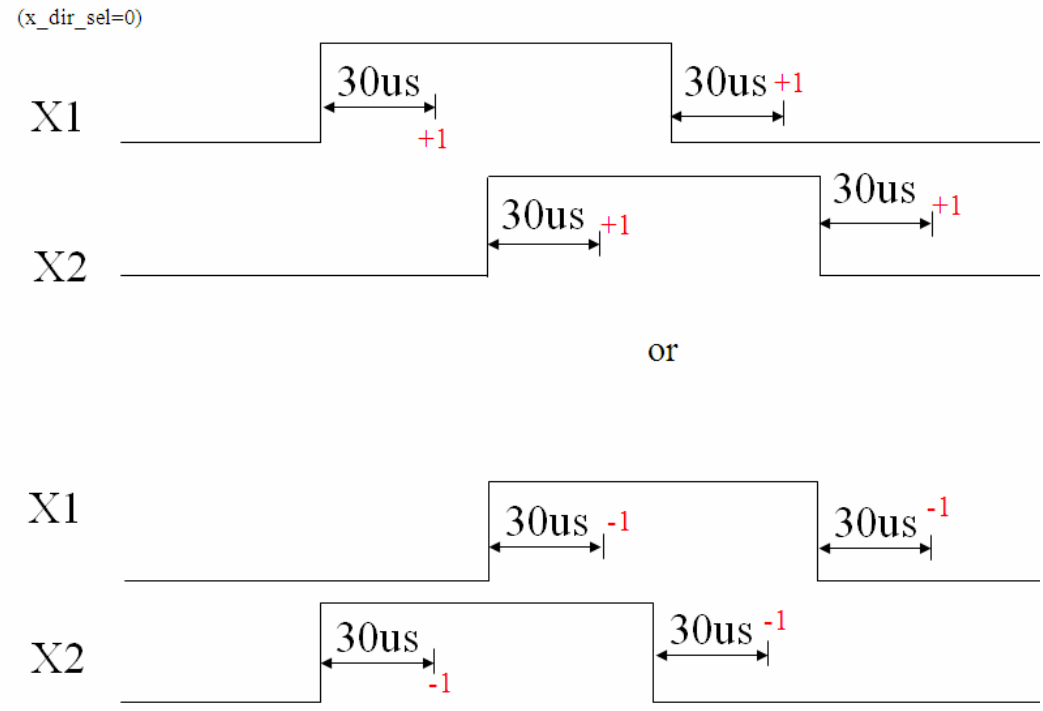


Figure 13-1 The behavior of the counter of x(y)-axis when x(y)_dir_sel=0

x1	x2	x_dir_sel	x_count
0->1	0	0	+1
0->1	1	0	-1
1->0	0	0	-1
1->0	1	0	+1
0	0->1	0	-1
1	0->1	0	+1
0	1->0	0	+1
1	1->0	0	-1

Table 13-1 The behavior of the counter of x(y)-axis

13.2 Z-axis

13.2.1 Overview

The z-axis supports two modes, standard-z and rambo-z. The standard-z is similar to xy-axis while rambo-z is specifically designed for Rogitech

13.2.2 Function Descriptions

13.2.2.1 standard-z

The standard-z uses two inputs (say z1 and z2). The z1 and z2 can be debounced. There are two modes of standard-z, z/2 mode and z/4 modes. The criterion of increasing/decreasing the counter of z/2 mode is shown in Figure 13-2 and z/4 mode in Figure 13-3 2. The figures show the case of z_dir_sel=0. If z_dir_sel=1, the value 1 and -1 are swapped. Hardware stores the counted value in a counter(z_count) and software can determine the rotation of the wheel.

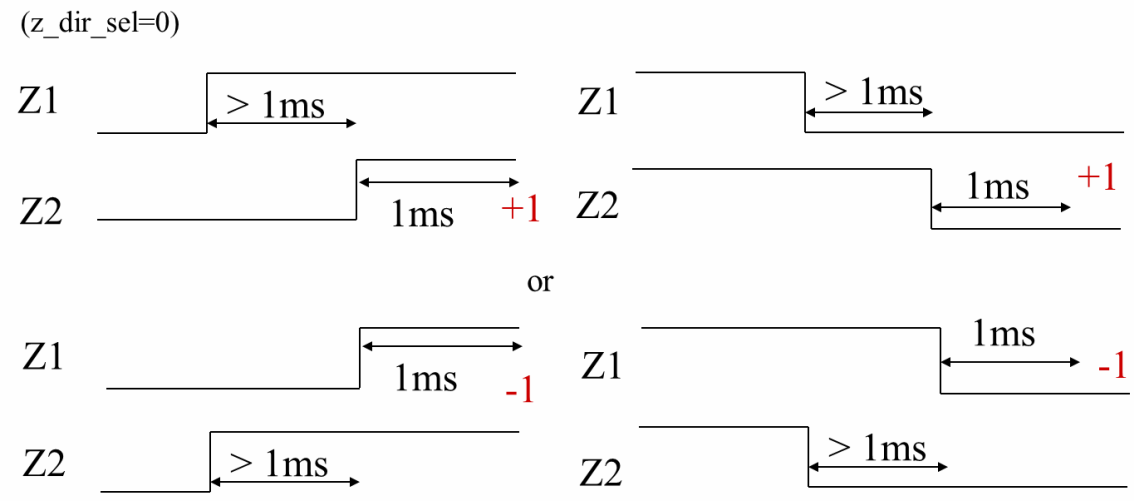


Figure 13-2 The behavior of the counter of z-axis in z/2 mode

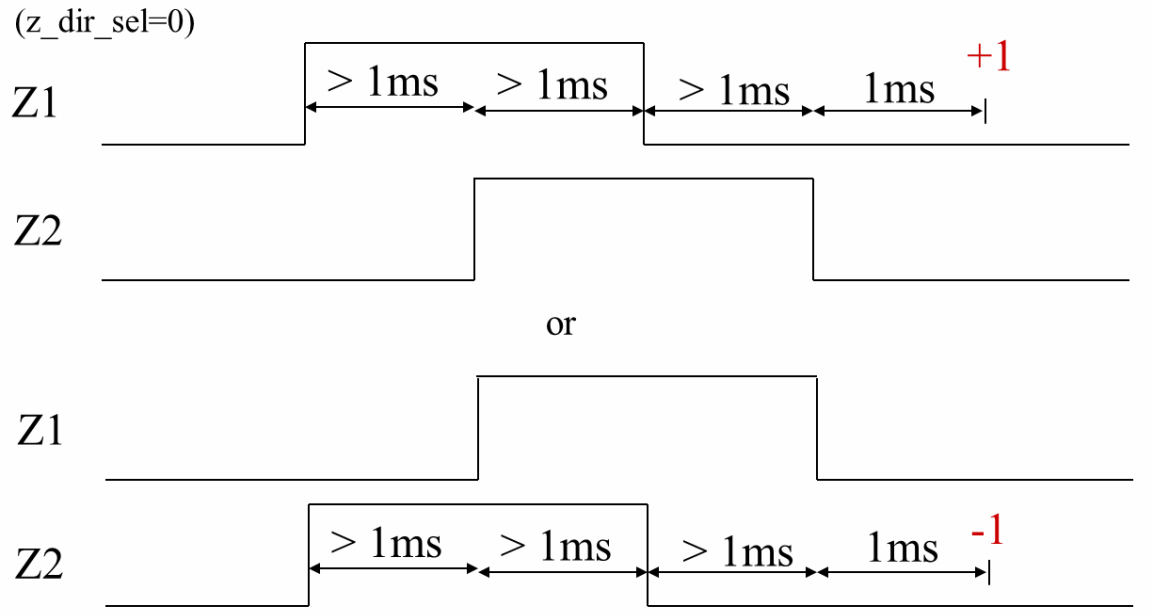


Figure 13-3 The behavior of the counter of z-axis in z/4 mode

13.2.2.1 rambo-z

The rambo-z uses one output and one inout ports. The output port is used for led light. The inout port is used for clock source to the sensor as output and sample the data from the sensor as input, shown in Figure 13-4 4. The rambo-z use wheel algorithm from Logitech to detect the rotation of the wheel, shown in Figure 13-5 5. There are four sampling rate modes which are automatically switched by hardware: walk, run, idle and sleep. In walk mode, the rate is configurable between 125us~4ms. In run mode, the rate is 125us. In idle mode, the rate is 4ms. In sleep mode, the rate is configurable between 4ms~64ms. There are examples of wheel algorithm in Figure 13-6 6.

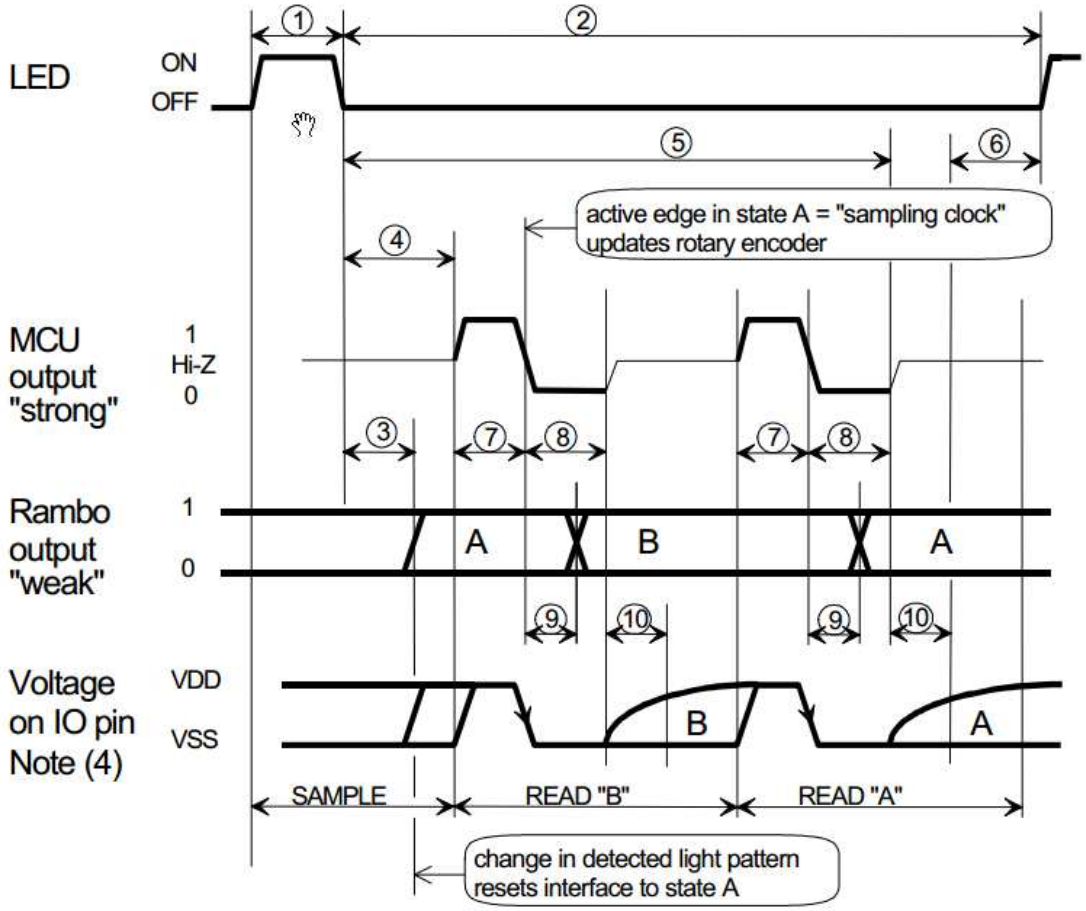


Figure 13-4 The waveform of rambo-z

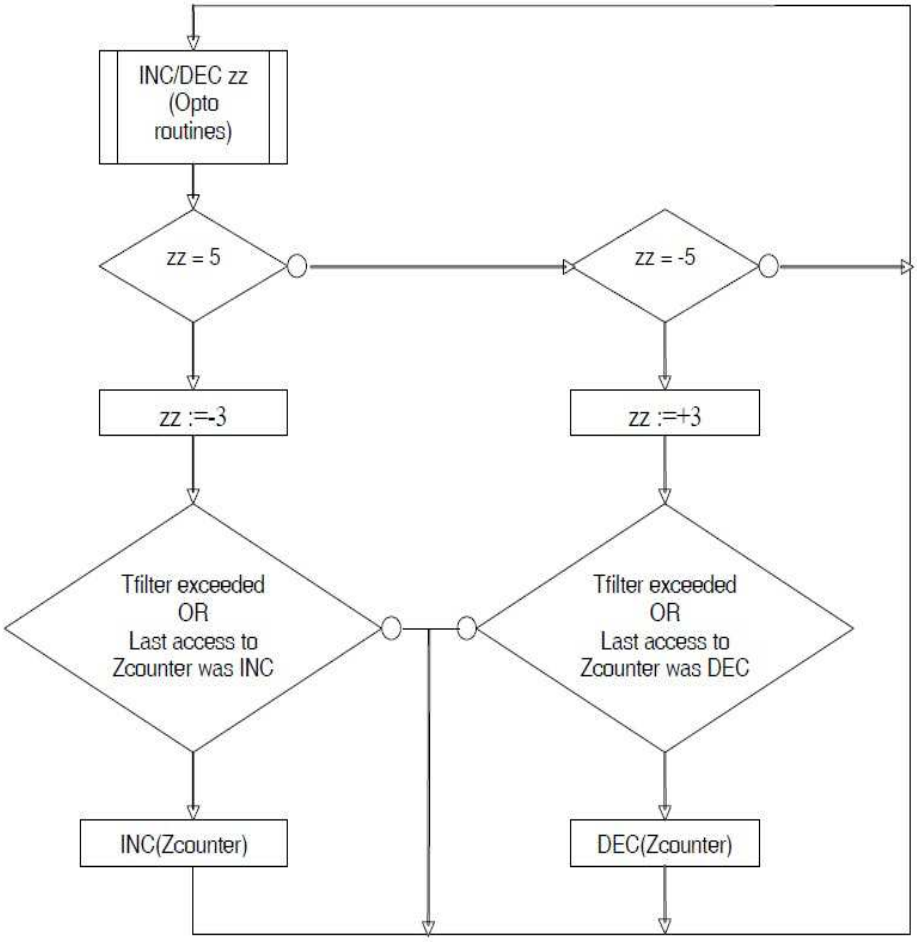


Figure 13-5 The wheel algorithm of rambo-z

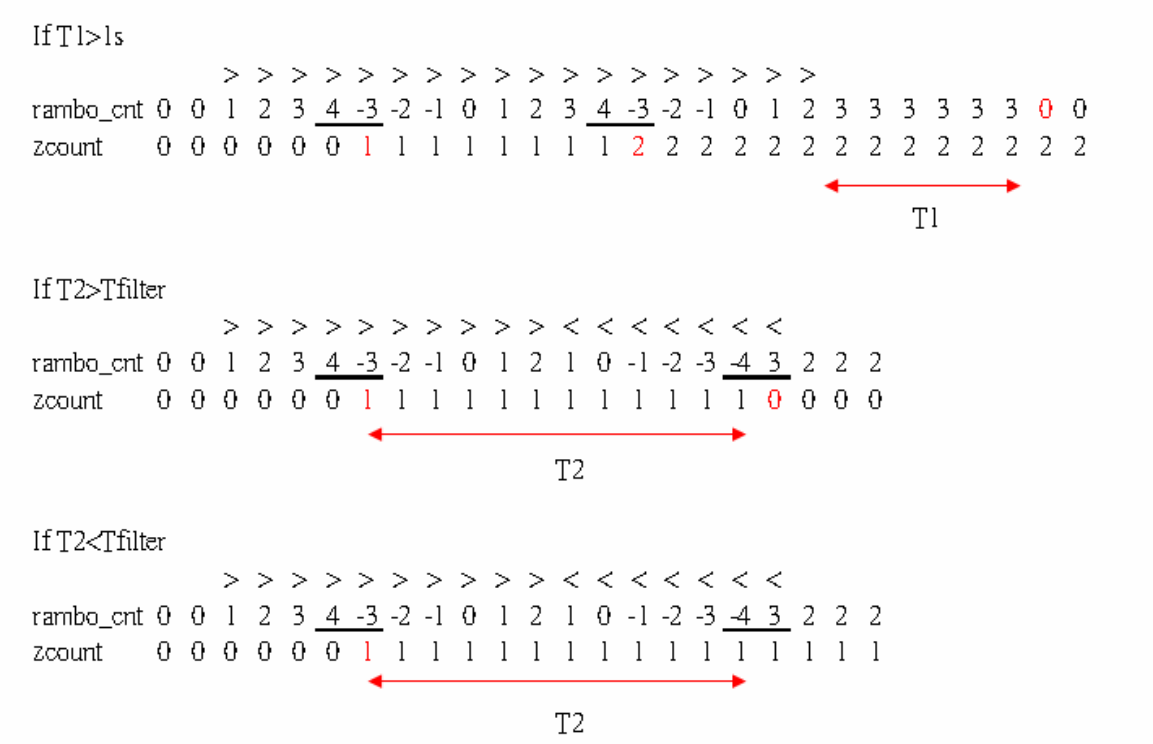


Figure 13-6 The examples of wheel algotirhm

13.3 Reading Counters

Due to the dynamic counters in mouse. Software should use the latch mechanism to read the counters for all axis and modes. For x,y and standard-z, when the x(yz)_latch_strobe is set, the x(yz)_count is latched to x(yz)_count_latch and the x(yz)_count is cleared. For rambo-z, when the rambo_latch_strobe is set, the rambo_cnt is latched to rambo_cnt_latch and the rambo_cnt is not influenced.

13.4 SFR table

mouse_ctrl (0x227000)

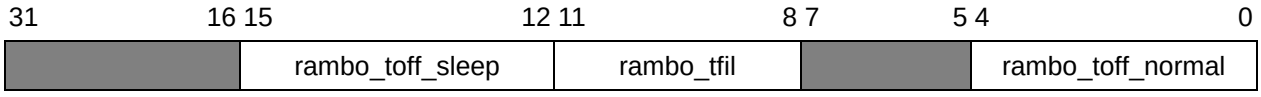
The enable and direction select of mouse.

31 8	7	6	5	4	3	2	1	0
	rambo_enable	deb_enable	z_enable	xy_enable	z_sel	z_dir_sel	y_dir_sel	x_dir_sel

Name	Bit	Write/Read	Reset value	Description
x_dir_sel	[0]	W/R	0	x direction select 1'b0: x1 lead x2 => counter+1 1'b1: x2 lead x1 => counter+1
y_dir_sel	[1]	W/R	0	y direction select 1'b0: y1 lead y2 => counter+1 1'b1: y2 lead y1 => counter+1
z_dir_sel	[2]	W/R	0	standard-z direction select 1'b0: z1 lead z2 => counter+1 1'b1: z2 lead z1 => counter+1
z_sel	[3]	W/R	0	standard-z mode select 1'b0, z/2 mode 1'b1, z/4 mode
xy_enable	[4]	W/R	0	enable xy axis
z_enable	[5]	W/R	0	standard-z enable
deb_enable	[6]	W/R	0	debounce enable for standard-z mode. When 1'b1, the z1 and z2 is debounced by 1ms (the transition in 1ms is filtered)
Rambo_enable	[7]	W/R	0	rambo-z enable. Note that the rambo_enable and z_enable can't be set 1'b1 simultaneously.

rambo_setting (0x227004)

The settings of rambo-z.



Name	Bit	Write/Read	Reset value	Description
rambo_toff_normal	[4:0]	W/R	5'd3	Led Toff period when normal work. The period is (rambo_toff_normal+1)*125us (@128k clk). Default is 500us.
rambo_tfil	[11:8]	W/R	4'd8	Led Tfil period. The direction change of z_count during Tfil will be filtered. The period is (rambo_tfil*8ms) (@128k clk). Default is 64ms.
rambo_toff_sleep	[15:12]	W/R	4'd15	Led Toff period when sleep & rambo idle. The period is (rambo_toff_sleep+1)*4ms (@128k clk). Default is 64ms.

clk_en (0x227008)

The clock enable for power saving.

31	2	1	0
		slow_clk_en	core_clk_en

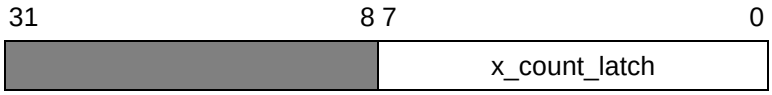
Name	Bit	Write/Read	Reset value	Description
core_clk_en	[0]	W/R	0	enable of core_clk. Should be 1’b1 when using mouse.
slow_clk_en	[1]	W/R	0	enable of slow_clk. Should be 1’b1 when using z-axis (standard-z and rambo-z).

x_latch_strobe (0x22700C)

31	1	0
		x_latch_strobe

Name	Bit	Write/Read	Reset value	Description
x_latch_strobe	[0]	WO	N/A	Write 1'b1 to latch x_count to x_count_latch & clear x_count

x_count_latch (0x227014)

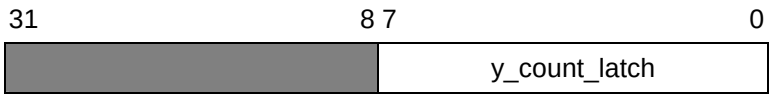


Name	Bit	Write/Read	Reset value	Description
x_count_latch	[7:0]	RO	N/A	After x_latch_strobe is set, this value is valid for software to read.

y_latch_strobe (0x227018)

Name	Bit	Write/Read	Reset value	Description
y_latch_strobe	[0]	WO	N/A	Write 1'b1 to latch y_count to y_count_latch & clear y_count

y_count_latch (0x227020)

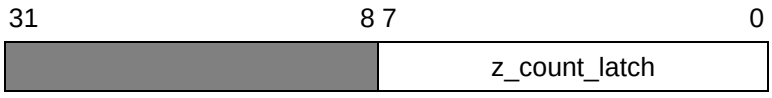


Name	Bit	Write/Read	Reset value	Description
y_count_latch	[7:0]	RO	N/A	After y_latch_strobe is set, this value is valid for software to read.

z_latch_strobe (0x227024)

Name	Bit	Write/Read	Reset value	Description
z_latch_strobe	[0]	WO	N/A	Write 1'b1 to latch z_count to z_count_latch & clear z_count

z_count_latch (0x227028)

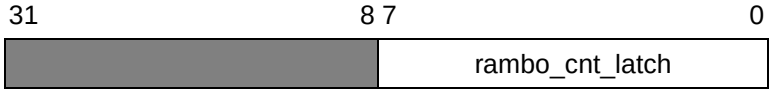


Name	Bit	Write/Read	Reset value	Description
z_count_latch	[7:0]	RO	N/A	After z_latch_strobe is set, this value is valid for software to read.

rambo_latch_strobe (0x22702C)

Name	Bit	Write/Read	Reset value	Description
z_latch_strobe	[0]	WO	N/A	Write 1'b1 to latch rambo_cnt to rambo_cnt_latch.

rambo_cnt_latch (0x227030)



Name	Bit	Write/Read	Reset value	Description
rambo_cnt_latch	[3:0]	RO	N/A	After rambo_latch_strobe is set, this value is valid for software to read. The value is between -4 to 4 according to the Rambo spec.

14 Smart card

14.1 Overview

The Smart card control is based on ISO/IEC 7816-3 standard. The following features are supported:

- ISO-7816-3 T=0, T=1 compliant
- Separate Tx/Rx 8 bytes FIFO buffers for data
- Programmable clock frequency for transmission
- Programmable rx buffer trigger level
- Programmble guard time period (11 ETU ~ 266 ETU)
- One 24-bit and Two 8-bit counters
- Support auto inverse convention
- Support to stop clock and to stop level function
- Support tx & rx parity error retry with limit number of times
- Support active, warm-reset, and deactive sequence

14.2 Block Diagrams

The basic function and interface of SC controller is shown in Figure 14-1 The Interface between system and SC Controller is APB Bus. The Interface between SC Controller and Smart Card Device is I/O Voltage Controller, which convert the voltage between the two devices and support voltage to Smart Card Device through SC_VCC pin.

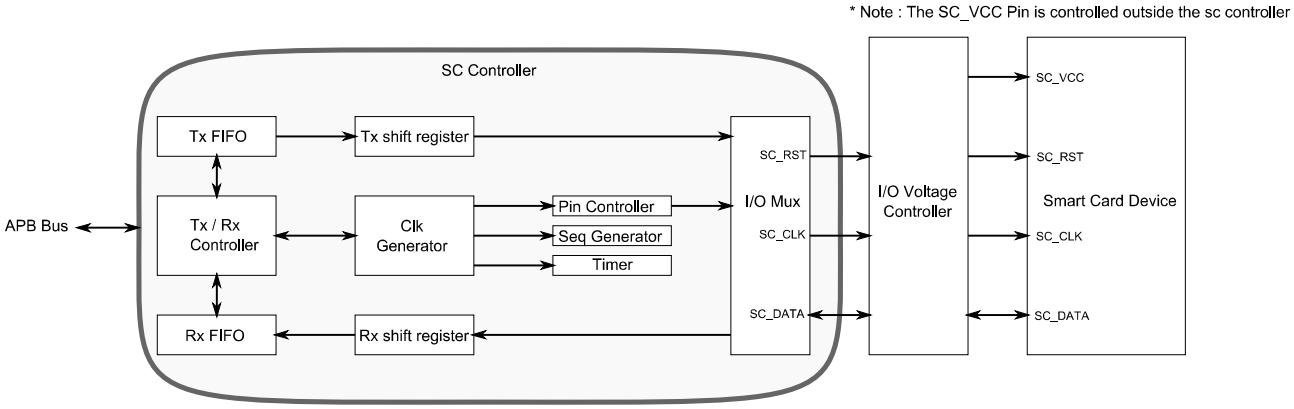


Figure 14-1 The architecture of Smart Card

Figure 14-2 2 shows the clock tick domain of SC Controller. There are two units: SC_CLK and ETU. The sequence generator uses SC_CLK as unit whereas the TX/RX ctrl & timers use ETU.

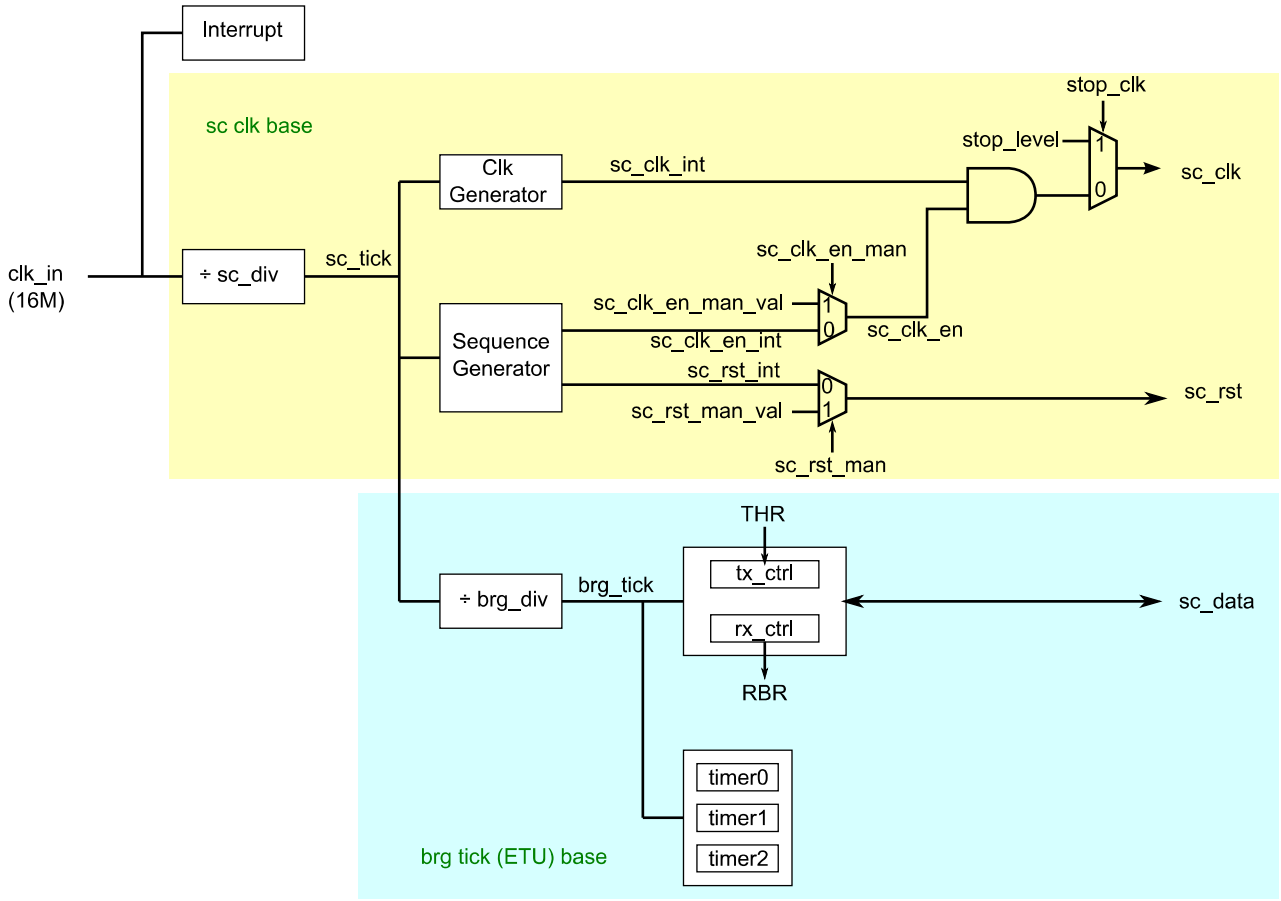


Figure 14-2 The clock domain of Smart Card

Figure 14-3 3shows the structure of I/O Voltage Controller. The VCC pin is generated by three power switches, which are controlled by 3 different GPIOs of our chip. Due to three different voltages in specification, there is a level shifter between the chip and Smart Card Device.

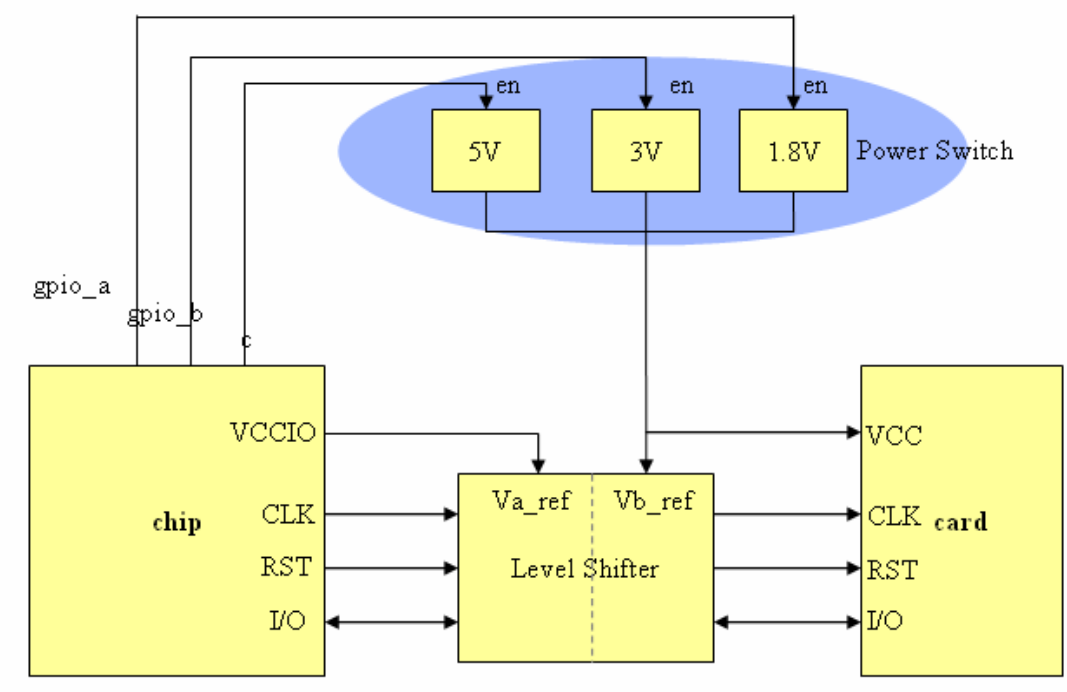


Figure 14-3 The structure of Voltage Controller

14.3 Function Descriptions

14.3.1 Character Format

The smart card interface acts as half-duplex asynchronous port. The character format is shown as Figure 14-4 . The period of “Pause” can be programmed by [tx_delay]. In T=0 mode, the “Pause” is (2 + [tx_delay]) ETUs. In T=1 mode, the “Pause” is (1 + [tx_delay]) ETUs.

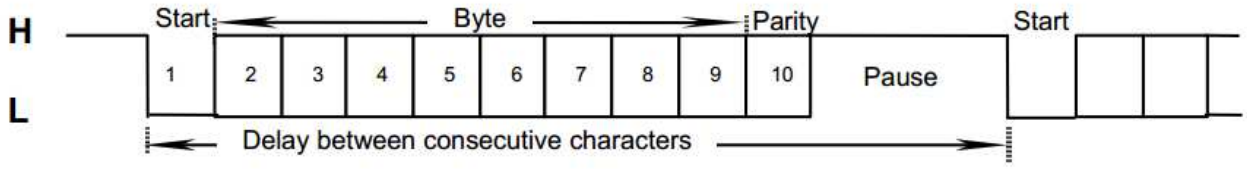


Figure 14-4 The character format

14.3.2 Sequences

The active sequence is used in the beginning of communication. Write [active_en] can trigger the sequence. The sequence is shown in Figure 14-5 . The parameter T1[seq_t1] is programmable.

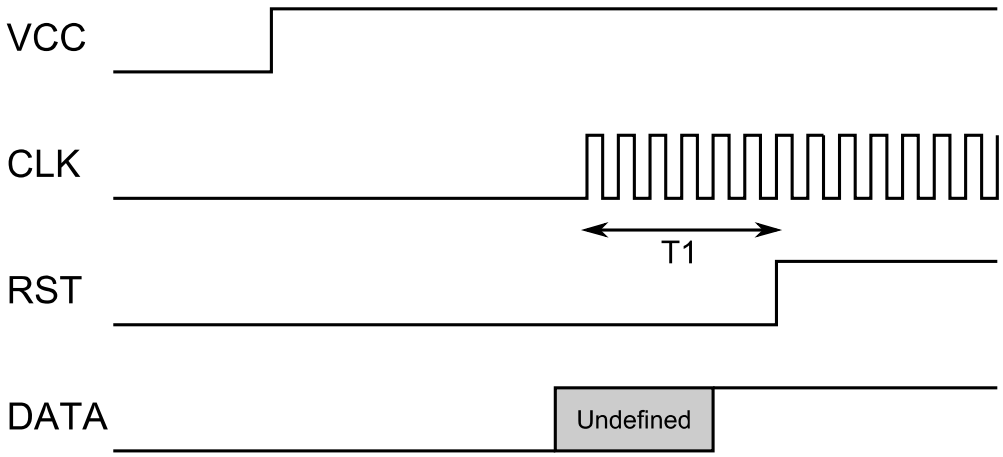


Figure 14-5 The waveform of active sequence

The warm-reset sequence is used during the communication at any time if needed. Write [warm_rst_en] can trigger the sequence. The sequence is shown in Figure 14-6 . The parameter T4[seq_t4] is programmable.

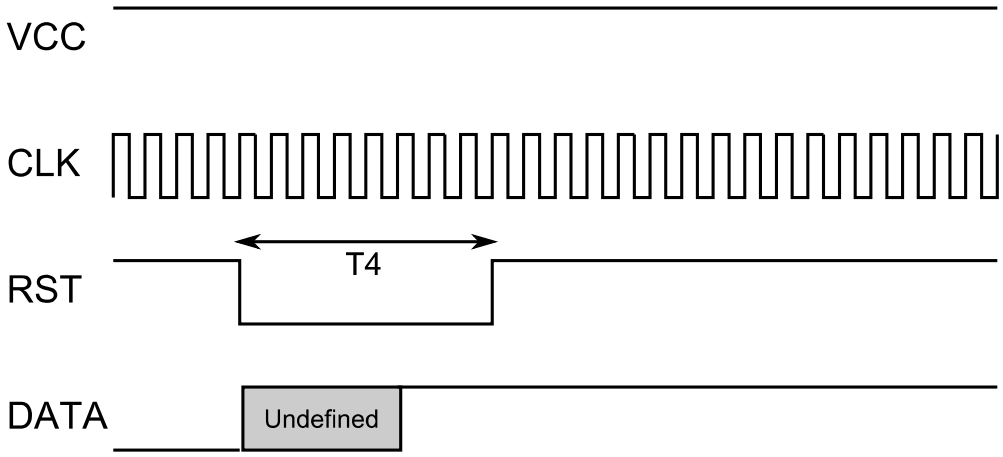


Figure 14-6 The waveform of warm-reset sequence

The deactive sequence is used in the end of communication. Write [deactive_en] can trigger the sequence. The sequence is shown in Figure 14-7 . The parameter T7[seq_t7] is programmable.

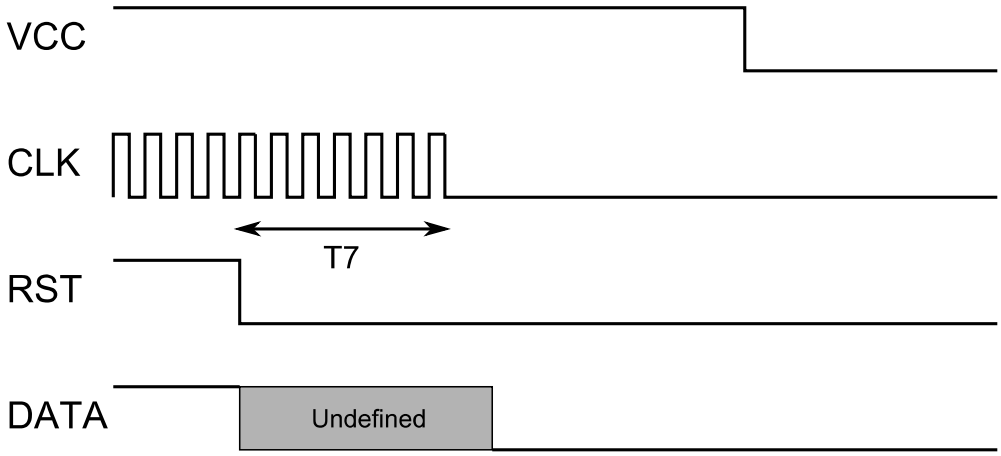


Figure 14-7 The waveform of deatctive sequence

14.3.3 Initial Character (TS)

After active sequence or warm-reset sequence, the smart card device should send Answer to Request (ATR), shown in Figure 14-8 , which composes of 1 TS character and then up to 32 characters. The TS character determines the format of character. At moment 1~10 (refer to Figure 14-10), if the TS is

LHHL LLL LLH, it sets up the inverse convention: state L encodes 1 and moment 2 conveys MSB. The conveyed byte is ‘3F’.

LHHL HHH LLH, it sets up the direct convention: state H encodes 1 and moment 2 conveys LSB. The conveyed byte is ‘3B’.

The character is shown in Figure 14-9 .

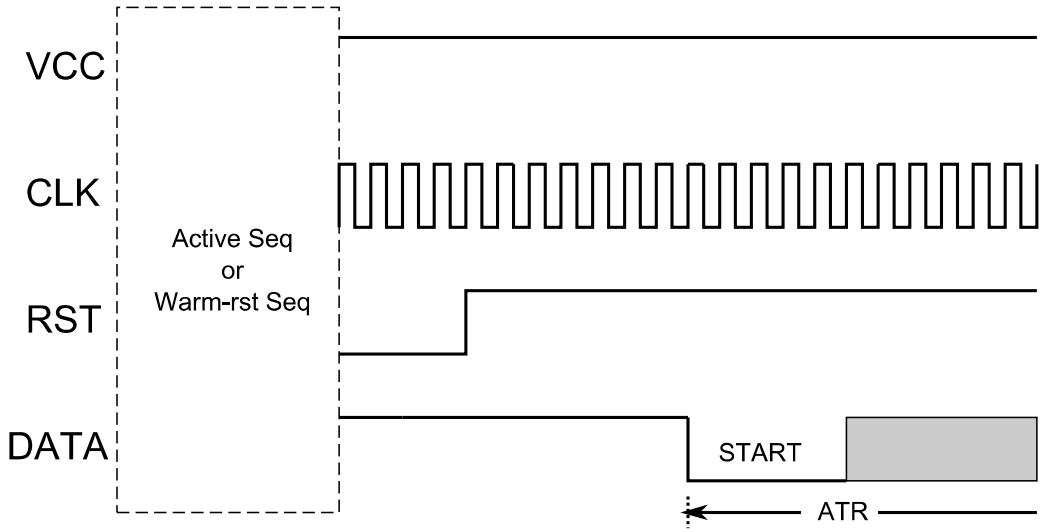


Figure 14-8 The timing of ATR

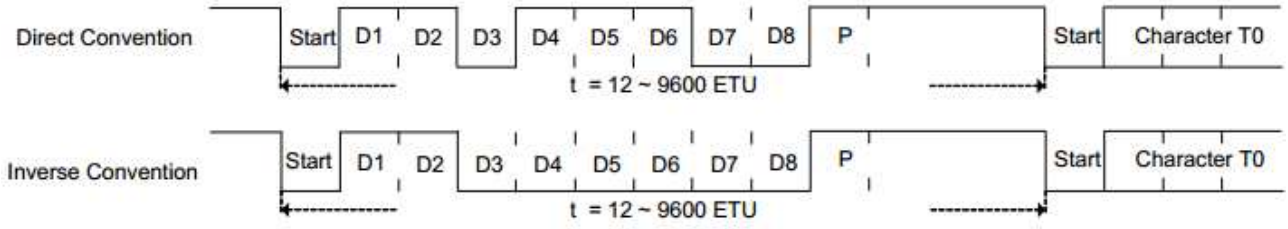


Figure 14-9 The format of TS character

14.3.4 Nack signal & Retransmission

In T=0 mode, if the receiver receives parity error, it should drive the DATA pin low for 1 or 2 bits [rx_nack_sel]. The transmitter should retransmit the character. Write [tx_retry_en] & [rx_retry_en] to enable the retransmission function. Set [tx_retry_num] & [rx_retry_num] to limit the number of retransmission. The retransmission mechanism is shown in Figure 14-10 .

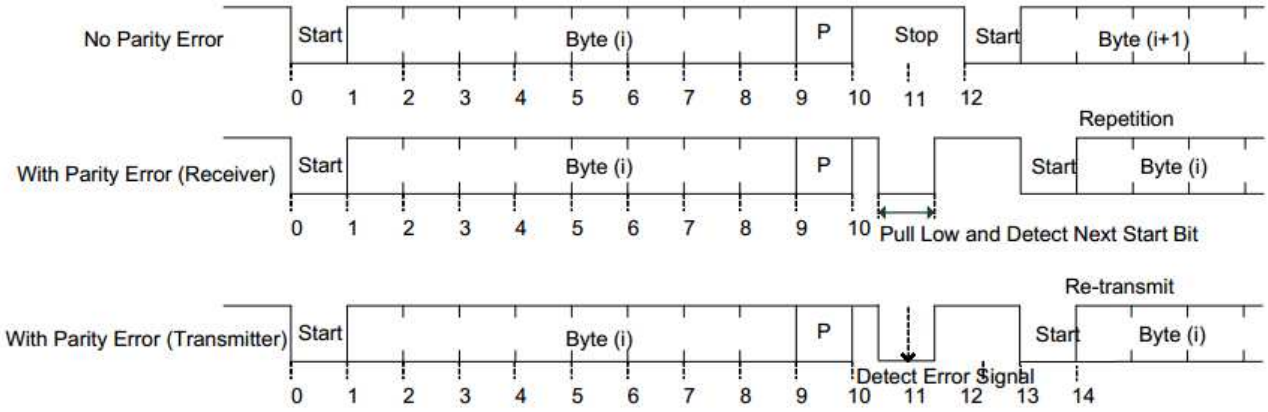


Figure 14-10 The waveform of transmission in T=0 mode

14.3.5 Timers

There are one 24-bit timer and two 8-bit timers. The unit of timer is ETU. Once the [tmr*X*_en] is set, the counter is reloaded with [tmr*X*_tv] and starts to count down. If the counter counts down to 0, the timer is disabled and [TMR*X*_IS] is asserted. There are 4 modes [tmr*X*_mode] of timer:

Mode 0

Start : Start to count when [tmr*X*_en] enabled

End : When the down counter is “0”, hardware will set [TMR*X*_IS] and clear [tmr*X*_en] automatically.

Mode 1

Start : Start to count when the first START bit (reception or transmission) detected after [tmr*X*_en] set to “1”.

End : When the down counter is “0”, hardware will set [TMR*X*_IS] and clear [tmr*X*_en] automatically.

Mode 2

Start : Start to count when the first START detected bit (reception) after [tmr*X*_en] set to “1”.

End : When the down counter is “0”, hardware will set [TMR*X*_IS] and clear [tmr*X*_en] automatically.

Mode 3

Start : Start to count when SC_RST deasserts after [tmr*X*_en] set to“1”

End : (a) When the down counter is “0” before ATR received, hardware will set [TMR*X*_IS] and clear [tmr*X*_en] automatically. (b) When ATR received and down counter is not “0”, hardware will clear [tmr*X*_en]

automatically.

14.3.6 Interrupts

There are two levels of interrupts. The first level is read and cleared directly by software. There are independent enables [IER] of first-level interrupt. The second level needs to be read and cleared via another sfr. The interrupts of second level in the same group share common interrupt enable[*1]. All interrupt status [*XXX*_IS] can still be read and cleared when [IER] is 0. The [IER] is used to determine whether to issue interrupt to MCU. The interrupt structure is shown in Figure 14-11 (For AB1520S, there is some variation).

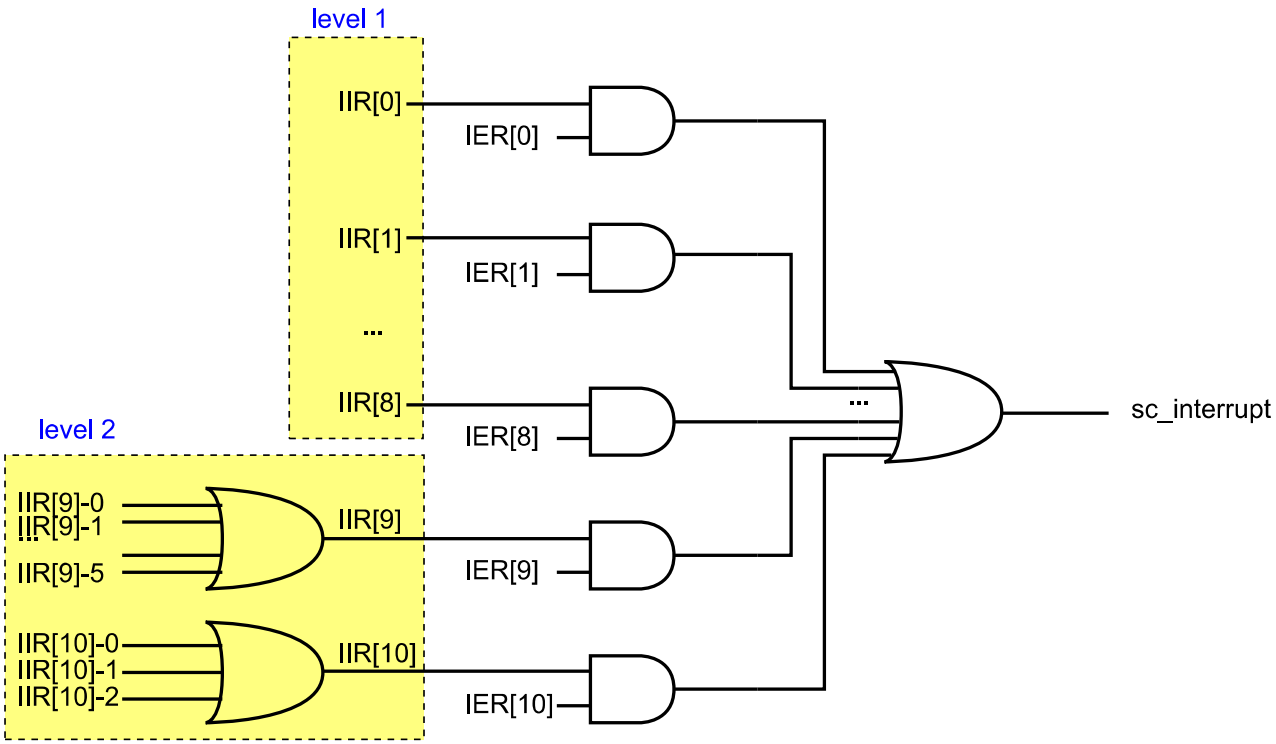


Figure 14-11 The structure of interrupt of Smart Card

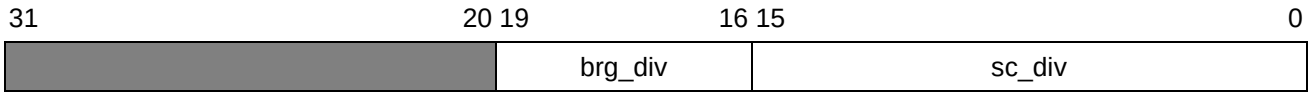
14.4 SFR table

Transmit Holding Register / Receive Buffer Register (THR/RBR

0x20C000)

Name	Bit	Write/Read	Reset value	Description
THR/RBR	[7:0]	R/W	0	W: Write sent data to fifo (fifo has depth 8) R: Read received data from fifo (fifo has depth 8)

clk_divisor (0x20C004)



Name	Bit	Write/Read	Reset value	Description
brg_div	[15:0]	R/W	0x174	The divisor of bit period(etu). The period of 1 etu is **F/D=brg_div/fsc**
sc_div	[19:16]	R/W	0x4	The divisor of core_clk to smart card. The freq of sc_clk pin is **fsc=fclk/sc_div,** where fclk is the freq of core_clk. Note that the minimum valid value of sc_div is 2.

Line Control Register (LCR - 0x20C008)

The LCR determines the format of character.

	sp	eps	

Name	Bit	Write/Read	Reset value	Description
eps	[4]	R/W	1	Format of parity bit. {sp,eps} = 2'b00 : odd parity 2'b01 : even parity 2'b10 : always 0 2'b11 : always 1
sp	[5]	R/W	0	

Interrupt Enable Register (IER - 0x20C00C)

The enable of interrupts.

6	5	4	3	2	1	0
IE_TS_ERR	IE_RXTO	IE_TMR2	IE_TMR1	IE_TMR0	IE_TBE	IE_RDY

31 ... 11	10	9	8 ... 7
	IE_SEQ	IE_RERR	

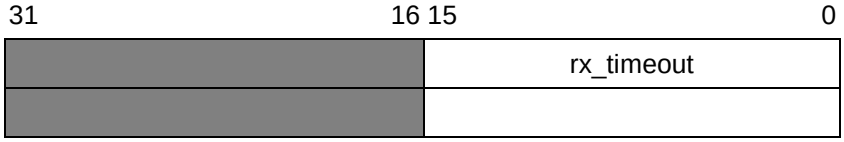
Name	Bit	Write/Read	Reset value	Description
IE_RDY	[0]	R/W	0	Interrupt enable of RDY_IS
IE_TBE	[1]	R/W	0	Interrupt enable of TBE_IS
IE_TMR0	[2]	R/W	0	Interrupt enable of TMR0_IS
IE_TMR1	[3]	R/W	0	Interrupt enable of TMR1_IS
IE_TMR2	[4]	R/W	0	Interrupt enable of TMR2_IS

IE_RXTO	[5]	R/W	0	Interrupt enable of RXTO_IS
IE_TS_ERR	[6]	R/W	0	Interrupt enable of TS_ERR_IS
IE_RERR	[9]	R/W	0	Interrupt enable of RERR_IS
IE_SEQ	[10]	R/W	0	Interrupt enable of SEQ_IS

tx_delay (0x20C010)

Name	Bit	Write/Read	Reset value	Description
tx_delay	[7:0]	R/W	0	Tx additional delay time between characters. It determines the guard time (GT) in specification. In T=0(tx_retry_en=0), pause is **(2+tx_delay) etu**; in T=1(tx_retry_en=1), the pause period is **(1+tx_delay) etu.**

rx_timeout (0x20C014)



Name	Bit	Write/Read	Reset value	Description
rx_timeout	[15:0]	R/W	0x1000	Rx timeout value. If (1) rx buffer is not empty and (2) there is no write/read operation of rx fifo and sc ctrl is NOT receiving data for a certain time, the rx timeout interrupt status asserts. The timeout time is **rx_timeout etu.**

FIFO Control Register (FCR - 0x20C018)

The FCR controls the setting of fifo.

Name	Bit	Write/Read	Reset value	Description
rxf_trigger_level	[1:0]	R/W	2	When FIFO is enabled, the rx data is ready to interrupt status when the number of data in rx fifo reach the corresponding value 2'd0 : 1 bytes 2'd1 : 2 bytes 2'd2 : 4 bytes 2'd3 : 6 bytes
fifo_en	[8]	R/W	1	Fifo enabled. If it's set 1'b0, the fifo is disabled and the THR/RBR can't be continuously written/received.

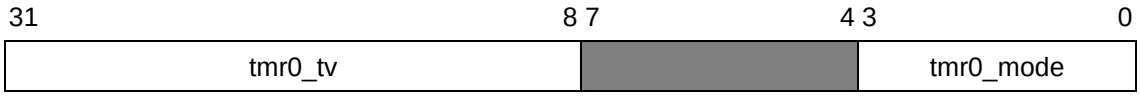
FIFO_reset (0x20C01C)

The Fifo_reset is used to reset the fifos.

Name	Bit	Write/Read	Reset value	Description
rxf_rst	[1]	Write 1 only & Auto clear	0	Rx fifo reset. It can only be set 1’b1 and the rx fifo is reset. After reset is done, it's automatically cleared. Also, it's automatically set 1’b1 when sc_enable is set. Software need to check it's 1'b0 before normal use.
txf_rst	[0]	Write 1 only & Auto clear	0	Tx fifo reset. It can only be set 1’b1 and the tx fifo is reset. After reset is done, it's automatically cleared. Also, it's automatically set 1’b1 when sc_enable is set. Software need to check it's 1'b0 before normal use.

tmr0 (0x20C020)

The tmr0 controls the Timer0.

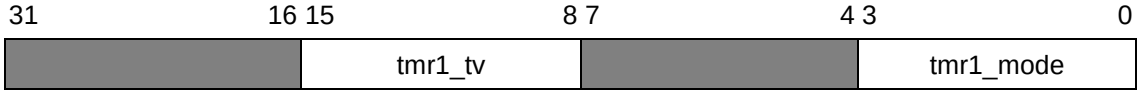


Name	Bit	Write/Read	Reset value	Description
tmr0_mode	[3:0]	R/W	0	Timer0 mode. Determine the behavior of timer0. 4'd0 : Start to count when tmr0_en is set. When timeout (counter counts down to 0), hardware will set TMR0_IS and clear tmr0_en automatically. 4'd1 : Start to count when the first START bit (reception or transmission) detected after tmr0_en is set. When timeout (counter counts

				down to 0), hardware will set TMR0_IS and clear tmr0_en automatically. 4'd2 : Start to count when the first START detected bit (reception) after tmr0_en is set. When timeout (counter counts down to 0) has happened, hardware will set TMR0_IS and clear tmr0_en automatically. 4'd3 : Start to count when SC_RST de-assertion after tmr0_en is set. When the first START bit of ATR response is received before timeout, the tmr0_en is cleared automatically. Otherwise, when timeout, hardware will set TMR0_IS and clear tmr0_en automatically. others : reserved.
tmr0_tv	[31:8]	R/W	0	Timer0 timeout value. When tmr0_en is set, this value is loaded into counter. The timeout time is **tmr0_tv etu.**

tmr1 (0x20C024)

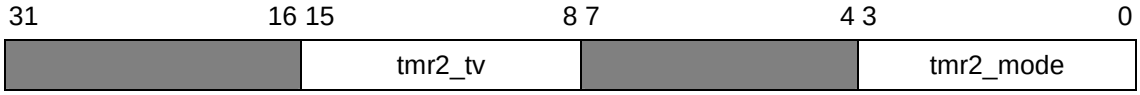
The tmr1 controls the Timer1.



Name	Bit	Write/Read	Reset value	Description
tmr1_mode	[3:0]	R/W	0	Timer1 mode. Determine the behavior of timer1. See tmr0_mode.
tmr1_tv	[15:8]	R/W	0	Timer1 timeout value. See tmr0_tv.

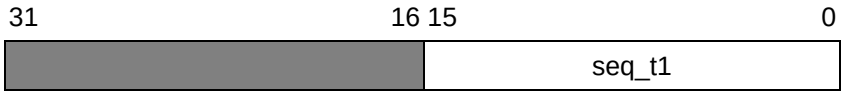
tmr2 (0x20C028)

The tmr2 controls the Timer2.



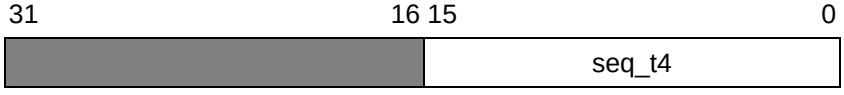
Name	Bit	Write/Read	Reset value	Description
tmr2_mode	[3:0]	R/W	0	Timer2 mode. Determine the behavior of timer2. See tmr0_mode.
tmr2_tv	[15:8]	R/W	0	Timer2 timeout value. See tmr0_tv.

seq_t1 (0x20C030)



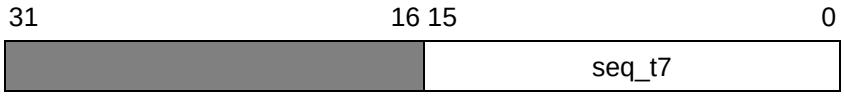
Name	Bit	Write/Read	Reset value	Description
seq_t1	[15:0]	R/W	0x300	Used for active sequence. The seq_t1 determines the period between sc_clk start and sc_rst de-assert. The unit is sc clk (1/fsc).

seq_t4 (0x20C034)



Name	Bit	Write/Read	Reset value	Description
seq_t4	[15:0]	R/W	0x300	Used for warm_rst sequence. The seq_t4 determines the period between sc_rst assert and sc_rst de-assert. The unit is sc clk (1/fsc).

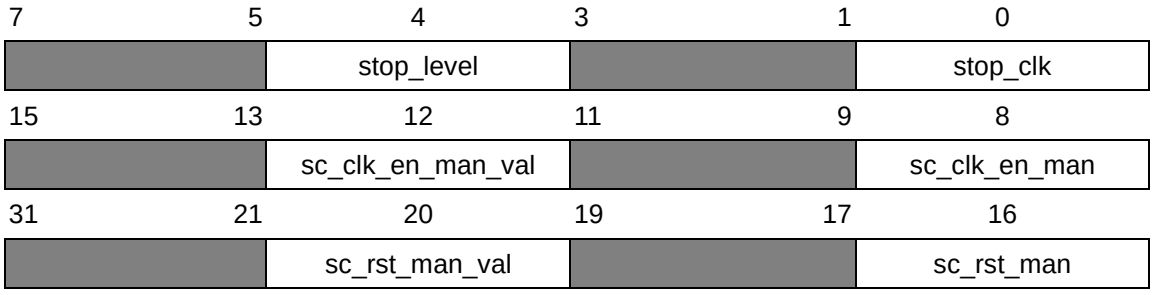
seq_t7 (0x20C038)



Name	Bit	Write/Read	Reset value	Description
seq_t7	[15:0]	R/W	0x300	Used for deactive sequence. The seq_t7 determines the period between sc_rst assert and sc_clk stop. The unit is sc clk (1/fsc).

pin_ctrl (0x20C03C)

The pin_ctrl controls the behavior of pins sc_clk and sc_rst.

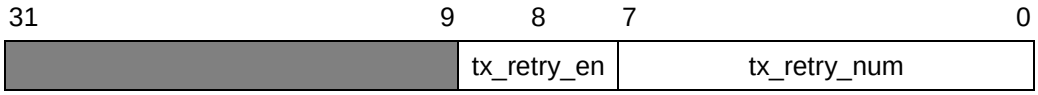


Name	Bit	Write/Read	Reset value	Description
stop_clk	[0]	R/W	0	Set 1'b1 to stop sc_clk. 1'b0 to restore sc_clk

stop_level	[4]	R/W	0	Determine the sc_clk level when stop_clk is 1b'1. 1'b1: sc_clk=1, 1'b0: sc_clk=0
sc_clk_en_man	[8]	R/W	0	Manually set sc_clk_en value by sc_clk_en_man_val
sc_clk_en_man_val	[12]	R/W	0	Determine sc_clk_en value when sc_clk_en_man is 1'b1. 1'b1: sc_clk_en=1, 1'b0: sc_clk_en=0
sc_rst_man	[16]	R/W	0	Manually set sc_rst value by sc_rst_man_val
sc_rst_man_val	[20]	R/W	0	Determine sc_rst value when sc_rst_man is 1'b1. 1'b1: sc_rst=1, 1'b0: sc_rst=0

tx_retry (0x20C040)

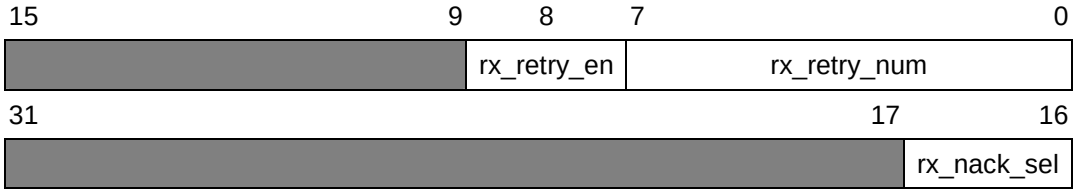
The tx_retry controls the retry function of tx.



Name	Bit	Write/Read	Reset value	Description
tx_retry_num	[7:0]	R/W	0x4	Indicates the maximum number of times of tx retry that are allowed when parity error has occurred.
tx_retry_en	[8]	R/W	1	Enables tx retry function when parity error has occurred. **Set 1'b1 when T=0.**

rx_retry (0x20C044)

The rx_retry controls the retry function and nack function of rx.



Name	Bit	Write/Read	Reset value	Description
rx_retry_num	[7:0]	R/W	0x4	Indicates the maximum number of times of rx retry that are allowed when parity error has occurred.
rx_retry_en	[8]	R/W	1	Enables rx retry function when parity error has occurred. **Set 1'b1 when T=0.**
rx_nack_sel	[16]	R/W	0	Determine the number of etu of nack period. 1'b0: 1etu, 1'b1: 2 etu

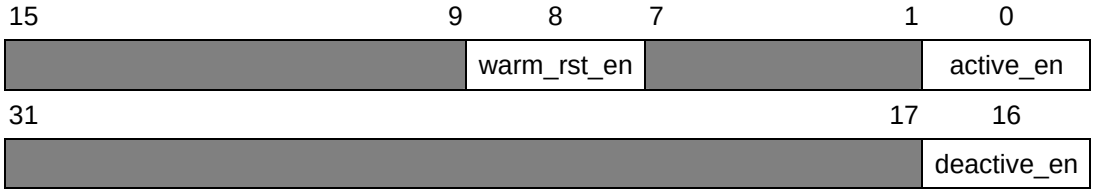
sc_enable (0x20C04C)

The main enable of sc ctrl.

Name	Bit	Write/Read	Reset value	Description
sc_enable	[0]	R/W	0	sc ctrl enable. Set 1’b1 to enable the sc ctrl.

seq_en (0x20C050)

The seq_en is used to trigger the sequences. Note that when any field is set 1'b1, the tx/rx fifo is cleared.



Name	Bit	Write/Read	Reset value	Description
active_en	[0]	Write 1 only & Auto clear	0	Enables activation sequence. When the activation sequence completed, this bit will be auto cleared and the ACT_IS [SEQ_IS] will be set.
warm_rst_en	[8]	Write 1 only & Auto clear	0	Enables warm_rst sequence. When the warm_rst sequence completed, this bit will be auto cleared and the WARM_IS [SEQ_IS] will be set.
deactive_en	[16]	Write 1 only & Auto clear	0	Enables deactivation sequence. When the deactivation sequence completed, this bit will be auto cleared and the DEACT_IS [SEQ_IS] will be set.

tmr0_en (0x20C054)

The tmr0_en is used to enable the Timer0.

Name	Bit	Write/Read	Reset value	Description
tmr0_en	[0]	See Description	0	Enables Timer0 to start counting. Software can set 1'b0 to stop it and set 1'b1 to reload and count. Also, when Timer0 count down to 0, this bit is auto cleared.

tmr1_en (0x20C058)

The tmr1_en is used to enable the Timer1.

Name	Bit	Write/Read	Reset value	Description
tmr1_en	[0]	See Description	0	Enables Timer1 to start counting. Software can set 1'b0 to stop it and set 1'b1 to reload and count. Also, when Timer1 count down to 0, this bit is auto cleared.

tmr2_en (0x20C05C)

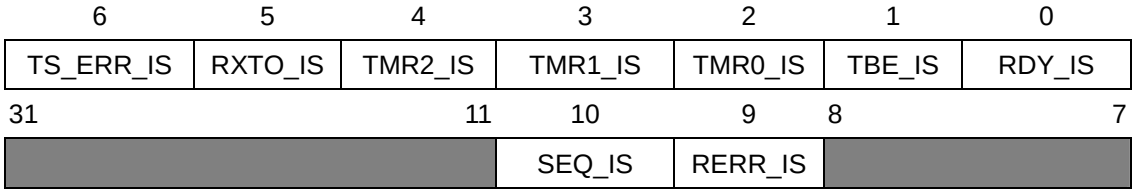
The tmr2_en is used to enable the Timer2.

Name	Bit	Write/Read	Reset value	Description
tmr2_en	[0]	See Description	0	Enables Timer2 to start counting. Software can set 1'b0 to stop it and set 1'b1 to reload and count. Also, when Timer2 count down to 0, this

				bit is auto cleared.

Interrupt Identify Regiser (IIR - 0x20C060)

The IIR collects all interrupt status.



Name	Bit	Write/Read	Reset value	Description
RDY_IS	[0]	W1C	0	Rx Data Ready. When rx fifo reach certain number of data, this bit is asserted.
TBE_IS	[1]	W1C	0	Tx Buffer Empty. When tx fifo is empty, this bit is asserted.
TMR0_IS	[2]	W1C	0	When the counter of timer0 down to 0, this bit is asserted.
TMR1_IS	[3]	W1C	0	When the counter of timer1 down to 0, this bit is asserted.
TMR2_IS	[4]	W1C	0	When the counter of timer2 down to 0, this bit is asserted.
RXTO_IS	[5]	W1C	0	Rx Timeout. If rx fifo is not empty and there is no write/read operation for a certain time, this bit is asserted.
TS_ERR_IS	[6]	W1C	0	TS Error. When received TS character is not 0x3B or 0x3F or parity error, this bit is asserted.
RERR_IS	[9]	RO	0	Rx Error. This bit is 1'b1 if any field of RERR_IIR is 1b'1. To clear this bit, clear all

				RERR_IIR field instead.
SEQ_IS	[10]	RO	0	Sequence Done. This bit is 1'b1 if any field of SEQ_IIR is 1b'1. To clear this bit, clear all SEQ_IIR field instead.

RERR_IIR (0x20C064)

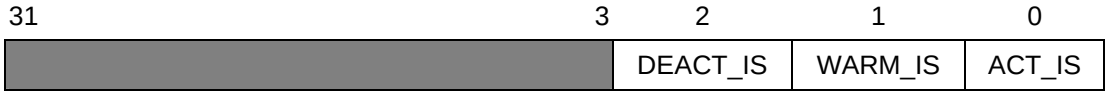
The RERR_IIR stores the interrupts status of rx error.

31 ... 6	5	4	3	2	1	0
	RX_RETRY_IS	TX_RETRY_IS		FRE_IS	PTE_IS	OE_IS

Name	Bit	Write/Read	Reset value	Description
OE_IS	[0]	W1C	0	Overrun Error. If rx fifo is full and new data is received, this bit will be asserted.
PTE_IS	[1]	W1C	0	Parity Error. If received data is parity error & the data is written into rx fifo, this bit will be asserted.
FRE_IS	[2]	W1C	0	Frame Error. If the rx stop bit is always 0, this bit will be asserted.
TX_RETRY_IS	[4]	W1C	0	If transmitted data with parity error and tx retry over certain times, this bit will be asserted
RX_RETRY_IS	[5]	W1C	0	If received data with parity error and rx retry over certain times, this bit will be asserted

SEQ_IIR (0x20C068)

The SEQ_IIR stores the interrupt status of sequence being done.



Name	Bit	Write/Read	Reset value	Description
ACT_IS	[0]	W1C	0	When active sequence is done, this bit will be asserted.
WARM_IS	[1]	W1C	0	When warm_reset sequence is done, this bit will be asserted.
DEACT_IS	[2]	W1C	0	When deactive sequence is done, this bit will be asserted.

tx_status (0x20C080)

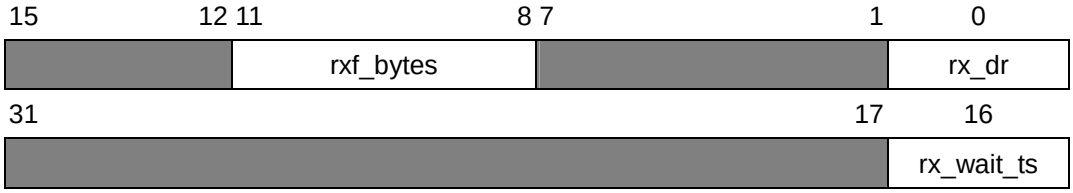
The tx_status reflects the status of tx.

Name	Bit	Write/Read	Reset value	Description
thre	[0]	RO	1	Tx fifo is empty
tsre	[1]	RO	1	Tx is IDLE. (No transmitting and tx fifo is empty)
txf_full	[2]	RO	0	Tx fifo is full

txf_bytes	[11:8]	RO	0	The number of data in tx fifo

rx_status (0x20C084)

The rx_status reflects the status of rx.



Name	Bit	Write/Read	Reset value	Description
rx_dr	[0]	RO	0	Rx FIFO has data.
rxf_bytes	[11:8]	RO	0	The numbe of data in rx fifo
rx_wait_ts	[16]	RO	0	Indicate wheter sc is waiting TS byte. When active or warm_rst sequence is trigger, it becomes 1. 1'b1: waiting 1'b0: not waiting

15 Random number generator

15.1 Overview

The Random Number Generator(RNG) is a block to generate 32-bit random numbers based on the thermal noise. The block diagram of the RNG is shown in Figure 15-1 .

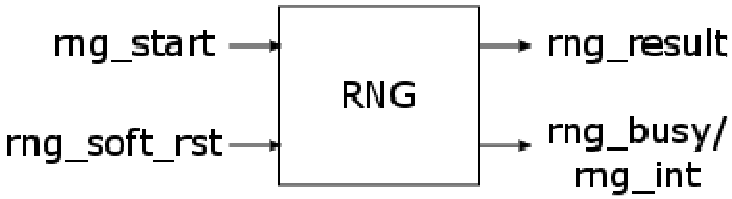


Figure 15-1 The Block Diagram of the RNG

15.2 Operation

Once the RNG is triggered, the RNG samples the noise and generates a binary sequence. The binary sequence is fed to a digital corrector to remove consecutive '1' or '0' which avoids biases toward 1 or 0 before the final result is outputted. After the 32-bit random number is generated, the RNG becomes idle.

The rng_start bit is written 1 to trigger the RNG. The random number is ready by polling the rng_busy status bit or wait rng_int interrupt. Because the random number is generated by a random process, the time needed to generate a random number is unpredictable. A timer of 10ms is recommended to implement to ensure the random number can be generated within a certain time. If time-out occurs, turn off(writing 1 to rng_soft_rst) the rng and restart it to generate a new random number.

15.3 Interrupt

The RNG provides an interrupt, rng_int, to indicate the RNG has generated a random number.

15.4 SFR Table

RNG_RESULT register: 0x200080

31 0

rng_result

Name	Bit	Write/Read	Reset value	Description
rng_result	[31:0]	WR	32'h0	rng result. Valid when rng_busy=0.

RNG_CTRL register: 0x200084

31 1 0

	rng_start/rng_busy

Name	Bit	Write/Read	Reset value	Description
rng_start	[0]	W	1'h0	Writes 1 to trigger rng start. Auto clear by hardware..
rng_busy	[0]	R	1'h0	Becomes 1 after rng_start is written 1; becomes 0 when rng_result is ready. 1'b1:rng result is not ready. 1'b0:rng result is ready.

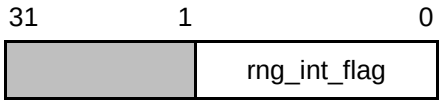
RNG_RST register: 0x200088

31 1 0

	rng_soft_rst

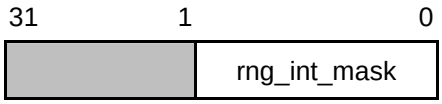
Name	Bit	Write/Read	Reset value	Description
rng_soft_rst	[0]	WR	1'h0	writing 1 to trigger rng soft reset. Auto clear by hardware.

RNG_INT_FLAG register: 0x20008C



Name	Bit	Write/Read	Reset value	Description
rng_int_flag	[0]	W1C	1’h0	rng interrupt flag. Writing 1 to clear it.

RNG_INT_MASK register: 0x200090



Name	Bit	Write/Read	Reset value	Description
rng_int_mask	[0]	WR	1’h0	rng interrupt mask. 1'b1:rng interrupt flag will be issued. 1'b0:rng interrupt flag will not be issued.