MXD2656A1 Product Datasheet v1.2

Key Features

- Bluetooth Low Energy
  - Complies with Bluetooth 5.0 (1Mbps/2Mbps, extended ADV payload)
  - -94dBm sensitivity in 1Mbps mode
  - -20dBm to +5dBm output power
  - Single-pin antenna interface
  - 13mA peak current in TX (0dBm)
  - 12mA peak current in RX
  - RSSI (1dB resolution)
- ARM Cortex-M0+ 32-bit processor, 48MHz
  - 65 uA/MHz running from SRAM
  - Single cycle multiplier
  - Serial Wire Debug
- Memory
  - 80kB ROM
  - 36kB SRAM, 4kB CACHE
  - 512kB flash
  - 32B eFuse
- Flexible power management
  - Supply voltage range 1.7V~3.6V
  - 2uA at 3V in hibernate mode (GPIO state retention), wakeup by GPIO or RTC
  - 5uA at 3V in sleep mode (BLE linked, all 40kB SRAM data retention), wakeup by GPIO, RTC or BLE
- Clock and timer
  - 16MHz crystal oscillator
  - 32KHz and 16MHz RC oscillator
  - 5x 16bit timer
  - RTC
  - Watchdog
- DMA 2 channels
- Peripheral
  - 19 general purpose I/Os with function any-route
  - 4-channel 9-bit general purpose ADC
  - 2x UART with CTS/RTS
  - SPI with master/slave configurable
  - I2C with master/slave configurable
  - 4x PWM
  - Infra-Red generator
  - Quadrature decoder (QDEC) interface
  - 7816 T-0 master interface
  - 12MHz clock output
- Audio Interface
  - Digital I2S mono or stereo data input
  - Digital PDM mono data input
  - 16-bit mono ADC, single-ended or differential analog MIC input, 30dB PGA (3dB step), MICBIAS integrated
- Package QFN32 4 x 4 x 0.75 mm

Applications

- Interactive entertainment devices
  - TV/setup-box remote controls
  - Gaming controllers
- Home automation, Building and Retail
  - E-lock
  - Smart Lighting
  - Electronic Shelf Label
  - Location Based Service
- Personal area networks
  - Medical devices
  - Health
  - Fitness
  - Key finder
- Beacons
- Remote control toys

Contents

1 Revision History

Version	Date	Description of Change
1.0	Sep 5th, 2019	the first release

2 Block Diagram

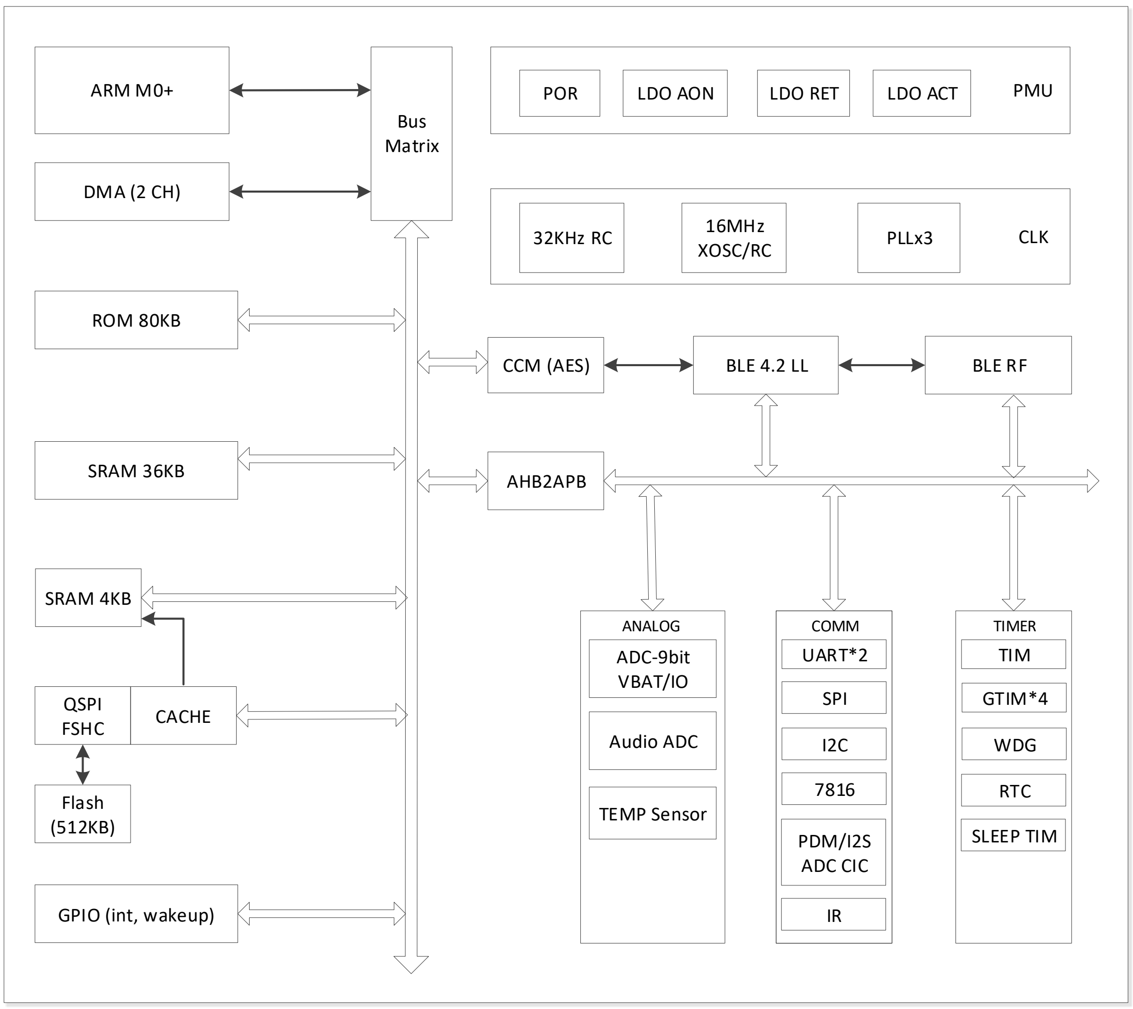


Figure 1 Block Diagram

3 Pin Assignments

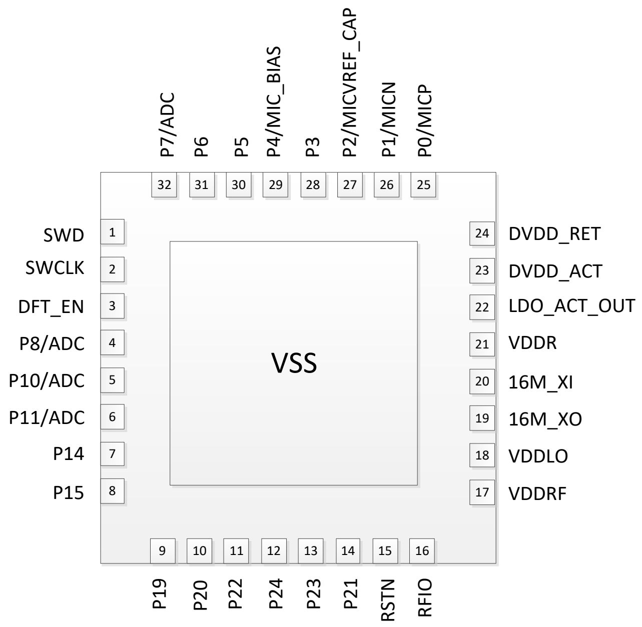


Figure 2 MXD2656A1 QFN32 4*4 Package Top View

Note

DIO (digital bidirectional), *DI* (digital input), *AI* (analog input), *AIO* (analog bidirectional).

I-PD (input pull-down), *I-PU* (input pull-up).

Table 1 Pin Description

Pin Name	Type	Drive (mA)	Reset State	Description
General Purpose I/Os				
P5	DIO	5/10/15/20	I-PD	INPUT/OUTPUT with selectable pull up/down resistor. Pull-down enabled during and after reset. State retention during sleep or hibernate power mode. Digital interface can be routed to any IO of Pn.
Pn	DIO	5/10/15/20	I-PU	INPUT/OUTPUT with selectable pull up/down resistor. Pull-up enabled during and after reset. State retention during sleep or hibernate power mode. Digital interface can be routed to any IO of Pn.
General Purpose I/Os with analog function				
P0/MICP	DIO	5/10/15/20	I-PD	INPUT/OUTPUT with selectable pull up/down resistor. Pull-down enabled during and after reset. State retention during sleep or hibernate power mode. Digital interface can be routed to any IO of

				Pn. It can be used as MICP. In non-MICP mode, P0 always has a 30KΩ pull down resistor. Application design should take care of this, otherwise there may be 0.1mA leakage.
P4/MIC_BIAS	DIO	5/10/15/20	I-PD	INPUT/OUTPUT with selectable pull up/down resistor. Pull-down enabled during and after reset. State retention during sleep or hibernate power mode. Digital interface can be routed to any IO of Pn. It can be used as MICBIAS
P1/MICN	DIO	5/10/15/20	I-PU	INPUT/OUTPUT with selectable pull up/down resistor. Pull-up enabled during and after reset. State retention during sleep or hibernate power mode. Digital interface can be routed to any IO of Pn. It can be used as MICN. In non-MICN mode, P1 always has a 30KΩ pull down resistor. Application design should take care of this, otherwise there may be 0.1mA leakage.
P2/MICVREF_CAP	DIO	5/10/15/20	I-PU	INPUT/OUTPUT with selectable pull up/down resistor. Pull-up enabled during and after reset. State retention during sleep or hibernate power mode. Digital interface can be routed to any IO of Pn. It can be used as MIC VREF Decoupling Capacitance
Pn/ADC	DIO	5/10/15/20	I-PU	INPUT/OUTPUT with selectable pull up/down resistor. Pull-up enabled during and after reset. Contains state retention mechanism during power down. Digital interface can be routed to any IO of Pn. It can be used as ADC input too.
Debug Interface				
SWCLK	DIO	5/10/15/20	I-PD	INPUT/OUTPUT with selectable pull up/down resistor. Pull-down enabled during and after reset. Serial wire clock signal. Can also be used as a GPIO (digital interface any route is not supported).
SWD	DIO	5/10/15/20	I-PU	INPUT/OUTPUT with selectable pull up/down resistor. Pull-up enabled during and after reset. Serial wire data signal. Can also be used as a GPIO (digital interface any route is not supported).

Clocks				
16M_XI	AI			INPUT. Crystal input for the 16MHz XTAL.
16M_XO	AO			OUTPUT. Crystal output for the 16MHz XTAL.
Radio transceiver				
RFIO	AIO			RF input/output.
Miscellaneous				
RSTN	DI			INPUT. Reset signal (active low). Can be floating if not used.
DFT_EN	DI			INPUT. Connect to GND for application.
DVDD_RET	AIO			Connect to external capacitor
DVDD_ACT	AIO			Connect to external capacitor
VDDRF	AIO			Connect to LDO_ACT_OUT
VDDLO	AIO			Connect to LDO_ACT_OUT
Power supply				
VDDR	AIO			Power supply 1.7V~3.6V
VSS	AIO			Ground

4 Specifications

4.1 Absolute Maximum Ratings

Table 2 Absolute Maximum Ratings

Parameter	Ratings			Unit
	Min.	Typ.	Max.	
Power Supply Voltage (VDDR)	-0.5	--	3.9	V
Maximum Junction Temperature	-40	--	125	℃
Storage Temperature	-40	--	125	℃
ESD HBM	-3000	--	3000	V
ESD CDM	-500	--	500	V

4.2 Recommended Operating Conditions

Table 3 Recommended Operating Conditions

Parameter	Ratings			Unit
	Min.	Typ.	Max.	
Power Supply Voltage (VDDR)	1.7	3.0	3.6	V
Operating Temperature	-40	--	85	℃

Ta=25℃, VDDR=3.3V, unless otherwise specified.

4.3 DC Characteristics

Table 4 DC Characteristics

Parameter		Ratings			Unit
		Min.	Typ.	Max.	
VIH (Logic-1 input voltage)	VDDR = 3.3V	2.0			V
	VDDR = 2.5V	1.7			V
	VDDR = 1.8V	1.2			V
IIH (Logic-1 input current)				+10	uA
VIL (Logic-0 input voltage)	VDDR = 3.3V			0.8	V
	VDDR = 2.5V			0.7	V
	VDDR = 1.8V			0.6	V
IIL (Logic-0 input current)		-10			uA
VOH (Logic-1 output voltage, IOL = -20/-15/-10/-5mA)		VDDR - 0.4			V
VOL (Logic-0 output voltage, IOH = -20/-15/-10/-5mA)				0.4	V
Rx Mode, 1Mbps mode			12		mA

TX mode, 0 dBm output power		13		mA
CPU running in SRAM		65		uA/MHz
Hibernate mode (SRAM no retention, IO retention and IO wakeup or RTC wakeup)		2		uA
Sleep mode (SRAM 40kB retention, BLE linked, flash sleep included)		5		uA

Ta=25℃, VDDR=3.0V, unless otherwise specified.

4.4 Transceiver Characteristics

Table 5 RX Characteristics

Parameters		Symbol	Min	Typ	Max	Unit
Sensitivity		P_{MIN}		-94		dBm
Sensitivity(dirty on)		P_{MIN}		-93		dBm
Maximum input power		P_{MAX}		0		dBm
In-band blocking	Co-channel interference	CI0		7		dB
	Interferer at f_{offs}= +1MHz	CI1		-1		dB
	Interferer at f_{offs}= -1MHz	CI1		-6		dB
	Interferer at f_{offs}= +2MHz	CI2		-38		dB
	Interferer at f_{offs}= -2MHz	CI2		-35		dB
	Interferer at f_{offs}= +3MHz	CI3		-42		dB
	Interferer at f_{offs}= -3MHz	CI3		-33		dB
	Interferer at image channel (Fimage)	CI4		-29		dB
	Interferer at image channel (Fimage+1MHz)	CI5		-33		dB
	Interferer at image channel (Fimage-1MHz)	CI5		-32		dB
Out-of-band blocking	f= 30–2000MHz			>-25		dB
	f= 2000–2399 MHz			>-30		dB
	f= 2484–3000 MHz			>-30		dB
	f= 3000–12750 MHz			>-25		dB
Intermodulation Performance for Wanted Signal at -64dBm and 1 Mbps BLE, 3rd, 4th and 5th offset channel				>-35		dB
Upper limit of input power range over which RSSI resolution is maintained		P_{RSSI_MAX}		-20		dBm

Ta=25℃, VDDR=3.0V, unless otherwise specified.

Table 6 TX Characteristics

Parameters	Symbol	Min	Typ	Max	Unit
Output power	P_{TX}	-20		+5	dBm
TX RF Output Steps			3		dB
Average Frequency deviation for 10101010 pattern	ΔF2AVG		238		KHz

Average Frequency deviation for 11110000 pattern		ΔF1AVG		260		KHz
Eye opening = ΔF2AVG/ΔF1AVG		EO	0.88	0.91	0.94	
Frequency Accuracy			-10		10	KHz
Maximum Frequency Drift			-6		5	KHz
Initial Frequency drift			-5		5	KHz
Drift rate		FDR	-5.5		4.5	KHz/50μs
Spurious Emissions	F < 1 GHz			-68		dBm
	F > 1 GHz including harmonics			-48		dBm
In-band Emissions	< f ± 2MHz (f=2400~2483.5MHz,Ptx=5dBm)			-45		dBm
	> f ± 3MHz (f=2400~2483.5MHz,Ptx=5dBm)			-49		dBm

Ta=25℃, VDDR=3.0V, unless otherwise specified.

4.5 Audio Characteristics

Table 7 Audio Characteristics

Parameters	Min	Typ	Max	Unit
ADC Input Signal Level (0dB)	-	0.8	-	Vrms
ADC ENOB	-	12	-	Bit
ADC THD+N	-	74	-	dB
ADC Conversion Rate	-	16	-	KHz
ADC Signal Bandwidth	20	-	8K	Hz
PGA gain range	0	-	30	dB
PGA step	-	3	-	dB
MIC bias output voltage	-	2.0, 2.5, or VDDR	-	V
MIC bias loading current	-	-	5	mA
Input Impedance (Resistance)	-	30	-	KΩ
Input Impedance (Capacitance)	-	1	-	pF
Current consumption	-	1	-	mA
Total startup time	-	10	-	ms

Ta=25℃, VDDR=3.0V, unless otherwise specified.

5 Function Blocks

5.1 Power Mode

Hibernate mode

All IOs hold their status they had before entering hibernate mode. All power domains are off except PD_AON domain (IO, PMU). All clock sources are off except RC32K. IO or RTC can wake up the device from the hibernate mode. After wakeup, CPU will reset and restart the instructions from 0x0 address. CPU can differentiate whether it is wakeup or reset (pin reset, power-on-reset) through the reset status register.

Sleep mode

All IOs and 40KB SRAMs hold their status they had before entering sleep mode. All clock sources are off except RC32K. This mode can be waked up by any I/O pin, RTC, or Link layer timer. After wakeup, CPU will continue the instructions from where it went into Sleep.

5.2 ARM Cortex-M0+ CPU

The Cortex-M0+ processor is a 32-bit Reduced Instruction Set Computing (RISC) processor with a von Neumann architecture (single bus interface). It implements the ARMv6-M architecture, which is based on the 16-bit Thumb (ARM7TDMI) instruction set and includes Thumb-2 technology. This provides the exceptional performance expected of a modern 32-bit architecture, with a higher code density than 8-bit and 16-bit microcontrollers.

Table 8 CPU Interrupt Map

CPU Interrupt Number #	Module
IRQ_CPU_NMI	Watchdog Non-Maskable Interrupt
IRQ_CPU_0	Power Management Unit Interrupt
IRQ_CPU_1	Link Layer Controller Interrupt
IRQ_CPU_2	UART0 Interrupt
IRQ_CPU_3	UART1 Interrupt
IRQ_CPU_4	SPI Interrupt
IRQ_CPU_5	-
IRQ_CPU_6	DMA Interrupt (DMA_INT0)
IRQ_CPU_7	DMA Interrupt (DMA_INT1)
IRQ_CPU_8	I2S Interrupt
IRQ_CPU_9	I2C Interrupt
IRQ_CPU_10	QDEC Interrupt
IRQ_CPU_11	SCC/7816 Interrupt
IRQ_CPU_12	Audio CIC Interrupt
IRQ_CPU_13	RTC Interrupt
IRQ_CPU_14	TIMER Interrupt (TIMER_INT0)

IRQ_CPU_15	TIMER Interrupt (TIMER_INT1)
IRQ_CPU_16	TIMER Interrupt (TIMER_INT2)
IRQ_CPU_17	GPIO Interrupt (GPIO_INT0)
IRQ_CPU_18	GPIO Interrupt (GPIO_INT1)
IRQ_CPU_19	GPIO Interrupt (GPIO_INT2)
IRQ_CPU_20	-
IRQ_CPU_21	CCM
IRQ_CPU_22	-
IRQ_CPU_23	-
IRQ_CPU_24	RF Calibration
IRQ_CPU_25	Software Defined Interrupt (SW0)
IRQ_CPU_26	Software Defined Interrupt (SW2)
IRQ_CPU_27	Software Defined Interrupt (SW3)
IRQ_CPU_28	Software Defined Interrupt (SW3)
IRQ_CPU_29~31	-

5.3 Bluetooth Low Energy

The BLE radio transmits and receives GFSK packets at 2Mbps/1Mbps over a 2.4GHz ISM band, which is compliant with Bluetooth specification 5.0. The RF transceiver contains an integrated balun, which provides a single-ended RF port pin to drive a 50Ω antenna via a matching/filtering network. In the receive direction, it converts the RF signal from the antenna to a digital bit stream after performing GFSK demodulation. In the transit direction, it performs GFSK modulation and then converts a digital baseband signal to a radio frequency before transmitting it to air through the antenna.

The BLE radio can estimate an accurate center frequency offset in case the opposite device crystal oscillator has a frequency PPM offset. In addition, the BLE radio can receive the packet with large carrier frequency offset (300KHz).

The BLE core is a qualified Bluetooth baseband controller compatible with the Bluetooth Smart specification and it is in charge of packet encoding/decoding and frame scheduling. It performs Link Layer Control management supporting the main BLE states, including advertising and connection.

Feature

1. All device classes support (Broadcaster, Central, Observer and Peripheral)
2. Simultaneous Master and Slave operation
3. Frequency Hopping
4. All packet types (Advertising / Data / Control)
5. Encryption (AES / CCM)
6. Bit stream processing (CRC, Whitening)
7. Operating clock 16MHz

8. Low power mode with 32KHz timer counting

5.4 Memories

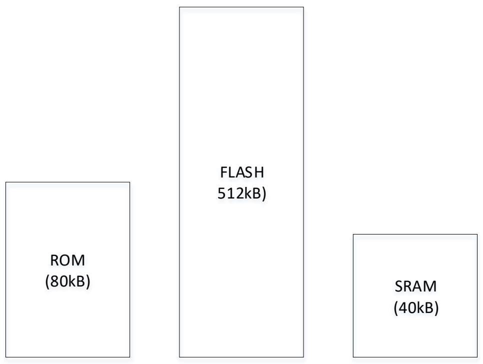


Figure 3 Memories

ROM

80kB ROM contains the Bluetooth Smart protocol stack as well as the boot code.

Retention SRAM

40kB retention SRAM stores various data of the Bluetooth Smart protocol as well as the system's global variables. The SRAM data is retained when the system goes into sleep mode. Storage of this data ensures secure and quick configuration of the BLE Core after the system wake up.

Flash

512KB Flash memory stores the application code and user data for all the Bluetooth Smart devices.

5.5 QSPI Flash Controller

The QSPI (Quad Serial Peripheral Interface) Flash Controller is used to read, write and erase FLASH memory. It supports the standard SPI, DSPI (Dual Serial Peripheral Interface) and QSPI (Quad Serial Peripheral Interface) interface formats. The QSPI flash controller has two AHB slave buses, one connecting with MCU to transfer data, and the other connecting with cache to fetch instructions.

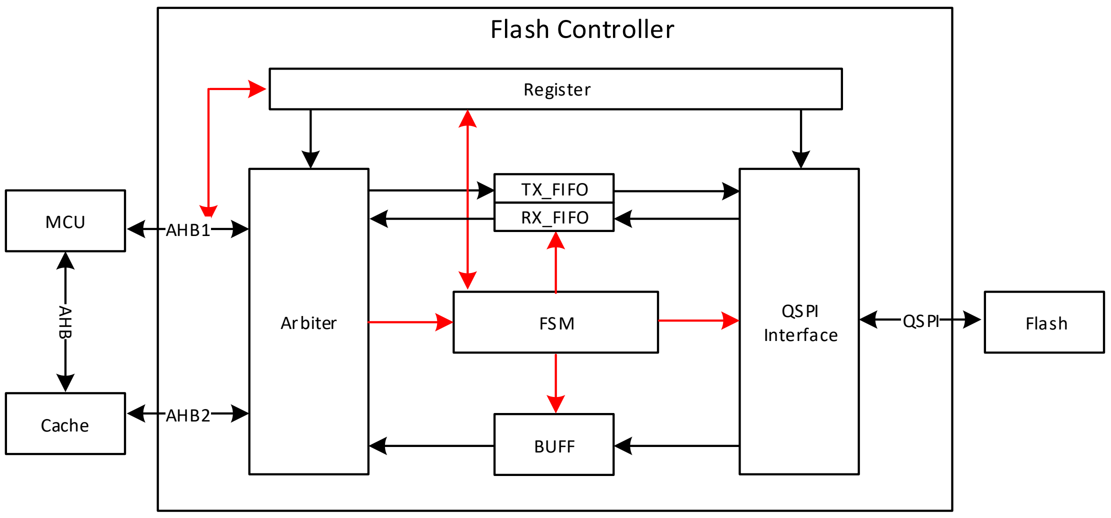


Figure 4 QSPI Flash Controller Block Diagram

5.5.1 Feature

1. SPI, DSPI, QSPI
2. Up to 48 MHz clock
3. Support CACHE direct instruction read from flash
4. Support MCU indirect data read from flash and data write to flash
5. Support MCU indirect status read from flash and status write to flash
6. Support chip/page/sector/block erase
7. 100,000 program/erase cycles
8. 20-year data retention

5.5.2 Function Description

5.5.2.1 Cache Direct Read

Cache direct read mode is used to read CPU instruction code. Cache fetches 4 words (32bits per word) for one cache line if cache miss.

5.5.2.2 MCU Indirect Access

There are four types of MCU indirect access types.

- Erase
- Program
- MCU read/write flash status

- MCU read data

When Erase or Program, the MCU must run from RAM and make sure all interrupts are disabled while in erasing or programming. CPU can read the flash status register (Read Status Register) to confirm that the erase or program is finished.

Erase operations include block erase, sector erase and page erase. Page is the smallest erase unit. After erasing a flash page, all bits in the page are set to '1'. The typical erase time is 8ms.

The flash can only be programmed from '1' to '0'. Before program, make sure that the address you want to program has been erased. The smallest program unit is 1 byte. Typical page program time is 2ms.

MCU read/write flash status/data via register. Read or write flash status register only support SPI mode.

The Flash can be read an unlimited number of times by the CPU via SPI, DSPI or QSPI mode, and burst length is unlimited. The first address byte can be at any location. The address is automatically increased to the next higher address after each byte data is shifted out, so the whole memory can be read out at a single READ instruction.

5.5.2.3 Flash Clock Selection

The Flash clock can select either 16M or PLL (48M) clock.

5.6 CLOCK

5.6.1 Feature

1. 16MHz crystal oscillator
2. 32KHz and 16MHz RC oscillator
3. 48MHz PLL

5.6.2 Function Description

The system clock tree is described in detail in the following figure. It depicts the possible clock sources as well as all different divisions and multiplexing paths towards the generation of each block’s clock.

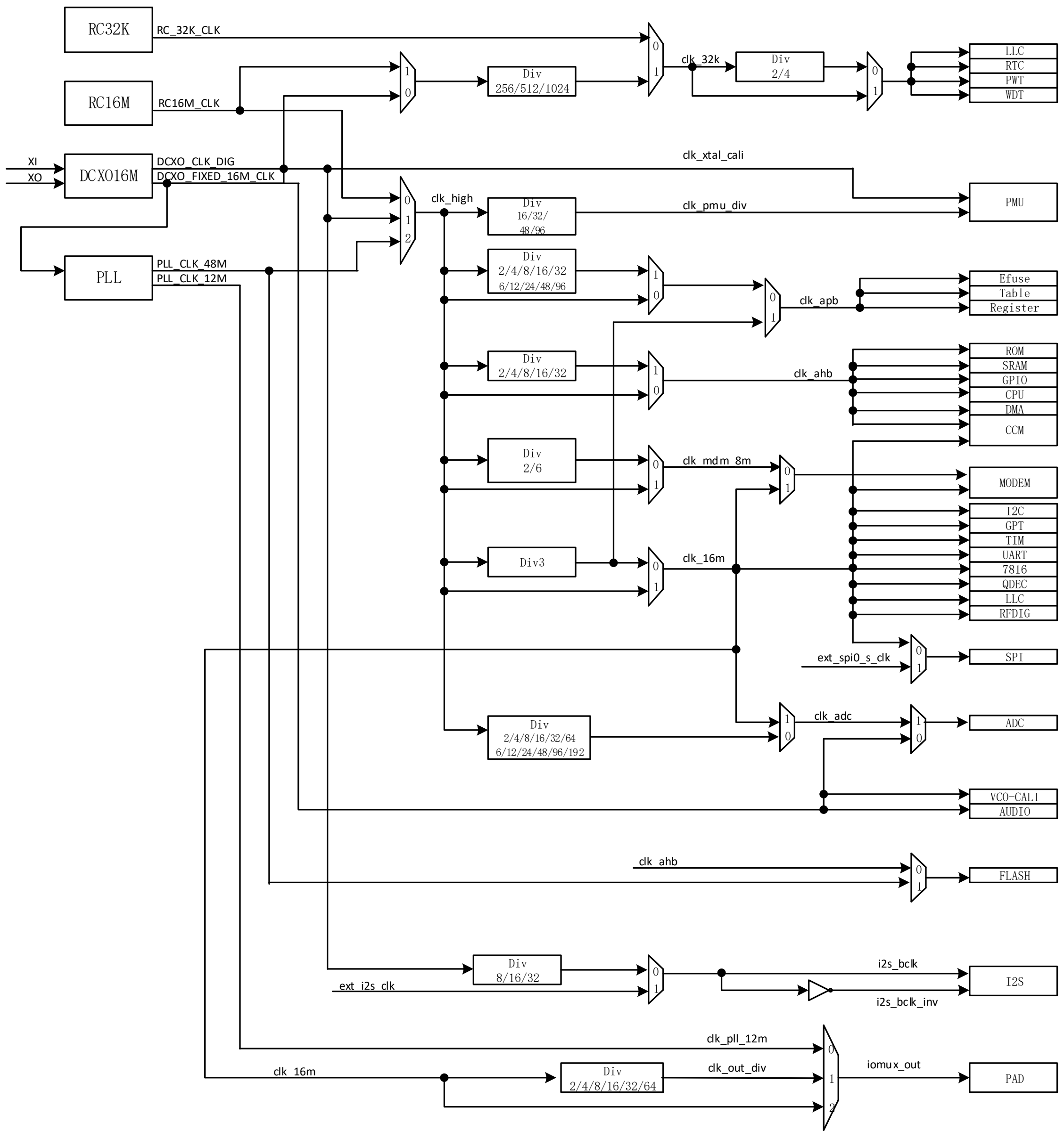


Figure 5 Clock Tree Diagram

Two main clocks:

- clk_high: This is the system clock for CPU, AMBA bus (hclk/pclk), memories and some peripherals. This clock source can be 16MHz RC oscillator, PLL or 16MHz crystal oscillator.
- clk_32KHz: This is the low power clock used for low power counter. The source can be either RC32KHz or RC16MHz.

The 16MHz Crystal Oscillator (XO) must be used for the BLE operations or in the application where a very accurate clock is required for the ARM subsystem operations.

5.7 RESET

The device has several sources of reset. Resets may result in reset of the following:

1. The entire chip
2. A power domain
3. A voltage domain
4. Some digital modules

Figure below shows the diagram of reset.

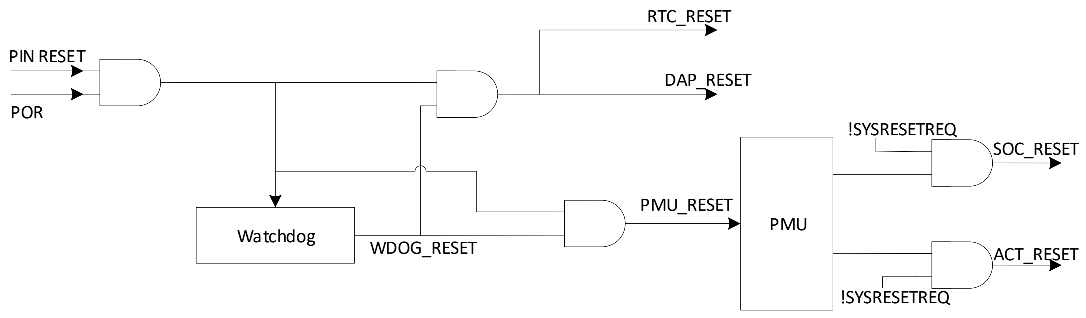


Figure 6 Reset Diagram

The Power-on-Reset (POR) signal is generated by the analog circuitry contained in the device. POR resets the whole chip.

Watchdog reset is used to recover from software crashes. The watchdog contains a 32-bit down counter with 32KHz internal RC clock which generates a Non-Maskable interrupt. If the interrupt is not serviced, the watchdog generates reset.

The system reset request is generated by the debug circuitry of the Cortex-M0. The debugger writes to the SYSRESETREQ bit. The system reset request does not affect the debugger, thus allowing the debugger to remain connected during the reset sequence.

The scope of all resets is listed in the following table.

Table 9 Reset Functions

Module	POR	Pin Reset	Watchdog Reset	System Reset	Hibernate Wakeup	Sleep Wakeup	Vector Reset
ARM_Debug	X	X	X				
RTC	X	X	X				
PMU	X	X	X				
ARM_Core	X	X	X	X	X		X
Configuration for BLE	X	X	X	P	X		
Data Path for BLE	X	X	X	X	X	X	
Watchdog Timer	X	X	X		X		
GPIO	X	X	X				

Peripherals	X	X	X	X	X	X	
RAM Retention	X	X	X	X	X		

Note:

- X: Indicate that this module will be reset or RAM's content will not be available.
- P: Indicate that most registers are reset except some special registers.

5.8 DMA

The DMA controller provides a way to offload data transfer tasks from the CPU. The DMA controller can perform transfers between memory and peripherals. The controller has dedicated channels for each supported on-chip module, and can be programmed to automatically perform transfers between peripherals and memories as the peripheral is ready to transfer more data.

5.8.1 Feature

1. Support memory to memory, memory to peripheral and peripheral to memory transmission
2. Support UART, SPI, I2C, 7816, I2S, CIC interface
3. 2 independent DMA channels
4. Configurable 2 level priority
5. Independent source and destination transfer size (8bit, 16bit, 32bit)
6. Support circular mode
7. Programmable number of data to be transferred: up to 65535
8. Source and destination address increment or no increment

5.8.2 Function Description

The DMA controller performs direct memory transfer by sharing the system bus with the other masters of the device. The DMA request may stop the CPU access to the system bus for some bus cycles, when the CPU and DMA are accessing the same destination (memory or peripheral). The bus matrix implements round-robin scheduling, thus ensuring at least half of the system bus bandwidth (both to memory and peripheral) for the CPU.

5.8.2.1 DMA Peripherals

The DMA controller has two channels for fast data transfers between SPI, UART, I2C, 7816, CIC/I2S and on-chip RAM. Every peripheral is allocated with one dedicated ID, and each DMA channel can select any peripheral per ID.

Table 10 DMA Peripherals Selection

Register (DMAx_PERI_SEL) Value	Name and Direction
0	UART0 TX
1	UART0 RX

2	UART1 TX
3	UART1 RX
4	SPI0 TX
5	SPI0 RX
6	Reserved
7	Reserved
8	I2C0 TX
9	I2C0 RX
10	Reserved
11	Reserved
12	7816 TX
13	7816 RX
14	CIC RX
15	I2S RX

5.8.2.2 DMA Channel Enable and Pause

A DMA channel is switched on with bit DMAx_EN. If DMA starts, data is transferred from address DMAx_SRC_ADDR to address DMAx_DST_ADDR for a length of DMAx_CFG_CNT, which can be 8, 16 or 32 bits wide. The address increment is implemented using a 16-bit index counter (DMAx_TCNT) which is initialized to ‘0’ when the transfer starts. This register is increased by 1 at the end of each DMA cycle and is then compared to DMAx_CFG_CNT (transfer completion or not). If completed, it is then automatically reset to ‘0’ again.

Each channel of the DMA controller can be temporarily disabled by writing a ‘1’ at bit DMAx_PAUSE in the DMAx_CONFIG register. To enable the channel again, a ‘0’ to bit at DMAx_PAUSE must be written.

5.8.2.3 DMA Transfer Size

A DMA transfer is the smallest unit of data that can be transferred by the DMA. The DMA supports byte, half-word and word sized transfers. The DMAx_SRC_SIZE/DMAx_DST_SIZE register specifies the data width of one DMA transfer for source/destination (0: 1byte, 1: 2bytes, 2/3: 4bytes).

The transfer count DMAx_CFG_CNT defines how many DMA transfers to perform. The total transfer byte number is (DMAx_CFG_CNT * DMAx_SRC_SIZE). Before DMA completion, the number of bytes transferred is (DMAx_TCNT * DMAx_SRC_SIZE).

5.8.2.4 DMA Arbitration

The arbiter manages the channel requests based on their priority and launches the peripheral/memory access sequences. The priority level of a DMA channel can be set with bit DMAx_PRI. These bits determine which DMA channel will be activated in case more than one DMA channel requests DMA. If two channels have the same priority, DMA0 will win the arbitration.

5.8.2.5 Circular Mode

Circular mode is available to handle circular buffers and continuous data flow. This feature can be enabled using the DMAx_CIRC bit in the DMAx_CONFIG register.

When circular mode is activated, after the last transfer, the number of data to be transferred is automatically reloaded with the initial value programmed during the channel configuration phase. DMA will serve the configured request as before.

5.8.2.6 TX Interval Mode

TX interval mode provides a quick DMA operation mode if the data is not all ready when triggering DMA. This mode is enabled by DMAx_INTER_TX_MODE bit.

The DMA transfer length is calculated as below.

- For the first time, it is DMAx_INTER_TX_OFFSET.
- For the later times, there are two scenarios.
  - ■ If the new DMAx_INTER_TX_OFFSET is larger than the old one, it is (new DMAx_INTER_TX_OFFSET - old DMAx_INTER_TX_OFFSET).
  - ■ If the new DMAx_INTER_TX_OFFSET is smaller than the old one, it is (new DMAx_INTER_TX_OFFSET - old DMAx_INTER_TX_OFFSET + DMAx_CFG_CNT).

5.8.3 Software Procedure

The following sequence should be followed to configure a DMA channel x (where x is the channel number).

- Data from peripheral to memory
  1. Set the peripheral register address in the DMAx_SRC_ADDR register. The data is moved from this address after the peripheral event.
  2. Set the memory address in the DMAx_DST_ADDR register. The data will be written to this memory address after the peripheral event.
  3. Configure the total number of data to be transferred in the bits DMAx_CFG_CNT in the DMAx_CONFIG register. After each peripheral event, this value is decremented.
  4. Configure the channel priority using the DMAx_PRI bit in the DMAx_CONFIG register.
  5. Configure data transfer direction, circular mode, peripheral and memory incremented mode, peripheral and memory data size, and interrupt after half and/or full transfer in the DMAx_CONFIG register.
  6. Activate the channel by setting the DMAx_EN bit in the DMAx_CONFIG register.

- Data from memory to peripheral
    1. Set the memory register address in the DMAx_SRC_ADDR register. The data is moved from this address before the peripheral event.
    2. Set the peripheral address in the DMAx_DST_ADDR register. The data will written be to this address.
    3. Configure the total number of data to be transferred in the bits DMAx_CFG_CNT in the DMAx_CONFIG register. After each peripheral event, this value is decremented.
    4. Configure the channel priority using the DMAx_PRI bit in the DMAx_CONFIG register.
    5. Configure data transfer direction, circular mode, peripheral and memory incremented mode, peripheral and memory data size, and interrupt after half and/or full transfer in the DMAx_CONFIG register.
    6. Activate the channel by setting the DMAx_EN bit in the DMAx_CONFIG register.

- Data from memory to memory
    1. Set the memory register address in the DMAx_SRC_ADDR register. The data is moved from this address.
    2. Set the memory address in the DMAx_DST_ADDR register. The data will be written to this address.
    3. Configure the total number of data to be transferred in the bits DMAx_CFG_CNT in the DMAx_CONFIG register. After each peripheral event, this value is decremented.
    4. Configure the channel priority using the DMAx_PRI bit in the DMAx_CONFIG register.
    5. Configure data transfer direction, circular mode, peripheral and memory incremented mode, peripheral and memory data size, and interrupt after half and/or full transfer in the DMAx_CONFIG register.
    6. Activate the channel by setting the DMAx_EN bit in the DMAx_CONFIG register.

As soon as the channel is enabled, it can serve any DMA request from the peripheral connected on the channel or memory connected. At the end of the transfer, the transfer complete interrupt is generated if the transfer complete interrupt enable bit (DMAx_4/4_DONE in the DMA_INT_MASK register) is set.

5.8.4 DMA Interrupt

DMA interrupt table is shown below.

Table 11 DMA Interrupt

Interrupt Number	Interrupt Name	Description
0	DMA0_TIMEOUT	DMA0 timeout, no data is transferred
1	DMA0_4/4_DONE	DMA0 has finished all data transfer length
2	DMA0_3/4_DONE	DMA0 has finished 3/4 data transfer length
3	DMA0_2/4_DONE	DMA0 has finished 2/4 data transfer length
4	DMA0_1/4_DONE	DMA0 has finished 1/4 data transfer length
5	DMA1_TIMEOUT	DMA1 timeout, no data is transferred
6	DMA1_4/4_DONE	DMA1 has finished all data transfer length
7	DMA1_3/4_DONE	DMA1 has finished 3/4 data transfer length
8	DMA1_2/4_DONE	DMA1 has finished 2/4 data transfer length
9	DMA1_1/4_DONE	DMA1 has finished 1/4 data transfer length

5.8.5 DMA Registers

Table 12 DMA Instance

Base Address	Peripheral	Instance	Description
0x40002000	DMA	DMA	Direct Memory Access

Table 13 DMA Register Map

Address Offset	Name	Description
0x00	DMA_INT_STATUS	DMA Interrupt Status
0x04	DMA_INT_MASK	DMA Interrupt Enable
0x08	DMA_INT_CLEAR	DMA Interrupt Flag Clear Register
0x0C	DMA0_CONFIG	DMA0 Configuration Register
0x10	DMA0_SRC_ADDR	DMA0 Source Start Address
0x14	DMA0_DST_ADDR	DMA0 Destination Start Address
0x18	DMA0_TCNT	DMA0 Transferred Data Number
0x1C	DMA1_CONFIG	DMA1 Configuration Register
0x20	DMA1_SRC_ADDR	DMA1 Source Start Address
0x24	DMA1_DST_ADDR	DMA1 Destination Start Address
0x28	DMA1_TCNT	DMA1 Transferred Data Number
0x2C	DMA_INT_SEL	DMA Interrupt Selection
0x30	DMA_TO_THLD	DMA Timeout Threshold
0x34	DMA0_INTER_TX_CFG	DMA0 TX Interval Mode Configuration
0x38	DMA1_INTER_TX_CFG	DMA1 TX Interval Mode Configuration

Table 14 DMA_INT_STATUS (DMA_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
9	DMA1_1/4_DONE	0	R	DMA1 has finished 1/4 data transfer length. 0: invalid 1: valid
8	DMA1_2/4_DONE	0	R	DMA1 has finished 2/4 data transfer length. 0: invalid 1: valid
7	DMA1_3/4_DONE	0	R	DMA1 has finished 3/4 data transfer length. 0: invalid 1: valid
6	DMA1_4/4_DONE	0	R	DMA1 has finished all data transfer length. 0: invalid 1: valid
5	DMA1_TIMEOUT	0	R	DMA1 timeout

				0: invalid 1: valid
4	DMA0_1/4_DONE	0	R	DMA0 has finished 1/4 data transfer length. 0: invalid 1: valid
3	DMA0_2/4_DONE	0	R	DMA0 has finished 2/4 data transfer length. 0: invalid 1: valid
2	DMA0_3/4_DONE	0	R	DMA0 has finished 3/4 data transfer length. 0: invalid 1: valid
1	DMA0_4/4_DONE	0	R	DMA0 has finished all data transfer length. 0: invalid 1: valid
0	DMA0_TIMEOUT	0	R	DMA0 timeout 0: invalid 1: valid

Table 15 DMA_INT_MASK (DMA_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
9	DMA1_1/4_DONE_MASK	0	RW	DMA1_1/4_DONE interrupt enable 0: disable 1: enable
8	DMA1_2/4_DONE_MASK	0	RW	DMA1_2/4_DONE interrupt enable 0: disable 1: enable
7	DMA1_3/4_DONE_MASK	0	RW	DMA1_3/4_DONE interrupt enable 0: disable 1: enable
6	DMA1_4/4_DONE_MASK	0	RW	DMA1_4/4_DONE interrupt enable 0: disable 1: enable
5	DMA1_TIMEOUT_MASK	0	RW	DMA1 Timeout interrupt enable 0: disable 1: enable
4	DMA0_1/4_DONE_MASK	0	RW	DMA0_1/4_DONE interrupt enable 0: disable 1: enable
3	DMA0_2/4_DONE_MASK	0	RW	DMA0_2/4_DONE interrupt enable 0: disable 1: enable
2	DMA0_3/4_DONE_MASK	0	RW	DMA0_3/4_DONE interrupt enable

				0: disable 1: enable
1	DMA0_4/4_DONE_MASK	0	RW	DMA0_4/4_DONE interrupt enable 0: disable 1: enable
0	DMA0_TIMEOUT_MASK	0	RW	DMA0 timeout interrupt enable 0: disable 1: enable

Table 16 DMA_INT_CLEAR (DMA_BASE_ADDR+0x08)

Bit	Field Name	Reset	RW	Description
9	DMA1_1/4_DONE_CLEAR	0	W	DMA1_1/4_DONE interrupt clear 0: invalid 1: clear
8	DMA1_2/4_DONE_CLEAR	0	W	DMA1_2/4_DONE interrupt clear 0: invalid 1: clear
7	DMA1_3/4_DONE_CLEAR	0	W	DMA1_3/4_DONE interrupt clear 0: Invalid 1: Clear
6	DMA1_4/4_DONE_CLEAR	0	W	DMA1_4/4_DONE interrupt clear 0: invalid 1: clear
5	DMA1_TIMEOUT_CLEAR	0	W	DMA1 timeout interrupt clear 0: invalid 1: clear
4	DMA0_1/4_DONE_CLEAR	0	W	DMA0_1/4_DONE interrupt clear 0: invalid 1: clear
3	DMA0_2/4_DONE_CLEAR	0	W	DMA0_2/4_DONE interrupt clear 0: invalid 1: clear
2	DMA0_3/4_DONE_CLEAR	0	W	DMA0_3/4_DONE interrupt clear 0: invalid 1: clear
1	DMA0_4/4_DONE_CLEAR	0	W	DMA0_4/4_DONE interrupt clear 0: invalid 1: clear
0	DMA0_TIMEOUT_CLEAR	0	W	DMA0 timeout interrupt clear 0: invalid 1: clear

Table 17 DMA0_CONFIG (DMA_BASE_ADDR+0x0C)

Bit	Field Name	Reset	RW	Description
31	DMA0_EN	0	RW	DMA0 enable control 0: disable 1: enable
30	DMA0_PAUSE	0	RW	DMA0 pause control 0: invalid 1: pause transfer
29	DMA0_SRC_INC	0	RW	DMA0 increment of source address 0: not increment, source address stays the same during the transfer 1: increment, according to the value DMA0_SRC_SIZE (by 1, when DMA0_SRC_SIZE is 0; by 2, when DMA0_SRC_SIZE is 1; by 4, when DMA0_SRC_SIZE is 2 or 3)
28:27	DMA0_SRC_SIZE	0x0	RW	DMA0 source bus transfer width 0: 1 byte 1: 2 bytes 2~3: 4 bytes
26	DMA0_DST_INC	0	RW	DMA0 increment of destination address 0: not increment, destination address stays the same during the transfer 1: increment, according to the value DMA0_DST_SIZE (by 1, when DMA0_DST_SIZE is 0; by 2, when DMA0_DST_SIZE is 1; by 4, when DMA0_DST_SIZE is 2 or 3)
25:24	DMA0_DST_SIZE	0x0	RW	DMA0 destination bus transfer width 0: 1 byte 1: 2 bytes 2~3: 4 bytes
23	DMA0_MEM_TO_MEM	0	RW	Data transfer from memory to memory 0: Data from memory to peripherals, or from peripherals to memory 1: Data from memory to memory
22	DMA0_MEM_TO_PERI	0	RW	Data transmission from memory to peripherals 0: Peripherals to memory 1: Memory to peripherals Note: DMA0_MEM_TO_MEM must be 0, otherwise this bit is invalid
21:18	DMA0_PERI_SEL	0x0	RW	DMA0 peripherals selection 0: uart 0 tx 1: uart 0 rx

				2: uart 1 tx 3: uart 1 rx 4: spi 0 tx 5: spi 0 rx 6: reserved 7: reserved 8: i2c 0 tx 9: i2c 0 rx 10: reserved 11: reserved 12: 7816 tx 13: 7816 rx 14: cic rx 15: i2s rx
17	DMA0_PRI	0	RW	DMA0 priority level 0: lowest priority 1: highest priority Note: The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are active and request the bus at the same time. The greater the value, the higher the priority. If different channels with the equal priority level values request the bus at the same time, DMA0 will first be granted access to the bus.
16	DMA0_CIRC	0	RW	DMA0 transfer data mode 0: normal mode. DMA0 stops after having completed the transfer of length determined by DMA0_CFG_CNT. 1: circular mode, DMA0 automatically starts a new transfer after having completed the transfer of length determined by DMA0_CFG_CNT register.
15:0	DMA0_CFG_CNT	0x0	RW	DMA0 transfer length, unit is register (DMA0_SRC_SIZE) value

Table 18 DMA0_SRC_ADDR (DMA_BASE_ADDR+0x10)

Bit	Field Name	Reset	RW	Description
31:0	DMA0_SRC_ADDR	0x0	RW	DMA0 source start address

Table 19 DMA0_DST_ADDR (DMA_BASE_ADDR+0x14)

Bit	Field Name	Reset	RW	Description

31:0	DMA0_DST_ADDR	0x0	RW	DMA0 destination start address

Table 20 DMA0_TCNT (DMA_BASE_ADDR+0x18)

Bit	Field Name	Reset	RW	Description
15:0	DMA0_TCNT	0x0	R	DMA0, The data number has been transferred, unit is register (DMA0_SRC_SIZE) value

Table 21 DMA1_CONFIG (DMA_BASE_ADDR+0x1C)

Bit	Field Name	Reset	RW	Description
31	DMA1_EN	0	RW	DMA1 enable control 0: disable 1: enable
30	DMA1_PAUSE	0	RW	DMA1 pause control 0: invalid 1: pause transfer
29	DMA1_SRC_INC	0	RW	DMA1 increment of source address 0: not increment, source address stays the same during the transfer 1: increment, according to the value DMA1_SRC_SIZE (by 1, when DMA1_SRC_SIZE is 0; by 2, when DMA1_SRC_SIZE is 1; by 4, when DMA1_SRC_SIZE is 2 or 3)
28:27	DMA1_SRC_SIZE	0x0	RW	DMA1 source bus transfer width 0: 1 byte 1: 2 bytes 2~3: 4 bytes
26	DMA1_DST_INC	0	RW	DMA1 increment of destination address 0: not increment, destination address stays the same during the transfer 1: increment, according to the value DMA1_DST_SIZE (by 1, when DMA1_DST_SIZE is 0; by 2, when DMA1_DST_SIZE is 1; by 4, when DMA1_DST_SIZE is 2 or 3)
25:24	DMA1_DST_SIZE	0x0	RW	DMA1 destination bus transfer width 0: 1 byte 1: 2 bytes 2~3: 4 bytes
23	DMA1_MEM_TO_MEM	0	RW	Data transfer from memory to memory 0: Data from memory to peripherals, or from peripherals to memory 1: Data from memory to memory
22	DMA1_MEM_TO_PERI	0	RW	Data transmission from memory to peripherals

				0: Peripherals to memory 1: Memory to peripherals Note: DMA1_MEM_TO_MEM must be 0, otherwise this bit is invalid
21:18	DMA1_PERI_SEL	0x0	RW	DMA1 peripherals selection 0: uart 0 tx 1: uart 0 rx 2: uart 1 tx 3: uart 1 rx 4: spi 0 tx 5: spi 0 rx 6: reserved 7: reserved 8: i2c 0 tx 9: i2c 0 rx 10: reserved 11: reserved 12: 7816 tx 13: 7816 rx 14: cic rx 15: i2s rx
17	DMA1_PRI	0	RW	DMA1 priority level 0: lowest priority 1: highest priority Note: The priority level determines which DMA channel will be granted access for transferring data, in case more than one channels are active and request the bus at the same time. The greater the value, the higher the priority. If different channels with the equal priority level values request the bus at the same time, DMA1 will first be granted access to the bus.
16	DMA1_CIRC	0	RW	DMA1 transfer data mode 0: normal mode. DMA1 stops after having completed the transfer of length determined by DMA1_CFG_CNT. 1: circular mode, DMA1 automatically starts a new transfer after having completed the transfer of length determined by DMA1_CFG_CNT register.
15:0	DMA1_CFG_CNT	0x0	RW	DMA1 transfer length, unit is register (DMA1_SRC_SIZE) value

Table 22 DMA1_SRC_ADDR (DMA_BASE_ADDR+0x20)

Bit	Field Name	Reset	RW	Description
31:0	DMA1_SRC_ADDR	0x0	RW	DMA1 source start address

Table 23 DMA1_DST_ADDR (DMA_BASE_ADDR+0x24)

Bit	Field Name	Reset	RW	Description
31:0	DMA1_DST_ADDR	0x0	RW	DMA1 destination start address

Table 24 DMA1_TCNT (DMA_BASE_ADDR+0x28)

Bit	Field Name	Reset	RW	Description
15:0	DMA1_TCNT	0x0	R	DMA1, The data number has been transferred, unit is register (DMA1_SRC_SIZE) value

Table 25 DMA_INT_SEL (DMA_BASE_ADDR+0x2C)

Bit	Field Name	Reset	RW	Description
1:0	DMA_INT_SEL	0x0	RW	Interrupt mapping to IRQ_CPU_6 or IRQ_CPU_7 bit[1], DMA1 interrupt mapping 0: IRQ_CPU_6 1: IRQ_CPU_7 bit[0], DMA0 interrupt mapping 0: IRQ_CPU_6 1: IRQ_CPU_7

Table 26 DMA_TO_THLD (DMA_BASE_ADDR+0x30)

Bit	Field Name	Reset	RW	Description
7:0	DMA_TO_THLD	0x0	RW	threshold of DMA0 and DMA1 timeout interrupt (N), time unit: 100us, so the time is N*100us

Table 27 DMA0_INTER_TX_CFG (DMA_BASE_ADDR+0x34)

Bit	Field Name	Reset	RW	Description
16	DMA0_INTER_TX_MODE	0	RW	DMA0 interval tx mode enable control 0: disable 1: enable Note: When in interval tx mode, the data will be transferred an offset value number (DMA0_TX_OFFSET). When a new offset value is set, DMA0 continues to work until the total number arrives at DMA0_CFG_CNT.
15:0	DMA0_TX_OFFSET	0x0	RW	DMA0 offset value of interval tx mode

Table 28 DMA1_INTER_TX_CFG (DMA_BASE_ADDR+0x38)

Bit	Field Name	Reset	RW	Description
16	DMA1_INTER_TX_MODE	0	RW	DMA1 interval tx mode enable control 0: disable 1: enable Note: When in interval tx mode, the data will be transferred an offset value number (DMA1_TX_OFFSET). When a new offset value is set, DMA1 continues to work until the total number arrives at DMA1_CFG_CNT.
15:0	DMA1_TX_OFFSET	0x0	RW	DMA1 offset value of interval tx mode

5.9 GPIO

5.9.1 Feature

1. Software defined GPIO input or output with internal configurable pull-up and pull-down
2. Configurable Schmitt trigger or CMOS input
3. All pins can be individually mapped to peripheral interface blocks for PCB layout flexibility
4. Interrupt trigger by state change on any pin
5. Wake-up trigger by level (high or low) or edge (positive, negative or both edge) on any pin

5.9.2 Function Description

The below figure shows a general overview. The design provides a flexible IO mux configuration, and most of the peripheral ports can be mapped to any of the physical I/O pads.

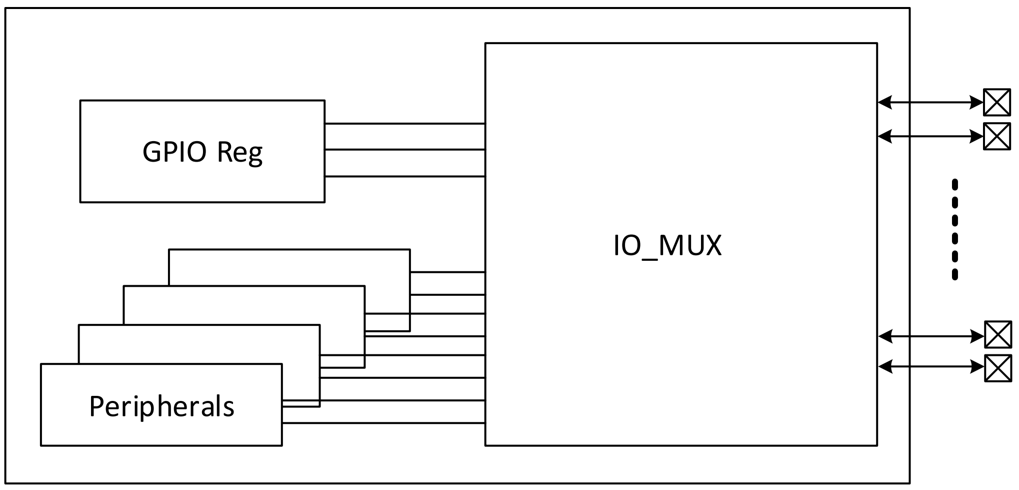


Figure 7 GPIO Overview

5.9.2.1 Pin Configuration

GPIO pad architecture is shown below.

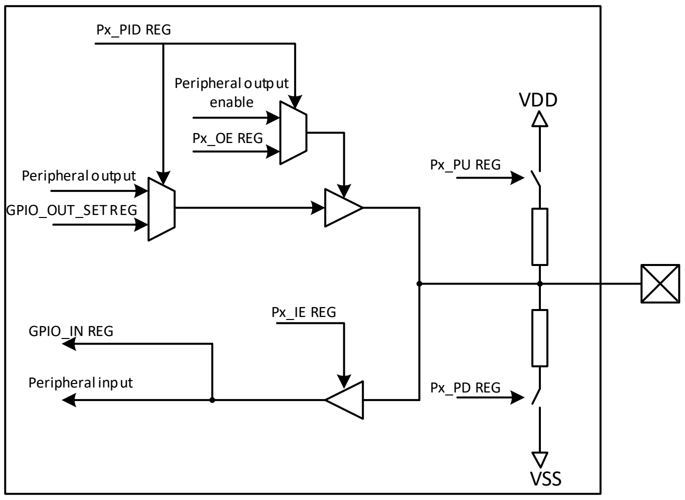


Figure 8 Pin Configuration

Each of these pins can be individually configured in the GPIO_Pxx_CONFIG registers (xx=0..24). The following parameters can be configured through these registers.

- Direction
- Drive strength
- Pull-up and pull-down

When the port is configured as an input, the data in Register (GPIO_IN) can be used to read the level of each pin in the port (bit n in the register is connected to pin n on the port). When configured as an output, the value of the data out Register (GPIO_OUT_SET) is driving the pin.

The output value can be changed in 2 different ways.

- Writing the GPIO_OUT_SET register sets the output bit
- Writing the GPIO_OUT_CLEAR register clears the output bit

Reading the GPIO_OUT_STATUS register will return its content.

The drive strength can be set to each pin individually. It's configured through GPIO_Pxx_DSH and GPIO_Pxx_DSL. The below table describes the driving strength definition.

Table 29 Driving Strength Degree table

GPIO_Pxx_DSH	GPIO_Pxx_DSL	Driving Strength Degree (OE=1)
0	0	5mA
0	1	10mA
1	0	15mA
1	1	20mA

In peripheral function mode, the registers GPIO_Pxx_PID need to be set for a flexible IO mapping.

5.9.2.2 Pin Mapping

The IO_MUX can map a number of peripheral modules such as GPIO, SPI, UART, I2C and I2S to any of the available I/Os. Each type of peripheral signal has a unique ID, and each IO can select one ID. To map a peripheral function to GPIOxx, where xx can range from 0 to a maximum of 24, the ID and pin configuration must be set in the corresponding bit GPIO_Pxx_PID in GPIO_Pxx_CONFIG register. The below table lists all the available peripheral IDs.

Table 30 Peripheral ID

ID	Port Description
0	Default GPIO usage
1	UART 0 TX pin
2	UART 0 RX pin
3	UART 0 CTS pin
4	UART 0 RTS pin
5	SPI CLOCK pin
6	SPI CSN pin
7	SPI MOSI pin
8	SPI MISO pin
9	I2C SCL pin
10	I2C SDA pin
11	PWM 0 pin
12	PWM 1 pin
13	PWM 2 pin
14	PWM 3 pin
15	7816 CLOCK pin
16	7816 DATA pin
17	7816 RESET pin
18	IR pin
19	I2S MCLK pin
20	I2S BCLK pin
21	I2S WCLK pin
22	I2S DATA pin
23	UART 1 TX pin
24	UART 1 RX pin
25	UART 1 CTS pin
26	UART 1 RTS pin
27	FLASH CLOCK pin
28	FLASH CS pin

29	FLASH SI pin
30	FLASH SO pin
31	FLASH WP pin
32	FLASH HOLD pin
33	PLL pin, output PLL clock 12M/16M/8M/4M/2M/1M.
34	QDEC PHASE A pin
35	QDEC PHASE B pin
36	QDEC LED pin
37~42	Reserved
43	TEST CLOCK pin, output clock RC32K/RC16M/XO16M. Only P21 can output test clock, other pins are reserved.
44~63	Reserved

Please note that:

- If a certain function (output pin like uart 0 txd) is selected on more than one port pin (like P00 and P01), and all ports (P00 and P01) will perform this function.
- If a certain function (input pin like uart 0 rxd) is selected on more than one port pin, the internal signal uart_rxd connected to UART module will perform OR operation of all ports.

5.9.2.3 Pin Wakeup

Any GPIO can wake up the chip from hibernate or sleep mode. As shown in the figure below, the wakeup request can be triggered through the pins by enabling the corresponding bit in the GPIO_PWT_EN register. When wakeup is enabled for the pin, the input filter (debounce function) can be optionally selected by GPIO_PWT_DEB register. The debouce circuit can avoid false wakeup caused by glitch. In addition, the polarity of the wakeup request can be selected using the GPIO_PWT_POL register.

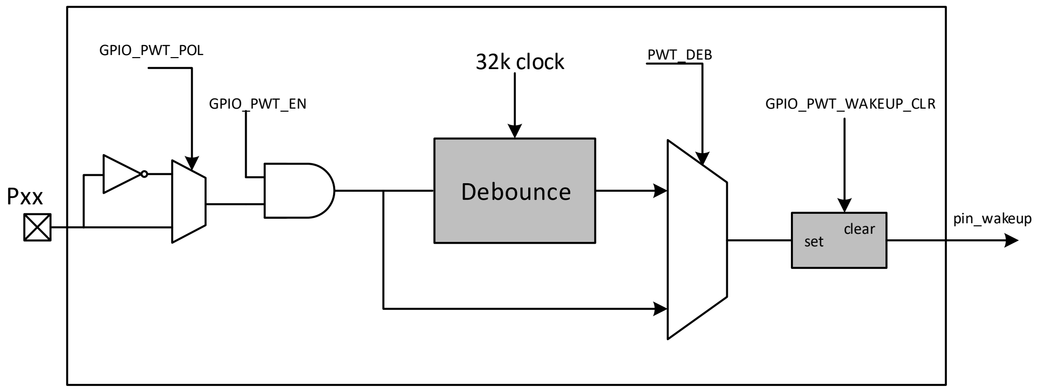


Figure 9 Pin Wakeup Logic

5.9.2.4 Interrupt Generation

The GPIO can generate an interrupt by level or edge. See the below figure Pin xx Interrupt Generation.

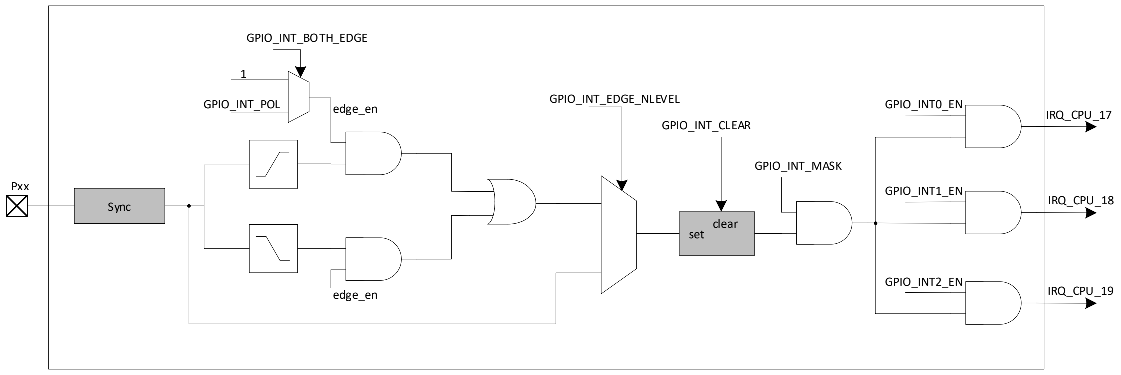


Figure 10 Pin xx Interrupt Generation

There are three GPIO interrupt groups (IRQ_CPU_17, IRQ_CPU_18, IRQ_CPU_19), and any pin can be grouped to any of the group for interrupt. The GPIO_INT0_EN, GPIO_INT1_EN and GPIO_INT2_EN registers select which ports in the group will trigger the interrupt.

For example, if GPIO_INT0_EN = 0xFF, then P00~P07 will be selected for generating interrupt IRQ_CPU_17. While GPIO_INT1_EN = 0xFF00, P08~P15 use the second interrupt group (IRQ_CPU_18).

In order to enable the level interrupt, set the GPIO_INT_EDGE_NLEVEL register to 0. Upon a level interrupt occurring, the corresponding interrupt polarity depends on the register GPIO_INT_POL.

The GPIO_INT_BOTH_EDGE and GPIO_INT_POL registers determines interrupt generation by rising or falling edge. If setting the GPIO_INT_BOTH_EDGE register, both rising and falling edges can generate interrupt. If the register GPIO_INT_BOTH_EDGE is '0', the register GPIO_INT_POL defines rising or falling edge.

5.9.3 Software Procedure

5.9.3.1 GPIO wakeup

The following sequence should be followed in GPIO wakeup flow.

Before hibernate/sleep.

1. Enable several GPIO as wakeup source
2. Set GPIO_PWT_DEB_CONFIG register bit GPIO_PWT_DEB. Select input filter (debounce) function.
3. Set GPIO_Pxx_CONFIG register. Configure wakeup source's IO mode to GPIO mode and set it to input mode.
4. Set GPIO_PWT_POL register. Configure wakeup source's polarity.
5. Set GPIO_PWT_EN register. Enable wakeup source IO.

6. Set GPIO_PWT_32K_LOAD register. Load registers (GPIO_PWT_DEB_CONFIG, GPIO_PWT_POL and GPIO_PWT_EN) to take effect in 32KHz working domain.
7. Go hibernate process.

Wakeup process.

1. One or more wakeup sources toggle for debounce time. Then internal signal io_pmu_wakeup toggles from '0' to '1'.
2. PMU goes to wakeup process.
3. After CPU wakeup, software set GPIO_PWT_EN register to '0'. After debounce time, the internal wakeup state will be cleared, but io_pmu_wakeup is still at high.
4. set GPIO_PWT_WAKEUP_CLR register to clear io_pmu_wakeup.
5. set GPIO_Pxx_CONFIG reg. The wakeup source can be used for normal function.

5.9.4 GPIO Interrupt

GPIO interrupts are described in register GPIO_INT_STATUS.

5.9.5 GPIO Registers

Table 31 GPIO Instance

Base Address	Peripheral	Instance	Description
0x40001000	GPIO	GPIO	General Purpose Input/Output

Table 32 GPIO Register Map

Address Offset	Name	Description
0x00	GPIO_INT_STATUS	GPIO Interrupt Status
0x04	GPIO_INT_MASK	GPIO Interrupt Enable
0x08	GPIO_INT_CLEAR	GPIO Interrupt Flag Clear Register
0x14	GPIO_OUT_STATUS	GPIO Output Driver Value
0x18	GPIO_OUT_SET	GPIO Output Driver Value Setting
0x1C	GPIO_OUT_CLEAR	GPIO Output Driver Value Clear
0x20	GPIO_INT_POL	GPIO Interrupt Polarity
0x24	GPIO_INT_EDGE_NLEVEL	GPIO Interrupt Mode
0x28	GPIO_INT_BOTH_EDGE	GPIO Interrupt Edge Mode
0x2C	GPIO_INT0_EN	GPIO Interrupt Connected to IRQ_GPIO_17
0x30	GPIO_INT1_EN	GPIO Interrupt Connected to IRQ_GPIO_18
0x34	GPIO_INT2_EN	GPIO Interrupt Connected to IRQ_GPIO_19
0x38	GPIO_IN	GPIO Input Value Register
0x3C	GPIO_P00_CONFIG	GPIO P00 Configuration Register
0x40	GPIO_P01_CONFIG	GPIO P01 Configuration Register

0x44	GPIO_P02_CONFIG	GPIO P02 Configuration Register
0x48	GPIO_P03_CONFIG	GPIO P03 Configuration Register
0x4C	GPIO_P04_CONFIG	GPIO P04 Configuration Register
0x50	GPIO_P05_CONFIG	GPIO P05 Configuration Register
0x54	GPIO_P06_CONFIG	GPIO P06 Configuration Register
0x58	GPIO_P07_CONFIG	GPIO P07 Configuration Register
0x5C	GPIO_P08_CONFIG	GPIO P08 Configuration Register
0x60	GPIO_P09_CONFIG	GPIO P09 Configuration Register
0x64	GPIO_P10_CONFIG	GPIO P10 Configuration Register
0x68	GPIO_P11_CONFIG	GPIO P11 Configuration Register
0x6C	GPIO_P12_CONFIG	GPIO P12 Configuration Register
0x70	GPIO_P13_CONFIG	GPIO P13 Configuration Register
0x74	GPIO_P14_CONFIG	GPIO P14 Configuration Register
0x78	GPIO_P15_CONFIG	GPIO P15 Configuration Register
0x7C	GPIO_P16_CONFIG	GPIO P16 Configuration Register
0x80	GPIO_P17_CONFIG	GPIO P17 Configuration Register
0x84	GPIO_P18_CONFIG	GPIO P18 Configuration Register
0x88	GPIO_P19_CONFIG	GPIO P19 Configuration Register
0x8C	GPIO_P20_CONFIG	GPIO P20 Configuration Register
0x90	GPIO_P21_CONFIG	GPIO P21 Configuration Register
0x94	GPIO_P22_CONFIG	GPIO P22 Configuration Register
0x98	GPIO_P23_CONFIG	GPIO P23 Configuration Register
0x9C	GPIO_P24_CONFIG	GPIO P24 Configuration Register
0xA0	GPIO_PWT_32K_LOAD	GPIO Load Enable
0xA4	GPIO_PWT_DEB_CONFIG	GPIO Debounce Configuration Register
0xA8	GPIO_PWT_POL	GPIO Wakeup Polarity
0xAC	GPIO_PWT_EN	GPIO Wakeup Enable
0xB0	GPIO_PWT_WAKEUP_CLR	GPIO Wakeup Flag Clear Register
0xB4	GPIO_IOMUX_DEBUG_CTRL	GPIO Debug Mode Configuration Register
0xDC	GPIO_SW_SET	Serial Wire Output Value Setting
0xE0	GPIO_SW_CLR	Serial Wire Output Value Clear
0xB0	GPIO_SWCLK_CONFIG	SWCLK Configuration Register
0xB4	GPIO_SWDIO_CONFIG	SWDIO Configuration Register
0xEC	GPIO_SW_IN	Serial Wire Input Value

Table 33 GPIO_INT_STATUS (GPIO_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_INT_STATUS	0x0	R	GPIO0-GPIO24 interrupt status 0: invalid 1: valid

Table 34 GPIO_INT_MASK (GPIO_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_INT_MASK	0x0	RW	GPIO0-GPIO24 interrupt enable 0: disable 1: enable

Table 35 GPIO_INT_CLEAR (GPIO_BASE_ADDR+0x08)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_INT_CLEAR	0x0	W	GPIO0-GPIO24 interrupt flag clear 0: invalid 1: clear

Table 36 GPIO_OUT_STATUS (GPIO_BASE_ADDR+0x14)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_OUT_STATUS	0x0	R	GPIO0-GPIO24 output driver value 0: low 1: high

Table 37 GPIO_OUT_SET (GPIO_BASE_ADDR+0x18)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_OUT_SET	0x0	W	GPIO0-GPIO24 output driver value setting 0: invalid 1: high

Table 38 GPIO_OUT_CLEAR (GPIO_BASE_ADDR+0x1C)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_OUT_CLEAR	0x0	W	GPIO0-GPIO24 output driver value clear 0: invalid 1: output driver value to low

Table 39 GPIO_INT_POL (GPIO_BASE_ADDR+0x20)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_INT_POL	0x0	RW	GPIO0-GPIO24 interrupt polarity when GPIO_INT_EDGE_NLEVEL is high and GPIO_INT_BOTH_EDGE is low 0: rising edge generates interrupt 1: falling edge generates interrupt when GPIO_INT_EDGE_NLEVEL is low 0: low level generates interrupt 1: high level generates interrupt

Table 40 GPIO_INT_EDGE_NLEVEL (GPIO_BASE_ADDR+0x24)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_INT_EDGE_NLEVEL	0x0	RW	GPIO0-GPIO24 interrupt mode 0: level trigger mode 1: edge trigger mode

Table 41 GPIO_INT_BOTH_EDGE (GPIO_BASE_ADDR+0x28)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_INT_BOTH_EDGE	0x0	RW	single or both edge mode for GPIO0-GPIO24 interrupt edge trigger 0: single edge trigger. When GPIO_INT_POL is high, falling edge trigger generate interrupt. When GPIO_INT_POL is low, rising edge trigger generate interrupt. 1: both edge trigger

Table 42 GPIO_INT0_EN (GPIO_BASE_ADDR+0x2C)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_INT0_EN	0x0	RW	GPIO0-GPIO24 selection for IRQ_GPIO_17 0: not selected 1: selected

Table 43 GPIO_INT1_EN (GPIO_BASE_ADDR+0x30)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_INT1_EN	0x0	RW	GPIO0-GPIO24 selection for IRQ_GPIO_18 0: not selected 1: selected

Table 44 GPIO_INT2_EN (GPIO_BASE_ADDR+0x34)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_INT2_EN	0x0	RW	GPIO0-GPIO24 selection for IRQ_GPIO_19 0: not selected 1: selected

Table 45 GPIO_IN (GPIO_BASE_ADDR+0x38)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_IN	0x0	R	GPIO0-GPIO24 input value, GPIO_P(00-24)_IE should be set high before reading GPIO_IN 0: input value is low 1: input value is high

Table 46 GPIO_P00_CONFIG (GPIO_BASE_ADDR+0x3C)

Bit	Field Name	Reset	RW	Description

12	GPIO_P00_CS	0	RW	Schmitt trigger mode for GPIO input buffer 0: no Schmitt trigger 1: Schmitt trigger
11	GPIO_P00_DSH	0	RW	driving strength high bit {dsh,dsl}: 00: lowest driving strength 01: lower driving strength 10: higher driving strength 11: highest driving strength
10	GPIO_P00_DSL	0	RW	driving strength low bit {dsh,dsl}: 00: lowest driving strength 01: lower driving strength 10: higher driving strength 11: highest driving strength
9	GPIO_P00_PU	1	RW	pull up enable control 0: disable 1: enable
8	GPIO_P00_PD	0	RW	pull down enable control 0: disable 1: enable
7	GPIO_P00_IE	1	RW	input enable control 0: disable 1: enable
6	GPIO_P00_OE	0	RW	output enable control 0: disable 1: enable
5:0	GPIO_P00_PID	0x0	RW	function of port 0: GPIO 1: uart 0 txd 2: uart 0 rxd 3: uart 0 cts 4: uart 0 rts 5: spi 0 clk 6: spi 0 csn 7: spi 0 mosi 8: spi 0 miso 9: i2c 0 scl 10: i2c 0 sda 11: pwm 0 12: pwm 1 13: pwm 2

				14: pwm 3 15: 7816 clk 16: 7816 data 17: 7816 rst_n 18: ir 19: i2s mclk 20: i2s bclk 21: i2s wclk 22: i2s data 23: uart 1 txd 24: uart 1 rxd 25: uart 1 cts 26: uart 1 rts 27: flash clk 28: flash cs 29: flash si 30: flash so 31: flash wp 32: flash hold 33: PLL12M/16M/8M/4M/2M/1M 34: qdec phase_a in 35: qdec phase_b in 36: qdec led_out 37~63: reserved

Table 47 GPIO_P01_CONFIG (GPIO_BASE_ADDR+0x40)

Bit	Field Name	Reset	RW	Description
12	GPIO_P01_CS	0	RW	see GPIO_P00
11	GPIO_P01_DSH	0	RW	see GPIO_P00
10	GPIO_P01_DSL	0	RW	see GPIO_P00
9	GPIO_P01_PU	1	RW	see GPIO_P00
8	GPIO_P01_PD	0	RW	see GPIO_P00
7	GPIO_P01_IE	1	RW	see GPIO_P00
6	GPIO_P01_OE	0	RW	see GPIO_P00
5:0	GPIO_P01_PID	0x0	RW	see GPIO_P00

Table 48 GPIO_P02_CONFIG (GPIO_BASE_ADDR+0x44)

Bit	Field Name	Reset	RW	Description
12	GPIO_P02_CS	0	RW	see GPIO_P00
11	GPIO_P02_DSH	0	RW	see GPIO_P00
10	GPIO_P02_DSL	0	RW	see GPIO_P00
9	GPIO_P02_PU	1	RW	see GPIO_P00

8	GPIO_P02_PD	0	RW	see GPIO_P00
7	GPIO_P02_IE	1	RW	see GPIO_P00
6	GPIO_P02_OE	0	RW	see GPIO_P00
5:0	GPIO_P02_PID	0x0	RW	see GPIO_P00

Table 49 GPIO_P03_CONFIG (GPIO_BASE_ADDR+0x48)

Bit	Field Name	Reset	RW	Description
12	GPIO_P03_CS	0	RW	see GPIO_P00
11	GPIO_P03_DSH	0	RW	see GPIO_P00
10	GPIO_P03_DSL	0	RW	see GPIO_P00
9	GPIO_P03_PU	1	RW	see GPIO_P00
8	GPIO_P03_PD	0	RW	see GPIO_P00
7	GPIO_P03_IE	1	RW	see GPIO_P00
6	GPIO_P03_OE	0	RW	see GPIO_P00
5:0	GPIO_P03_PID	0x0	RW	see GPIO_P00

Table 50 GPIO_P04_CONFIG (GPIO_BASE_ADDR+0x4C)

Bit	Field Name	Reset	RW	Description
12	GPIO_P04_CS	0	RW	see GPIO_P00
11	GPIO_P04_DSH	0	RW	see GPIO_P00
10	GPIO_P04_DSL	0	RW	see GPIO_P00
9	GPIO_P04_PU	1	RW	see GPIO_P00
8	GPIO_P04_PD	0	RW	see GPIO_P00
7	GPIO_P04_IE	1	RW	see GPIO_P00
6	GPIO_P04_OE	0	RW	see GPIO_P00
5:0	GPIO_P04_PID	0x0	RW	see GPIO_P00

Table 51 GPIO_P05_CONFIG (GPIO_BASE_ADDR+0x50)

Bit	Field Name	Reset	RW	Description
12	GPIO_P05_CS	0	RW	see GPIO_P00
11	GPIO_P05_DSH	0	RW	see GPIO_P00
10	GPIO_P05_DSL	0	RW	see GPIO_P00
9	GPIO_P05_PU	1	RW	see GPIO_P00
8	GPIO_P05_PD	0	RW	see GPIO_P00
7	GPIO_P05_IE	1	RW	see GPIO_P00
6	GPIO_P05_OE	0	RW	see GPIO_P00
5:0	GPIO_P05_PID	0x0	RW	see GPIO_P00

Table 52 GPIO_P06_CONFIG (GPIO_BASE_ADDR+0x54)

Bit	Field Name	Reset	RW	Description
12	GPIO_P06_CS	0	RW	see GPIO_P00

11	GPIO_P06_DSH	0	RW	see GPIO_P00
10	GPIO_P06_DSL	0	RW	see GPIO_P00
9	GPIO_P06_PU	1	RW	see GPIO_P00
8	GPIO_P06_PD	0	RW	see GPIO_P00
7	GPIO_P06_IE	1	RW	see GPIO_P00
6	GPIO_P06_OE	0	RW	see GPIO_P00
5:0	GPIO_P06_PID	0x0	RW	see GPIO_P00

Table 53 GPIO_P07_CONFIG (GPIO_BASE_ADDR+0x58)

Bit	Field Name	Reset	RW	Description
12	GPIO_P07_CS	0	RW	see GPIO_P00
11	GPIO_P07_DSH	0	RW	see GPIO_P00
10	GPIO_P07_DSL	0	RW	see GPIO_P00
9	GPIO_P07_PU	1	RW	see GPIO_P00
8	GPIO_P07_PD	0	RW	see GPIO_P00
7	GPIO_P07_IE	1	RW	see GPIO_P00
6	GPIO_P07_OE	0	RW	see GPIO_P00
5:0	GPIO_P07_PID	0x0	RW	see GPIO_P00

Table 54 GPIO_P08_CONFIG (GPIO_BASE_ADDR+0x5C)

Bit	Field Name	Reset	RW	Description
12	GPIO_P08_CS	0	RW	see GPIO_P00
11	GPIO_P08_DSH	0	RW	see GPIO_P00
10	GPIO_P08_DSL	0	RW	see GPIO_P00
9	GPIO_P08_PU	1	RW	see GPIO_P00
8	GPIO_P08_PD	0	RW	see GPIO_P00
7	GPIO_P08_IE	1	RW	see GPIO_P00
6	GPIO_P08_OE	0	RW	see GPIO_P00
5:0	GPIO_P08_PID	0x0	RW	see GPIO_P00

Table 55 GPIO_P09_CONFIG (GPIO_BASE_ADDR+0x60)

Bit	Field Name	Reset	RW	Description
12	GPIO_P09_CS	0	RW	see GPIO_P00
11	GPIO_P09_DSH	0	RW	see GPIO_P00
10	GPIO_P09_DSL	0	RW	see GPIO_P00
9	GPIO_P09_PU	1	RW	see GPIO_P00
8	GPIO_P09_PD	0	RW	see GPIO_P00
7	GPIO_P09_IE	1	RW	see GPIO_P00
6	GPIO_P09_OE	0	RW	see GPIO_P00
5:0	GPIO_P09_PID	0x0	RW	see GPIO_P00

Table 56 GPIO_P10_CONFIG (GPIO_BASE_ADDR+0x64)

Bit	Field Name	Reset	RW	Description
12	GPIO_P10_CS	0	RW	see GPIO_P00
11	GPIO_P10_DSH	0	RW	see GPIO_P00
10	GPIO_P10_DSL	0	RW	see GPIO_P00
9	GPIO_P10_PU	1	RW	see GPIO_P00
8	GPIO_P10_PD	0	RW	see GPIO_P00
7	GPIO_P10_IE	1	RW	see GPIO_P00
6	GPIO_P10_OE	0	RW	see GPIO_P00
5:0	GPIO_P10_PID	0x0	RW	see GPIO_P00

Table 57 GPIO_P11_CONFIG (GPIO_BASE_ADDR+0x68)

Bit	Field Name	Reset	RW	Description
12	GPIO_P11_CS	0	RW	see GPIO_P00
11	GPIO_P11_DSH	0	RW	see GPIO_P00
10	GPIO_P11_DSL	0	RW	see GPIO_P00
9	GPIO_P11_PU	1	RW	see GPIO_P00
8	GPIO_P11_PD	0	RW	see GPIO_P00
7	GPIO_P11_IE	1	RW	see GPIO_P00
6	GPIO_P11_OE	0	RW	see GPIO_P00
5:0	GPIO_P11_PID	0x0	RW	see GPIO_P00

Table 58 GPIO_P12_CONFIG (GPIO_BASE_ADDR+0x6C)

Bit	Field Name	Reset	RW	Description
12	GPIO_P12_CS	0	RW	see GPIO_P00
11	GPIO_P12_DSH	0	RW	see GPIO_P00
10	GPIO_P12_DSL	0	RW	see GPIO_P00
9	GPIO_P12_PU	1	RW	see GPIO_P00
8	GPIO_P12_PD	0	RW	see GPIO_P00
7	GPIO_P12_IE	1	RW	see GPIO_P00
6	GPIO_P12_OE	0	RW	see GPIO_P00
5:0	GPIO_P12_PID	0x0	RW	see GPIO_P00

Table 59 GPIO_P13_CONFIG (GPIO_BASE_ADDR+0x70)

Bit	Field Name	Reset	RW	Description
12	GPIO_P13_CS	0	RW	see GPIO_P00
11	GPIO_P13_DSH	0	RW	see GPIO_P00
10	GPIO_P13_DSL	0	RW	see GPIO_P00
9	GPIO_P13_PU	1	RW	see GPIO_P00
8	GPIO_P13_PD	0	RW	see GPIO_P00
7	GPIO_P13_IE	1	RW	see GPIO_P00

6	GPIO_P13_OE	0	RW	see GPIO_P00
5:0	GPIO_P13_PID	0x0	RW	see GPIO_P00

Table 60 GPIO_P14_CONFIG (GPIO_BASE_ADDR+0x74)

Bit	Field Name	Reset	RW	Description
12	GPIO_P14_CS	0	RW	see GPIO_P00
11	GPIO_P14_DSH	0	RW	see GPIO_P00
10	GPIO_P14_DSL	0	RW	see GPIO_P00
9	GPIO_P14_PU	1	RW	see GPIO_P00
8	GPIO_P14_PD	0	RW	see GPIO_P00
7	GPIO_P14_IE	1	RW	see GPIO_P00
6	GPIO_P14_OE	0	RW	see GPIO_P00
5:0	GPIO_P14_PID	0x0	RW	see GPIO_P00

Table 61 GPIO_P15_CONFIG (GPIO_BASE_ADDR+0x78)

Bit	Field Name	Reset	RW	Description
12	GPIO_P15_CS	0	RW	see GPIO_P00
11	GPIO_P15_DSH	0	RW	see GPIO_P00
10	GPIO_P15_DSL	0	RW	see GPIO_P00
9	GPIO_P15_PU	1	RW	see GPIO_P00
8	GPIO_P15_PD	0	RW	see GPIO_P00
7	GPIO_P15_IE	1	RW	see GPIO_P00
6	GPIO_P15_OE	0	RW	see GPIO_P00
5:0	GPIO_P15_PID	0x0	RW	see GPIO_P00

Table 62 GPIO_P16_CONFIG (GPIO_BASE_ADDR+0x7C)

Bit	Field Name	Reset	RW	Description
12	GPIO_P16_CS	0	RW	see GPIO_P00
11	GPIO_P16_DSH	0	RW	see GPIO_P00
10	GPIO_P16_DSL	0	RW	see GPIO_P00
9	GPIO_P16_PU	1	RW	see GPIO_P00
8	GPIO_P16_PD	0	RW	see GPIO_P00
7	GPIO_P16_IE	1	RW	see GPIO_P00
6	GPIO_P16_OE	0	RW	see GPIO_P00
5:0	GPIO_P16_PID	0x0	RW	see GPIO_P00

Table 63 GPIO_P17_CONFIG (GPIO_BASE_ADDR+0x80)

Bit	Field Name	Reset	RW	Description
12	GPIO_P17_CS	0	RW	see GPIO_P00
11	GPIO_P17_DSH	0	RW	see GPIO_P00
10	GPIO_P17_DSL	0	RW	see GPIO_P00

9	GPIO_P17_PU	1	RW	see GPIO_P00
8	GPIO_P17_PD	0	RW	see GPIO_P00
7	GPIO_P17_IE	1	RW	see GPIO_P00
6	GPIO_P17_OE	0	RW	see GPIO_P00
5:0	GPIO_P17_PID	0x0	RW	see GPIO_P00

Table 64 GPIO_P18_CONFIG (GPIO_BASE_ADDR+0x84)

Bit	Field Name	Reset	RW	Description
12	GPIO_P18_CS	0	RW	see GPIO_P00
11	GPIO_P18_DSH	0	RW	see GPIO_P00
10	GPIO_P18_DSL	0	RW	see GPIO_P00
9	GPIO_P18_PU	1	RW	see GPIO_P00
8	GPIO_P18_PD	0	RW	see GPIO_P00
7	GPIO_P18_IE	1	RW	see GPIO_P00
6	GPIO_P18_OE	0	RW	see GPIO_P00
5:0	GPIO_P18_PID	0x0	RW	see GPIO_P00

Table 65 GPIO_P19_CONFIG (GPIO_BASE_ADDR+0x88)

Bit	Field Name	Reset	RW	Description
12	GPIO_P19_CS	0	RW	see GPIO_P00
11	GPIO_P19_DSH	0	RW	see GPIO_P00
10	GPIO_P19_DSL	0	RW	see GPIO_P00
9	GPIO_P19_PU	1	RW	see GPIO_P00
8	GPIO_P19_PD	0	RW	see GPIO_P00
7	GPIO_P19_IE	1	RW	see GPIO_P00
6	GPIO_P19_OE	0	RW	see GPIO_P00
5:0	GPIO_P19_PID	0x0	RW	see GPIO_P00

Table 66 GPIO_P20_CONFIG (GPIO_BASE_ADDR+0x8C)

Bit	Field Name	Reset	RW	Description
12	GPIO_P20_CS	0	RW	see GPIO_P00
11	GPIO_P20_DSH	0	RW	see GPIO_P00
10	GPIO_P20_DSL	0	RW	see GPIO_P00
9	GPIO_P20_PU	1	RW	see GPIO_P00
8	GPIO_P20_PD	0	RW	see GPIO_P00
7	GPIO_P20_IE	1	RW	see GPIO_P00
6	GPIO_P20_OE	0	RW	see GPIO_P00
5:0	GPIO_P20_PID	0x0	RW	see GPIO_P00

Table 67 GPIO_P21_CONFIG (GPIO_BASE_ADDR+0x90)

Bit	Field Name	Reset	RW	Description

12	GPIO_P21_CS	0	RW	see GPIO_P00
11	GPIO_P21_DSH	0	RW	see GPIO_P00
10	GPIO_P21_DSL	0	RW	see GPIO_P00
9	GPIO_P21_PU	1	RW	see GPIO_P00
8	GPIO_P21_PD	0	RW	see GPIO_P00
7	GPIO_P21_IE	1	RW	see GPIO_P00
6	GPIO_P21_OE	0	RW	see GPIO_P00
5:0	GPIO_P21_PID	0x0	RW	function of port 0: GPIO 1: uart 0 txd 2: uart 0 rxd 3: uart 0 cts 4: uart 0 rts 5: spi 0 clk 6: spi 0 csn 7: spi 0 mosi 8: spi 0 miso 9: i2c 0 scl 10: i2c 0 sda 11: pwm 0 12: pwm 1 13: pwm 2 14: pwm 3 15: 7816 clk 16: 7816 data 17: 7816 rst_n 18: ir 19: i2s mclk 20: i2s bclk 21: i2s wclk 22: i2s data 23: uart 1 txd 24: uart 1 rxd 25: uart 1 cts 26: uart 1 rts 27: flash clk 28: flash cs 29: flash si 30: flash so 31: flash wp 32: flash hold 33: PLL12M/16M/8M/4M/2M/1M

				34: qdec phase_a in 35: qdec phase_b in 36: qdec led_out 37~42: reserved 43: test clock 44~63: reserved

Table 68 GPIO_P22_CONFIG (GPIO_BASE_ADDR+0x94)

Bit	Field Name	Reset	RW	Description
12	GPIO_P22_CS	0	RW	see GPIO_P00
11	GPIO_P22_DSH	0	RW	see GPIO_P00
10	GPIO_P22_DSL	0	RW	see GPIO_P00
9	GPIO_P22_PU	1	RW	see GPIO_P00
8	GPIO_P22_PD	0	RW	see GPIO_P00
7	GPIO_P22_IE	1	RW	see GPIO_P00
6	GPIO_P22_OE	0	RW	see GPIO_P00
5:0	GPIO_P22_PID	0x0	RW	see GPIO_P00

Table 69 GPIO_P23_CONFIG (GPIO_BASE_ADDR+0x98)

Bit	Field Name	Reset	RW	Description
12	GPIO_P23_CS	0	RW	see GPIO_P00
11	GPIO_P23_DSH	0	RW	see GPIO_P00
10	GPIO_P23_DSL	0	RW	see GPIO_P00
9	GPIO_P23_PU	1	RW	see GPIO_P00
8	GPIO_P23_PD	0	RW	see GPIO_P00
7	GPIO_P23_IE	1	RW	see GPIO_P00
6	GPIO_P23_OE	0	RW	see GPIO_P00
5:0	GPIO_P23_PID	0x0	RW	see GPIO_P00

Table 70 GPIO_P24_CONFIG (GPIO_BASE_ADDR+0x9C)

Bit	Field Name	Reset	RW	Description
12	GPIO_P24_CS	0	RW	see GPIO_P00
11	GPIO_P24_DSH	0	RW	see GPIO_P00
10	GPIO_P24_DSL	0	RW	see GPIO_P00
9	GPIO_P24_PU	1	RW	see GPIO_P00
8	GPIO_P24_PD	0	RW	see GPIO_P00
7	GPIO_P24_IE	1	RW	see GPIO_P00
6	GPIO_P24_OE	0	RW	see GPIO_P00
5:0	GPIO_P24_PID	0x0	RW	see GPIO_P00

Table 71 GPIO_PWT_32K_LOAD (GPIO_BASE_ADDR+0xA0)

Bit	Field Name	Reset	RW	Description
0	GPIO_PWT_32K_LOAD	0	W	Load event for registers (GPIO_PWT_DEB_CONFIG, GPIO_PWT_POL and GPIO_PWT_EN) to take effect in 32KHz clock working domain 0: invalid 1: load enable

Table 72 GPIO_PWT_DEB_CONFIG (GPIO_BASE_ADDR+0xA4)

Bit	Field Name	Reset	RW	Description
6	GPIO_PWT_MS_N32K	0	RW	pin wakeup debounce time unit 0: 30us 1: 1ms
5:0	GPIO_PWT_DEB	0x0	RW	pin wakeup debounce time counter (N) 0: no debounce 1~63: debounce time is N*PWT_MS_N32K Note: debounce function can improve pin wake up robustness

Table 73 GPIO_PWT_POL (GPIO_BASE_ADDR+0xA8)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_PWT_POL	0x0	RW	GPIO0-GPIO24 pin wakeup level polarity 0: low level wakeup 1: high level wakeup

Table 74 GPIO_PWT_EN (GPIO_BASE_ADDR+0xAC)

Bit	Field Name	Reset	RW	Description
24:0	GPIO_PWT_EN	0x0	RW	GPIO0-GPIO24 pin wakeup enable control 0: disable 1: enable

Table 75 GPIO_PWT_WAKEUP_CLR (GPIO_BASE_ADDR+0xB0)

Bit	Field Name	Reset	RW	Description
0	GPIO_PWT_WAKEUP_CLR	0	W	pin wakeup (debounce mode) source clear 0: invalid 1: clear

Table 76 GPIO_IOMUX_DEBUG_CTRL (GPIO_BASE_ADDR+0xB4)

Bit	Field Name	Reset	RW	Description
2	GPIO_IOMUX_RX_EN	0	RW	rx or tx selection for Bluetooth debug mode 0: tx mode, GPIO inputs GFSK data 1: rx mode, GPIO outputs ADC data

1	GPIO_IOMUX_MANUAL	0	RW	rx mode enable selection in debug mode 0: rx mode enable selection from trxs modu le 1: rx mode enable selection from registers
0	GPIO_IOMUX_DEBUG	0	RW	GPIO debug mode 0: GPIO for normal function mode 1: GPIO for Bluetooth debug mode

Table 77 GPIO_SW_SET (GPIO_BASE_ADDR+0xDC)

Bit	Field Name	Reset	RW	Description
1	GPIO_SWDIO_SET	0	W	swdio output driver value setting 0: invalid 1: high
0	GPIO_SWCLK_SET	0	W	swclk output driver value setting 0: invalid 1: high

Table 78 GPIO_SW_CLR (GPIO_BASE_ADDR+0xE0)

Bit	Field Name	Reset	RW	Description
1	GPIO_SWDIO_CLR	0	W	swdio output driver value clear 0: invalid 1: output driver value to low
0	GPIO_SWCLK_CLR	0	W	swclk output driver value clear 0: invalid 1: output driver value to low

Table 79 GPIO_SWCLK_CONFIG (GPIO_BASE_ADDR+0xE4)

Bit	Field Name	Reset	RW	Description
6	GPIO_SWCLK_CS	0	RW	see GPIO_P00
5	GPIO_SWCLK_DSH	0	RW	see GPIO_P00
4	GPIO_SWCLK_DSL	0	RW	see GPIO_P00
3	GPIO_SWCLK_PU	0	RW	see GPIO_P00
2	GPIO_SWCLK_PD	1	RW	see GPIO_P00
1	GPIO_SWCLK_IE	1	RW	see GPIO_P00
0	GPIO_SWCLK_OE	0	RW	see GPIO_P00

Table 80 GPIO_SWDIO_CONFIG (GPIO_BASE_ADDR+0xE8)

Bit	Field Name	Reset	RW	Description
6	GPIO_SWDIO_CS	0	RW	see GPIO_P00
5	GPIO_SWDIO_DSH	0	RW	see GPIO_P00
4	GPIO_SWDIO_DSL	0	RW	see GPIO_P00

3	GPIO_SWDIO_PU	0	RW	see GPIO_P00
2	GPIO_SWDIO_PD	1	RW	see GPIO_P00
1	GPIO_SWDIO_IE	1	RW	see GPIO_P00
0	GPIO_SWDIO_OE	0	RW	see GPIO_P00

Table 81 GPIO_SW_IN (GPIO_BASE_ADDR+0xEC)

Bit	Field Name	Reset	RW	Description
1	GPIO_SWDIO_IN	0	R	swdio input value, GPIO_SWDIO_IE should be set high before reading 0: input value is low 1: input value is high
0	GPIO_SWCLK_IN	0	R	swclk input value, GPIO_SWCLK_IE should be set high before reading 0: input value is low 1: input value is high

5.10 General Purpose ADC

5.10.1 Feature

The GPADC (General Purpose Analog-to-Digital Converter) is a 9-bit successive approximation type ADC. The ADC has its voltage regulator (LDO) of 1.2V, which represents the full scale reference voltage. The main features are:

1. 9-bit dynamic ADC with configured clock frequency (up to 16MHz)
2. Four single-ended external GPIO input channels
3. Battery monitoring function
4. On-chip temperature sensor function
5. Digital down-sample with configurable ratio

5.10.2 Function Description

The chip has a multiplexer between the ADC and four specific GPIO ports (GPIO7, GPIO8, GPIO10, GPIO11), and the GPIO input voltage range is 0~1.2 V. Furthermore, the ADC can be used to measure the battery voltage (VDDR) and the internal temperature. Only one function can be selected at the same time.
Down-sample filtered data is stored in the register, which can be accessed through APB interface.

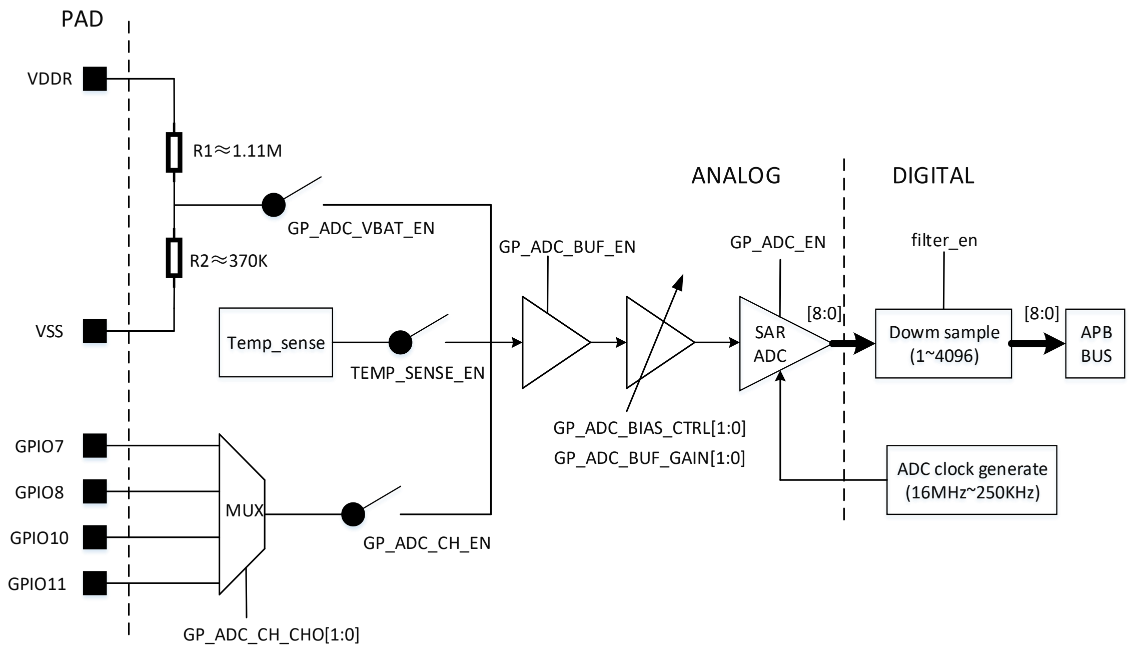


Figure 11 General Purpose ADC Block Diagram

Enabling/disabling of the ADC is triggered by configuring bit GP_ADC_EN (it enables ADC LDO at the same time), and a settling time of 50 us is required before an AD-conversion can be started.

The conversion itself is fast and takes approximately one clock cycle of 16 MHz (if ADC clock selected 16MHz), though the data handling will require several additional clock cycles, depending on the software code style. The fastest code can handle the data in four clock cycles of 16 MHz, resulting to a highest sampling rate of 16 MHz/5 = 3.2 MSamples/s.

The ADC result also has some noises. The down-sample filter corrects ADC result (set bit FILTER_EN to 1). The filter takes more samples and the calculated average value reduces the noise and increases the resolution. The down-sample ratio can be configured to 2~4096.

The Temperature ADC data vs Temperature range is show in the figure below.

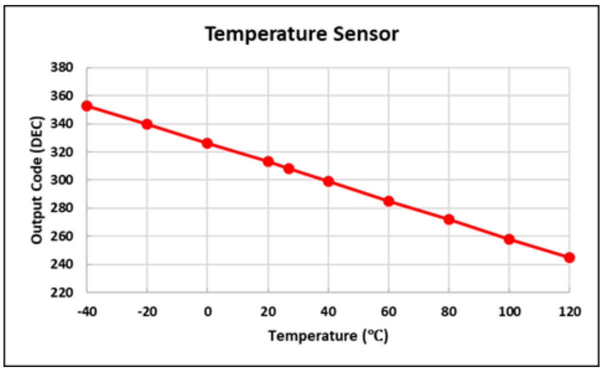


Figure 12 ADC data vs Temperature

5.10.3 Software Procedure

The GPADC read procedure.

1. GPADC enable (set GP_ADC_EN) and wait 50us for ADC to stabilize
2. Restart GPADC module (set CLR_GPADC)
3. Select source to GPADC (GPIOs, Battery or Temperature)
4. GPADC buffer enable (set GP_ADC_BUF_EN)
5. If ADC_OK is 1 and GPADC_BREAK is 0, the data is valid. Read ADC data.

5.10.4 Registers

Table 82 GPADC Instance

Base Address	Peripheral	Instance	Description
0x40017000	GPADC	GPADC	General Purpose ADC
0x4001e000	MISC	MISC	Miscellaneous

Table 83 GPADC Register Map

Address Offset	Name	Description
0x04	GPADC_CFG1	GPADC configure register 1
0x08	GPADC_SAD	GPADC state and data
0x0c	GPADC_DET_TH	GPADC data overflow threshold
0x10	GPADC_CLR	GPADC clears register
0x14	GPADC_CTRL	GPADC control register

Table 84 GPADC_CFG1 (GPADC_BASE_ADDR +0x4)

Bit	Field Name	Reset	RW	Description
10	GP_ADC_VBAT_EN	0	RW	VDDR measurement selection 0 : VDDR not selected 1 : VDDR selected
9:7	GP_ADC_CH_CHO	0	RW	GPIOs channel selection 0 : select GPIO_7 1 : select GPIO_8 2 : select GPIO_10 3 : select GPIO_11 4~7 : reserved
6	GP_ADC_CH_EN	0	RW	GPIOs measurement selection 0 : GPIOs not selected 1 : GPIOs selected
5:4	GP_ADC_BIAS_CTRL	0	RW	Bias current selection , for debug
3:2	GP_ADC_BUF_GAIN	0	RW	Buffer gain

				0 : 6dB 1 : 9dB 2 : 0dB 3 : 12dB
1	GP_ADC_BUF_EN	0	RW	GPADC Buffer enable 0 : disable 1 : enable
0	GP_ADC_EN	0	RW	GPADC enable 0 : disable 1 : enable

Table 85 GPADC_SAD (GPADC_BASE_ADDR +0x8)

Bit	Field Name	Reset	RW	Description
31	ADC_OK	0	R	GPADC data valid flag 0 : invalid 1 : valid
30	ADC_OVERFLOW	0	R	GPADC data overflow flag 0 : ADC_DATA less than register (ADC_DET_TH) 1 : ADC_DATA greater than register (ADC_DET_TH)
29	GPADC_BREAK	0	R	GPADC break flag 0 : no break 1 : break
15:7	ADC_DATA	0	R	9 bits ADC data , if ADC_OK is 1 and GPADC_BREAK is 0, the data is valid

Table 86 GPADC_DET_TH (GPADC_BASE_ADDR +0xC)

Bit	Field Name	Reset	RW	Description
8:0	GPADC_DET_TH	0	RW	ADC_DATA overflow threshold

Table 87 GPADC_CLR (GPADC_BASE_ADDR +0x10)

Bit	Field Name	Reset	RW	Description
1	CLR_GPADC	0	W	restart GPADC module 0 : invalid 1 : restart
0	CLR_OVERFLOW	0	W	clear ADC_OVERFLOW flag 0 : invalid 1 : clear

Table 88 GPADC_CTRL (GPADC_BASE_ADDR +0x14)

Bit	Field Name	Reset	RW	Description
13:6	EN_DIG_DLY	0x40	RW	GPADC stable time from GP_ADC_EN setting to 1. Unit : 1 ADC clock cycle

5	CAP_EDGE_SEL	0	RW	clock edge selection to sample GPADC data 0 : negedge 1 : posedge
4:1	DOWNSAMPLE_SEL	0x6	RW	GPADC down-sample frequency selection 1 : 2 2 : 4 3 : 8 4 : 16 5 : 32 6 : 64 7 : 128 8 : 256 9 : 512 10 : 1024 11 : 2048 12 : 4096 0,13~15 : reserved for future use
0	FILTER_EN	0x1	RW	GPADC down-sample filter enable 0 : disable 1 : enable

Table 89 MISC Register Map

Address Offset	Name	Description
0xF0	GPADC_CFG2	GPADC configure register 2

Table 90 TEMP_SENSE_EN(MISC_BASE_ADDR +0xF0)

Bit	Field Name	Reset	RW	Description
7	TEMP_SENSE_EN	0	RW	TEMP_SENSE measurement selection 0 : TEMP_SENSE not selected 1 : TEMP_SENSE selected

5.11 UART

The UART is compliant to the industry-standard 16550 and is used for serial communication with a peripheral data set. Data is written from a master (CPU/DMA) over the APB bus to the UART and it is converted to serial format and transmitted to the destination device. Serial data is also received by the UART and stored in internal FIFO, and the master (CPU/DMA) can read them back. Both UARTs support hardware flow control signals (RTS, CTS).

5.11.1 Feature

1. 8 bytes transmit and receive FIFOs
2. Hardware flow control support (CTS/RTS)
3. Functionality based on the 16550 industry standard
4. Programmable serial data baud rate
5. Programmable character properties, such as number of data bits per character (5-8)
6. Programmable parity bit (with odd/even/stick/no parity)
7. Programmable stop bits (1, 1.5 or 2)
8. Line break generation and detection
9. RX timeout interrupt support

5.11.2 Function Description

An overview of the UART module is shown in figure below.

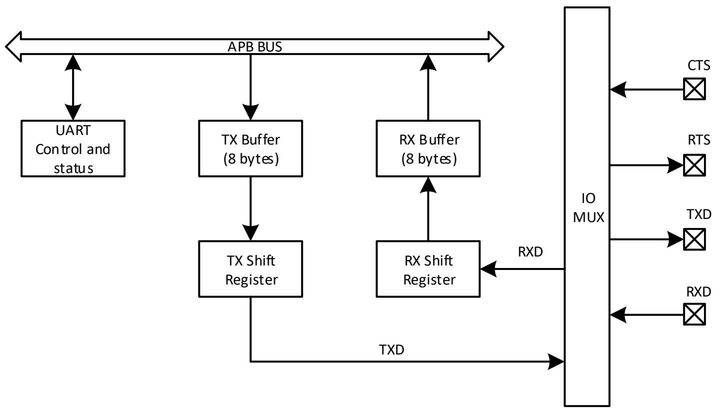


Figure 13 UART overview

5.11.2.1 Frame Format

The frame format consists of start bit, data bits, optional parity bit and stop bit(s), as shown in the figure below.

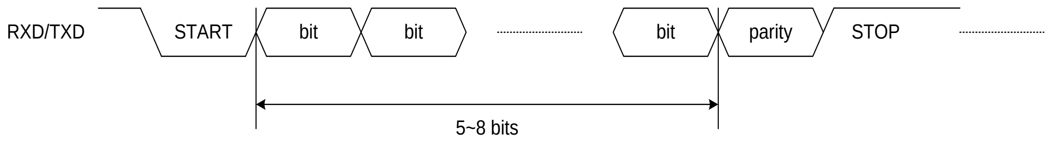


Figure 14 UART Frame Format

A frame starts with one start bit, where the line is driven low for one bit-period. The start bit indicates the start of a frame, and it is used for synchronization.

A parity bit may be added to the serial character. This bit appears after the last data bit and before the stop bit(s), and it provides the UART with the ability to perform simple error checking on the received data.

The number of data bits in a frame is set by UARTx_BYTE_SIZE in UARTx_CONFIG register, see the table below.

Table 91 UART Data Bits

BYTE_SIZE	Number of Date Bits
0	5
1	6
2	7
3	8 (default)

The number of stop-bits is set by UARTx_STOP_BIT in UARTx_CONFIG register, see the table below.

Table 92 UART Stop Bits

STOP_BIT	Number of Stop Bits
0	1 (default)
1	1.5
2/3	2

The order in which the data bits are transmitted and received is defined by UARTx_MSB in UARTx_CONFIG register. When UARTx_MSB is low, data in a frame is sent and received with the least significant bit first. When it is high, the most significant bit comes first.

5.11.2.2 Transmission

The first step of a transmission is storing bytes in the transmit buffer. When the transmission shift register is empty and ready for new data, a frame from the transmit buffer is loaded into the shift register, and transmission begins. When the frame has been transmitted, a new frame is loaded into the shift register if available, and transmission continues. If the transmit buffer is empty, the transmitter goes to an idle state, waiting for a new frame available.

If flow control is enabled through the UARTx_FLOW_CTRL_EN field in the UARTx_CONFIG register, a transmission will be automatically suspended when CTS is deactivated and resumed when CTS is activated again.

A frame can be loaded into the buffer by writing to UARTx_TFIFO_WR_DATA. When writing more frames to the transmit buffer than the free space there is, the TX FIFO overflow interrupt flag in UARTx_INT_STATUS will be set, indicating the overflow. The data already in the transmit buffer is preserved in this case, and no new data is written in.

The transmit buffer and the transmit shift register can be cleared by setting UARTx_FIFO_CLR. This will prevent the UART from transmitting the old data in the buffer and shift register, and the space is ready for new data after clearing.

5.11.2.3 Reception

Data reception is enabled when UARTx clock is enabled. When the receiver is enabled, it actively samples the input looking for a transition from high to low indicating the start baud of a new frame. When a start baud is found, reception of the new frame begins if the receive shift register is empty and ready for new data. When the frame has been received, it is pushed into the receive buffer, making the shift register ready for new frame. If the receive buffer is full, the received frame remains in the shift register until more space in the receive buffer is available. If an incoming frame is detected while both the receive buffer and the receive shift register are full, the data in the shift register is overwritten.

If flow control is enabled through the UARTx_FLOW_CTRL_EN field in the UARTx_CONFIG register, the RTS signal will be activated when the UART receives eight bytes in its internal RX FIFO.

Data can be read from the receive buffer through UARTx_RFIFO_RD_DATA register. When the data is available in the receive buffer, the RX FIFO not empty flag in UARTx_INT_STATUS is set. When the buffer becomes full, RX FIFO overflow interrupt flag in UARTx_INT_STATUS is set. To know how many bytes have been received into the RX buffer, the CPU can read the UARTx_RFIFO_CNT field in UARTx_FIFO_CNT register.

The receive buffer and the receive shift register can be cleared by setting UARTx_FIFO_CLR. Any frame currently being received will be discarded.

5.11.2.4 Parity Error and Framing Error

When a parity error is detected in an incoming frame, the interrupt flag UARTx_RXD_PARITY_ERROR is set. Frames with parity errors are not loaded into the receive buffer.

When a framing error (stop bit error) is detected in an incoming frame, the interrupt flag UARTx_RXD_STOP_ERROR is set. Frames with framing errors are not loaded into the receive buffer.

5.11.3 Software Procedure

The transmit buffer and receive buffer can be accessed by CPU and DMA.

CPU transmit flow:

1. CPU receives an under threshold interrupt when TX FIFO data counter under threshold.
2. CPU sends data to TX FIFO until TX FIFO data counter is full.
3. CPU clears interrupt.
4. Repeat 1~3 until transfer is done.

CPU receive flow:

1. CPU receives an over threshold interrupt when RX FIFO data counter over threshold.
2. CPU read RX FIFO data until RX FIFO data counter is empty.
3. CPU clears interrupt.
4. Repeat 1~3 until transfer is done.

DMA transmit flow:

1. CPU configures DMA registers for selecting UART interface and TX FIFO address.
2. UART send TX request to DMA when TX FIFO data counter under threshold.
3. DMA writes data to TX FIFO.
4. Repeat 2~3 until reach DMA channel counter.

DMA receive flow:

1. CPU configures DMA registers for selecting UART interface and RX FIFO address.
2. DMA receives RX request when RX FIFO data counter over threshold.
3. DMA read RX FIFO data until RX FIFO data counter is under threshold.
4. Repeat 2~3 until reach DMA channel counter.

5.11.4 UART Interrupt

UART interrupt table as shown in below.

Table 93 UART Interrupt

Interrupt Number	Interrupt name	Description
0	STOP_ERROR	UART receives an error stop bit.
1	PARITY_ERROR	UART receives an error parity bit.
2	LINE_BREAK	UART detects line break event.
3	TIMEOUT	RX FIFO is not empty, and no further data is received.
4	TX_FIFO_UNDER_THLD	TX FIFO data number is under threshold.
5	TX_FIFO_EMPTY	TX FIFO is empty.
6	RX_FIFO_OVER_THLD	RX FIFO data number is over threshold.
7	RX_FIFO_OVERFLOW	RX FIFO data number is overflow.
8	RX_FIFO_NOT_EMPTY	TX FIFO is not empty.
9	RX_FIFO_UNDERFLOW	RX FIFO data number is under threshold.
10	TX_FIFO_OVERFLOW	TX FIFO data number is overflow.

5.11.4.1 Error Interrupt

The error interrupt is asserted when an error occurs in the reception of data by the UART. The interrupt can be caused by the following error conditions:

- Framing (STOP_ERROR)

- Parity (PARITY_ERROR)
- Break (LINE_BREAK)

The cause of the interrupt is available by reading the UARTx_UART_INT_STATUS registers. The interrupt can be cleared by writing to the relevant bits of the UARTx_UART_INT_CLEAR register.

5.11.4.2 Timeout Interrupt

The receive timeout interrupt is asserted when the RX FIFO is not empty, and no further data is received (or no correct start bit of a frame is detected in the RXD line) over a programmable timeout period (TO_THLD in UARTx_CONFIG register). This mechanism ensures that the user is aware that data is still present in the RX FIFO and requires servicing. The receive timeout interrupt is cleared when a '1' is written to the corresponding bit of the UARTx_UART_INT_CLEAR register.

5.11.4.3 Transmit Interrupt

The transmit interrupt is asserted HIGH when one of the following conditions occurs:

- TX_FIFO_EMPTY
  The interrupt is set when no data in TX FIFO, or all written data popped out from the TX FIFO. TX FIFO EMPTY doesn't mean the shift register is empty too. Software should wait more time for the shift register bits shifting out.
- TX_FIFO_UNDER_THLD
  If the number of data in the TX FIFO is less than the register (TFIFO_THLD), this bit is asserted.
- TX_FIFO_OVERFLOW
  If the written data in TX FIFO is full (the number is eight), a new write will trigger the bit to high state.

5.11.4.4 Receive Interrupt

The receive interrupt is asserted HIGH when one of the following conditions occurs:

- RX_FIFO_NOT_EMPTY
  The interrupt is set when RX FIFO is not empty.
- RX_FIFO_UNDERFLOW
  If RX FIFO is empty, a new read command from CPU or DMA will trigger this bit to high state.
- RX_FIFO_OVERFLOW
  If RX FIFO is full (the data number is eight), a new write will trigger this bit to high state.
- RX_FIFO_OVERF_THLD
  If the data number in the RX FIFO is larger than the register (RFIFO_THLD), this bit is asserted.

5.11.5 UART Registers

Table 94 Instance

Base Address	Peripheral	Instance	Description
0x40011800	UART	UART0	UART0 Registers
0x40011C00	UART	UART1	UART1 Registers

Table 95 Register Map

Address Offset	Name	Description
0x00	UARTx_TFIFO_WR_DATA	UARTx TX FIFO Write Register
0x04	UARTx_RFIFO_RD_DATA	UARTx RX FIFO Read Register
0x08	UARTx_INT_STATUS	UARTx Interrupt Status
0x0C	UARTx_INT_EN	UARTx Interrupt Enable
0x10	UARTx_INT_CLEAR	UARTx Interrupt Flag Clear Register
0x14	UARTx_CONFIG	UARTx Configuration Register
0x18	UARTx_DIVISOR	UARTx Data Divisor
0x1C	UARTx_FIFO_CNT	UARTx FIFO Counter
0x20	UARTx_FIFO_CLR	UARTx FIFO and Data Clear

Table 96 UARTx_TFIFO_WR_DATA (UARTx_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
7:0	UARTx _TFIFO_WR_DATA	0x0	W	DMA/CPU write UARTx transmit fifo data via this register

Table 97 UARTx_RFIFO_RD_DATA (UARTx_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
7:0	UARTx_RFIFO_RD_DATA	0x0	R	DMA/CPU read UARTx receive fifo data via this register

Table 98 UARTx_INT_STATUS (UARTx_BASE_ADDR+0x08)

Bit	Field Name	Reset	RW	Description
10	UARTx_TX_FIFO_OVERFLOW	0	R	TX FIFO overflow 0: invalid 1: valid
9	UARTx_RX_FIFO_UNDERFLOW	0	R	RX FIFO underflow 0: invalid 1: valid
8	UARTx_RX_FIFO_NOT_EMPTY	0	R	RX FIFO not empty 0: invalid 1: valid
7	UARTx_RX_FIFO_OVERFLOW	0	R	RX FIFO overflow

				0: invalid 1: valid
6	UARTx_RX_FIFO_OVER_THLD	0	R	RX FIFO over threshold 0: invalid 1: valid
5	UARTx_TX_FIFO_EMPTY	0	R	TX FIFO empty 0: invalid 1: valid
4	UARTx_TX_FIFO_UNDER_THLD	0	R	TX FIFO under threshold 0: invalid 1: valid
3	UARTx_TIMEOUT	0	R	RX timeout error. The time between two RX FIFO read/write exceeds timeout threshold (UARTx_TO_THLD) when RX FIFO is not empty. 0: invalid 1: valid
2	UARTx_RXD_LINE_BREAK	0	R	RXD line break detected 0: invalid 1: valid
1	UARTx_RXD_PARITY_ERROR	0	R	RXD parity error 0: invalid 1: valid
0	UARTx_RXD_STOP_ERROR	0	R	RXD stop bit error 0: invalid 1: valid

Table 99 UARTx_INT_MASK (UARTx_BASE_ADDR+0x0C)

Bit	Field Name	Reset	RW	Description
10	UARTx_TX_FIFO_OVERFLOW_MASK	0	RW	TX FIFO overflow interrupt enable 0: disable 1: enable
9	UARTx_RX_FIFO_UNDERFLOW_MASK	0	RW	RX FIFO underflow interrupt enable 0: disable 1: enable
8	UARTx_RX_FIFO_NOT_EMPTY_MASK	0	RW	RX FIFO not empty interrupt enable 0: disable 1: enable
7	UARTx_RX_FIFO_OVERFLOW_MASK	0	RW	RX FIFO overflow interrupt enable 0: disable 1: enable
6	UARTx_RX_FIFO_OVER_THLD_MASK	0	RW	RWX FIFO over threshold interrupt

				enable 0: disable 1: enable
5	UARTx_TX_FIFO_EMPTY_MASK	0	RW	TX FIFO empty interrupt enable 0: disable 1: enable
4	UARTx_TX_FIFO_UNDER_THLD_MASK	0	RW	TX FIFO under threshold interrupt enable 0: disable 1: enable
3	UARTx_TIMEOUT_MASK	0	RW	RX timeout error interrupt enable 0: disable 1: enable
2	UARTx_RXD_LINE_BREAK_MASK	0	RW	RXD line break detected interrupt enable 0: disable 1: enable
1	UARTx_RXD_PARITY_ERROR_MASK	0	RW	RXD parity error interrupt enable 0: disable 1: enable
0	UARTx_RXD_STOP_ERROR_MASK	0	RW	RXD stop bit error interrupt enable 0: disable 1: enable

Table 100 UARTx_INT_CLEAR (UARTx_BASE_ADDR+0x10)

Bit	Field Name	Reset	RW	Description
10	UARTx_TX_FIFO_OVERFLOW_CLEAR	0	W	TX FIFO overflow interrupt clear 0: invalid 1: valid
9	UARTx_RX_FIFO_UNDERFLOW_CLEAR	0	W	RX FIFO underflow interrupt clear 0: invalid 1: valid
8	UARTx_RX_FIFO_NOT_EMPTY_CLEAR	0	W	RX FIFO not empty interrupt clear 0: invalid 1: valid
7	UARTx_RX_FIFO_OVERFLOW_CLEAR	0	W	RX FIFO overflow interrupt clear 0: invalid 1: valid
6	UARTx_RX_FIFO_OVER_THLD_CLEAR	0	W	RX FIFO over threshold interrupt clear 0: invalid 1: valid
5	UARTx_TX_FIFO_EMPTY_CLEAR	0	W	TX FIFO empty interrupt clear

				0: invalid 1: valid
4	UARTx_TX_FIFO_UNDER_THLD_CLEAR	0	W	TX FIFO under threshold interrupt clear 0: invalid 1: valid
3	UARTx_TIMEOUT_CLEAR	0	W	RX timeout error interrupt clear. 0: invalid 1: valid
2	UARTx_RXD_LINE_BREAK_CLEAR	0	W	RXD line break detected interrupt clear 0: invalid 1: valid
1	UARTx_RXD_PARITY_ERROR_CLEAR	0	W	RXD parity error interrupt clear 0: invalid 1: valid
0	UARTx_RXD_STOP_ERROR_CLEAR	0	W	RXD stop bit error interrupt clear 0: invalid 1: valid

Table 101 UARTx_CONFIG (UARTx_BASE_ADDR+0x14)

Bit	Field Name	Reset	RW	Description
30	UARTx_ERR_DISCARD	0	RW	discard event enable when error parity bit or error stop bit data received 0: disable 1: enable
29:26	UARTx_RTS_THLD	0x0	RW	When the number of data in rx fifo is larger than RTS threshold then suspend
25	UARTx_FLOW_CTRL_EN	0	RW	hardware flow control 0: disable 1: enable
24	UARTx_LB_SND	0	RW	line break enable 0: disable 1: enable
23	UARTx_MSB	0	RW	big-endian mode or little-endian mode selection 0: little-endian 1: big-endian
22:20	UARTx_PARITY_MODE	0x0	RW	parity mode 0: parity odd 1: parity even 2: parity stick at 1 3: parity stick at 0 4~7: reserved

19:18	UARTx_BYTE_SIZE	0x3	RW	transmitted data size 0: 5 bit 1: 6 bit 2: 7 bit 3: 8 bit
17:16	UARTx_STOP_BIT	0x0	RW	stop bit length 0: 1 bit 1: 1.5 bit 2~3: 2 bit
15:8	UARTx_TO_THLD	0xFF	RW	RX time out threshold unit is 1 symbol (byte)
7:4	UARTx_RFIFO_THLD	0x4	RW	RX FIFO threshold When the data number in rx fifo arrived at RX threshold, interrupt is generated.
3:0	UARTx_TFIFO_THLD	0x4	RW	TX FIFO threshold When the data number in tx fifo arrived at TX threshold, interrupt is generated.

Table 102 UARTx_DIVISOR (UARTx_BASE_ADDR+0x18)

Bit	Field Name	Reset	RW	Description
25:16	UARTx_TRIG_INTV	0x13	RW	input RXD detected interval should be set (cap_intv+1)/7
15:0	UARTx_CAP_INTV	0x89	RW	Baud rate setting Baud rate = 16MHz/(cap_intv+1)

Table 103 UARTx_FIFO_CNT (UARTx_BASE_ADDR+0x1C)

Bit	Field Name	Reset	RW	Description
7:4	UARTx_RFIFO_CNT	0x0	R	rx fifo counter
3:0	UARTx_TFIFO_CNT	0x0	R	tx fifo counter

Table 104 UARTx_FIFO_CLR (UARTx_BASE_ADDR+0x20)

Bit	Field Name	Reset	RW	Description
0	UARTx_FIFO_CLR	0	W	counter (rx and tx) and data (rx and tx) both cleared 0: invalid 1: clear

5.12 SPI

5.12.1 Feature

SPI is a serial parallel interface compatible with Motorola standard. The main features are:

1. Master and slave mode
2. Programmable TX only, RX only and TRX mode
3. Support CPU and DMA access
4. Programmable output frequency in master mode. Up to 8MHz clock for both master and slave mode
5. Programmable clock phase and polarity
6. Programmable data frame size from 8-bits, 16-bits, 32-bits and 64-bits.
7. 8 bytes transmit FIFOs, 8 bytes receive FIFOs

5.12.2 Function Description

The SPI performs serial-to-parallel conversion on data received from a peripheral device on the MISO pin in master mode and on the MOSI pin in slave mode. The SPI performs parallel-to-serial conversion on data written by the CPU/DMA for transmission on the MOSI pin in master mode and on the MISO pin in slave mode.

The SPI has a programmable bit-rate clock divider to generate the serial output clock signal on the SCLK pin. The transmission and reception paths have individual 8 bytes FIFO memories. FIFOs may be burst-loaded or emptied by the CPU/DMA.

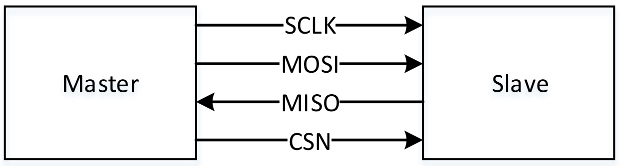


Figure 15 SPI Interface Signals

5.12.2.1 Clock Mode (Phase and Polarity)

The CPOL bit determines the polarity of the clock.

- CPOL=0 means idle state is ‘0’, and each cycle consists of a pulse of 1. That is, the leading edge is a rising edge, and the trailing edge is a falling edge.
- CPOL=1 means idle state is ‘1’, and each cycle consists of a pulse of 0. That is, the leading edge is a falling edge, and the trailing edge is a rising edge.

The CPHA bit determines the timing of the data bit relative to the clock pulse.

- For CPHA=0, the ‘out’ side changes the data on the trailing edge of the preceding clock cycle, while the ‘in’ side captures the data on (or shortly after) the leading edge of the clock cycle. The outside holds the data valid until the trailing edge of the current clock cycle. For the first cycle, the first bit must be on the MOSI line before the leading clock edge.

- For CPHA=1, the 'out' side changes the data on the leading edge of the current clock cycle, while the 'in' side captures the data on (or shortly after) the trailing edge of the clock cycle. The outside holds the data valid until the leading edge of the following clock cycle. For the last cycle, the slave holds the MISO line valid until slave select is deserted.

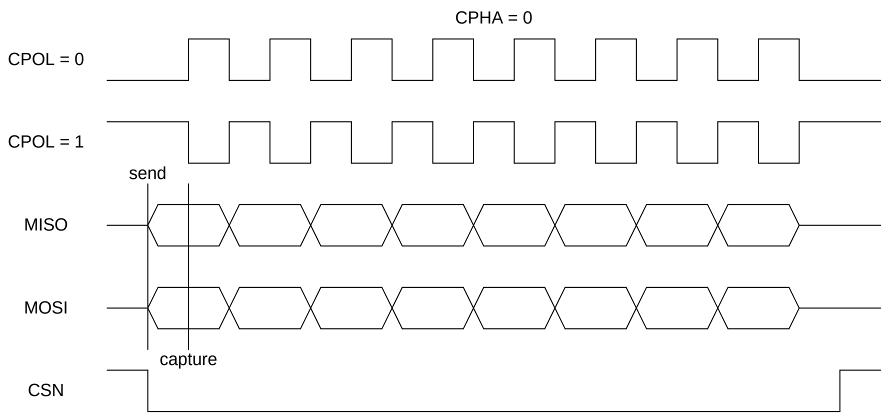


Figure 16 SPI Clock Mode Diagram (CPHA=0)

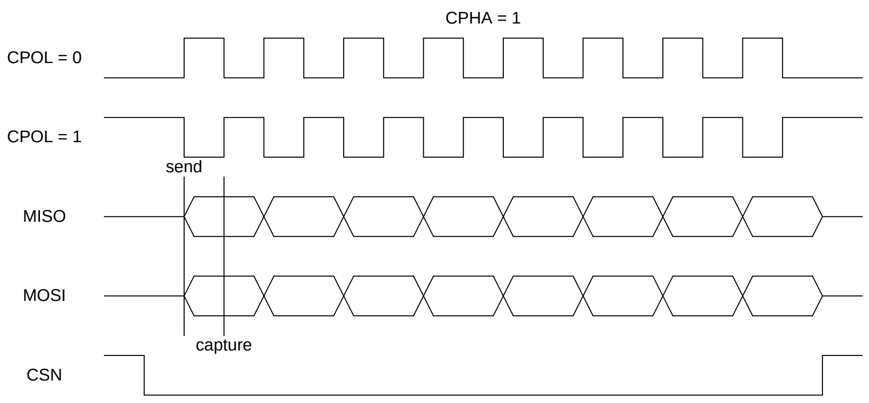


Figure 17 SPI Clock Mode Diagram (CPHA=1)

5.12.2.2 Clock Generation

In master mode, the SCLK frequency is configurable.

- SCLK = 8MHz / (1 + register: SPI_CLK_DIVISOR)

In slave mode, the SCLK is from the external master, and the supported highest clock is 8MHz.

5.12.2.3 TX Idle Data

In master mode, when TX FIFO is empty and RX only mode, SPI transmits register value (TX_IDLE_DATA) to external slave.
In slave mode, when TX FIFO is empty, SPI transmits register value (TX_IDLE_DATA) to external master.

5.12.3 Software Procedure

5.12.3.1 Receive Procedure

1. Empty the receive FIFO by set SPI_FIFO_CLR register.
2. Program GPIO to route SPI port signals on those GPIO (refer to GPIO registers).
3. Program the SPI_CONFIG register to RX/TRX mode.
4. Receive data
    a) If in the master and RX only mode, set SPI enable (RX_EN) and RX length (MRXO_LEN).
    b) If in the master and TRX mode, write the register (SPI_TX_FIFO).
    c) If in the slave mode, wait external master transmit data.

5.12.3.2 Transmit Procedure

1. Empty the transmit FIFO by set SPI_FIFO_CLR register.
2. Program GPIO to rout SPI port signals on those GPIO (refer to GPIO registers).
3. Program the SPI_CONFIG register to TX/TRX mode.
4. Transmit data
    a) If in the master mode, write the register (SPI_TX_FIFO).
    b) If in the slave mode, write the register (SPI_TX_FIFO) and wait for an external master's SCLK.

5.12.4 Interrupt

SPI interrupt table is shown below.

Table 105 SPI Interrupt

Interrupt Number	Interrupt name	Description
10	TX_FIFO_FULL	Set during transmit if the register (TFIFO_CNT) is filled to 8
9	RX_FIFO_FULL	Set during receive if the register (RFIFO_CNT) is filled to 8
8	TX_FIFO_OVERFLOW	Set during transmit if the register (TFIFO_CNT) is filled to 8 and the DMA/CPU attempts to issue another data by writing to the register (TX_FIFO)

7	TX_FIFO_UNDERFLOW (only SPI slave mode)	Set during transmit if the register (TFIFO_CNT) is equal to 0 and the external SPI master send next frame completed
6	RX_FIFO_UNDERFLOW	Set if the DMA/CPU attempts to read the receive buffer when it is empty by reading from the register (RX_FIFO)
5	RX_FIFO_NOT_EMPTY	Set during receive if the register (RFIFO_CNT) isn't equal to 0
4	RX_FIFO_OVERFLOW (only SPI slave mode)	Set during receive if the register (RFIFO_CNT) is filled to 8 and the external SPI master send next frame completed Indicate when a DMA/CPU read register (RX_FIFO) operation is not completed in time
3	RX_FIFO_OVERF_THLD	Set when the receive buffer reaches or goes above the register (RFIFO_THLD)
2	TX_FIFO_EMPTY	Set during transmit if the register (TFIFO_CNT) is equal to 0
1	TX_FIFO_UNDER_THLD	Set when the transmit buffer reaches or goes below the register (TFIFO_THLD)
0	RX_TIMEOUT	In slave mode, set when the RX FIFO is not empty, and no further data is received over a programmable timeout period (TO_THLD in SPI_CONFIG register).

5.12.5 Registers

Table 106 SPI Instance

Base Address	Peripheral	Instance	Description
0x40017000	SPI	SPI	Serial-Peripheral-Interface

Table 107 SPI Register Map

Address Offset	Name	Description
0x00	SPI_TX_FIFO	DMA/CPU write SPI transmit FIFO data via this register
0x04	SPI_RX_FIFO	DMA/CPU read SPI receive FIFO data via this register
0x08	SPI_INT_FLAG	SPI interrupt flag
0x0C	SPI_INT_EN	SPI interrupt enable
0x10	SPI_INT_CLR	SPI interrupt clear
0x14	SPI_CONFIG	SPI configuration
0x18	SPI_DIVISOR	SPI master mode clock divisor
0x1C	SPI_FIFO_CNT	SPO TX and RX FIFO counter
0x20	SPI_TX_IDLE	SPI transmit idle data
0x24	SPI_EXTM_READ_LEN	external master read FIFO counter
0x28	SPI_FIFO_CLR	SPI 8x8 FIFO clear

Table 108 SPI_TX_FIFO (SPI_BASE_ADDR +0x0)

Bit	Field Name	Reset	RW	Description
7:0	SPI_TX_FIFO	0x0	W	DMA/CPU write SPI transmit FIFO data via this register

Table 109 SPI_RX_FIFO (SPI_BASE_ADDR +0x4)

Bit	Field Name	Reset	RW	Description
7:0	SPI_RX_FIFO	0x0	R	DMA/CPU read SPI receive FIFO data via this register

Table 110 SPI_INT_FLAG (SPI_BASE_ADDR +0x8)

Bit	Field Name	Reset	RW	Description
10	TX_FIFO_FULL	0	R	See Interrupt
9	RX_FIFO_FULL	0	R	See Interrupt
8	TX_FIFO_OVERFLOW	0	R	See Interrupt
7	TX_FIFO_UNDERFLOW	0	R	See Interrupt
6	RX_FIFO_UNDERFLOW	0	R	See Interrupt
5	RX_FIFO_NOT_EMPTY	0	R	See Interrupt
4	RX_FIFO_OVERFLOW	0	R	See Interrupt
3	RX_FIFO_OVERF_THLD	0	R	See Interrupt
2	TX_FIFO_EMPTY	0	R	See Interrupt
1	TX_FIFO_UNDER_THLD	0	R	See Interrupt
0	RX_TIMEOUT	0	R	See Interrupt

Table 111 SPI_INT_EN (SPI_BASE_ADDR +0xC)

Bit	Field Name	Reset	RW	Description
10	TX_FIFO_FULL_EN	0	RW	Interrupt enable register 0 : disable 1 : enable
9	RX_FIFO_FULL_EN	0	RW	Interrupt enable register 0 : disable 1 : enable
8	TX_FIFO_OVERFLOW_EN	0	RW	Interrupt enable register 0 : disable 1 : enable
7	TX_FIFO_UNDERFLOW_EN	0	RW	Interrupt enable register 0 : disable 1 : enable
6	RX_FIFO_UNDERFLOW_EN	0	RW	Interrupt enable register 0 : disable 1 : enable
5	RX_FIFO_NOT_EMPTY_EN	0	RW	Interrupt enable register 0 : disable 1 : enable
4	RX_FIFO_OVERFLOW_EN	0	RW	Interrupt enable register 0 : disable

				1 : enable
3	RX_FIFO_OVERF_THLD_EN	0	RW	Interrupt enable register 0 : disable 1 : enable
2	TX_FIFO_EMPTY_EN	0	RW	Interrupt enable register 0 : disable 1 : enable
1	TX_FIFO_UNDER_THLD_EN	0	RW	Interrupt enable register 0 : disable 1 : enable
0	RX_TIMEOUT_EN	0	RW	Interrupt enable register 0 : disable 1 : enable

Table 112 SPI_INT_CLR (SPI_BASE_ADDR +0x10)

Bit	Field Name	Reset	RW	Description
10	TX_FIFO_FULL_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear
9	RX_FIFO_FULL_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear
8	TX_FIFO_OVERFLOW_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear
7	TX_FIFO_UNDERFLOW_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear
6	RX_FIFO_UNDERFLOW_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear
5	RX_FIFO_NOT_EMPTY_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear
4	RX_FIFO_OVERFLOW_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear
3	RX_FIFO_OVERF_THLD_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear
2	TX_FIFO_EMPTY_CLR	0	W	Interrupt clear register 0 : invalid

				1 : clear
1	TX_FIFO_UNDER_THLD_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear
0	RX_TIMEOUT_CLR	0	W	Interrupt clear register 0 : invalid 1 : clear

Table 113 SPI_CONFIG (SPI_BASE_ADDR +0x14)

Bit	Field Name	Reset	RW	Description
31	RX_EN	0	RW	SPI RX enable 0 : disable 1 : enable
30	MASTE_SLAVE	0	RW	Master or slave mode selection 0 : slave mode 1 : master mode
29	CPOL	0	RW	See Clock Mode
28	CPHA	0	RW	See Clock Mode
27:20	MRXO_LEN	0	RW	SPI receive data length when master mode (when master is set to '1') and RX only mode (when TRX_MODE[1:0] is set to '2') Unit : 1 CSN frame length
19:18	SIZE	0	RW	SPI master mode CSN frame size 0 : 8bit 1 : 16bit 2 : 32bit 3 : 64bit
17:16	TRX_MODE	0x2	RW	SPI TRX mode: 0 : transmit enable, receive enable (full duplex mode) 1 : transmit enable, receive disable (TX only mode) 2 : transmit disable, receive enable (TX only mode) 3 : transmit & receive disable
15:8	TO_THLD	0xFF	RW	SPI RX time out threshold Unit : 1 byte
7:4	RFIFO_THLD	0x4	RW	receive FIFO threshold
3:0	TFIFO_THLD	0x4	RW	transmit FIFO threshold

Table 114 SPI_CONFIG (SPI_BASE_ADDR +0x18)

Bit	Field Name	Reset	RW	Description
7:0	SPI_CLK_DIVISOR	0	RW	See Clock Generation

Table 115 SPI_FIFO_CNT (SPI_BASE_ADDR +0x1c)

Bit	Field Name	Reset	RW	Description

7:4	RFIFO_CNT	0	R	receive FIFO counter
3:0	TFIFO_CNT	0	R	transmit FIFO counter

Table 116 SPI_TX_IDLE (SPI_BASE_ADDR +0x20)

Bit	Field Name	Reset	RW	Description
7:0	TX_IDLE_DATA	0	RW	See TX Idle Data

Table 117 SPI_EXTM_READ_LEN (SPI_BASE_ADDR +0x24)

Bit	Field Name	Reset	RW	Description
8	SPI_EXTM_READ_LEN_EN	0	RW	SPI in slave mode. if received byte (in the current frame) matches SPI_EXTM_READ_LEN_CMD, SPI transmits one byte data (4bits TFIFO_CNT, 4bits RFIFO_CNT) for next frame to external master. 0 : disable 1 : enable
7:0	SPI_EXTM_READ_LEN_CMD	0	RW	See SPI_EXTM_READ_LEN_EN

Table 118 SPI_FIFO_CLR (SPI_BASE_ADDR +0x28)

Bit	Field Name	Reset	RW	Description
0	SPI_FIFO_CLR	0	W	SPI 8x8 TRX FIFO clear 0 : invalid 1 : clear

5.13 I2C

The I2C is a programmable control bus that provides support for the communications link between Integrated Circuits in a system.

5.13.1 Feature

1. Two-wire I2C serial interface consists of a serial data line (SDA) and a serial clock (SCL)
2. Support both master and slave modes
3. Standard mode (0 to 100 KHz) and Fast mode (<= 400 KHz)
4. 8 bytes transmit FIFOs, 8 bytes receive FIFOs
5. Support 7-bit and 10-bit addressing
6. Configurable slave address
7. Support restart command
8. Handle Bit and Byte waiting
9. Support CPU and DMA masters

5.13.2 I2C Protocol

The I2C controller has the following protocols.

- START and STOP Conditions
- Addressing Slave Protocol
- Transmitting and Receiving Protocol

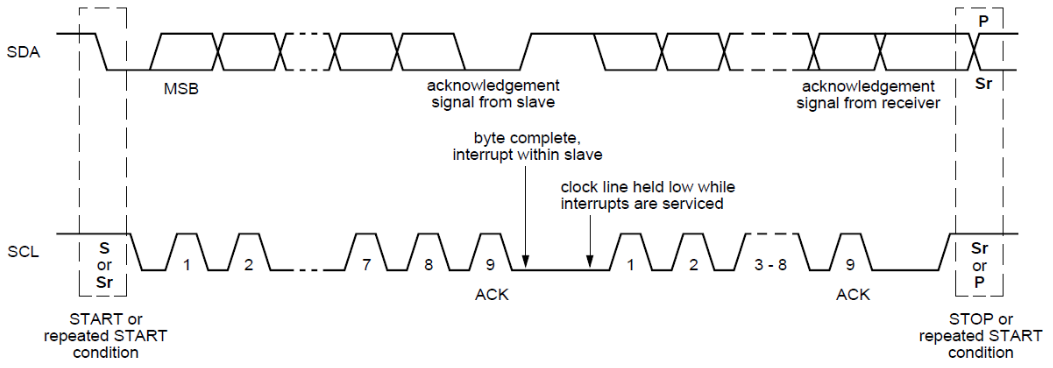

SDA
SCL
MSB
acknowledgement signal from slave
acknowledgement signal from receiver
byte complete, interrupt within slave
clock line held low while interrupts are serviced
S or Sr
1
2
7
8
9
1
2
3 - 8
9
ACK
ACK
P
Sr
Sr or P
START or repeated START condition
STOP or repeated START condition

Figure 18 Data Transfer on the I2C Bus

The I2C is a synchronous serial interface. The SDA line is a bidirectional signal and changes only while the SCL line is low, except for STOP, START and RESTART conditions. The output drivers are open-drain to perform wire-AND functions on the bus.

5.13.2.1 Addressing Slave Protocol

There are two address formats, the 7-bit address format and the 10-bit address format.

7-bit Address Format

In the 7-bit address format, the first seven bits (bits 7:1) of the first byte set by the slave address and the LSB bit (bit 0) is the R/W bit. When bit 0 (R/W) is set to ‘0’, the master writes to the slave. When bit 0 (R/W) is set to ‘1’, the master reads from the slave.

10-bit Address Format

During 10-bit addressing, two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first five bits (bits 7:3) notify the slaves that this is a 10-bit transfer followed by the next two bits (bits 2:1), which set the slaves address bits 9:8, and the LSB bit (bit 0) is the R/W bit. The second byte transferred sets bits 7:0 of the slave address.

5.13.2.2 Transmitting and Receiving Protocol

The master can initiate data transmission and reception to/from the bus, acting as either a master-transmitter or master-receiver. A slave responds to requests from the master to either transmit data or receive data to/from the bus, acting as either a slave-transmitter or slave-receiver, respectively.

Master-Transmitter and Slave-Receiver

All data is transmitted in byte format, with no limit on the number of bytes transferred per data transfer. After the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave receiver must respond with the acknowledge signal (ACK). When a slave-receiver does not respond with an ACK pulse, the master aborts the transfer by issuing a STOP condition. The slave must leave the SDA line high so that the master can abort the transfer. If the master-transmitter is transmitting data, then the slave-receiver responds to the master-transmitter with an acknowledge pulse after every data is received.

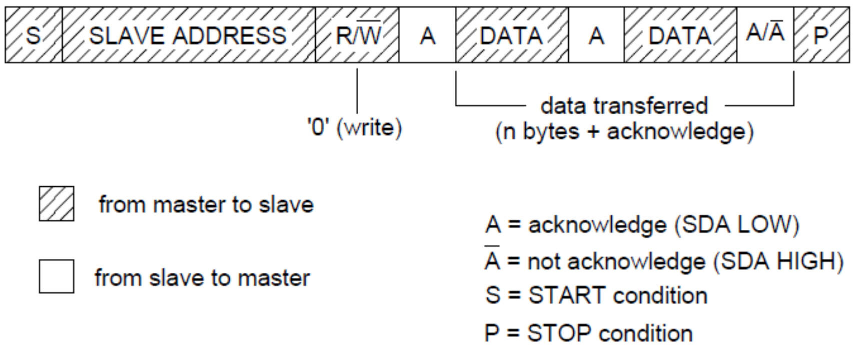


Figure 19 Master-Transmitter Protocol

Master-Receiver and Slave-Transmitter

If the master is receiving data then the master responds to the slave-transmitter with an acknowledge pulse after a byte of data has been received, except for the last byte. This is the way the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge (NACK) so that the master can issue a STOP condition. When a master does not want to relinquish the bus with a STOP condition, the master can issue a RESTART condition. This is identical to a START condition except it occurs after the ACK pulse. The master can then communicate with the same slave or a different slave.

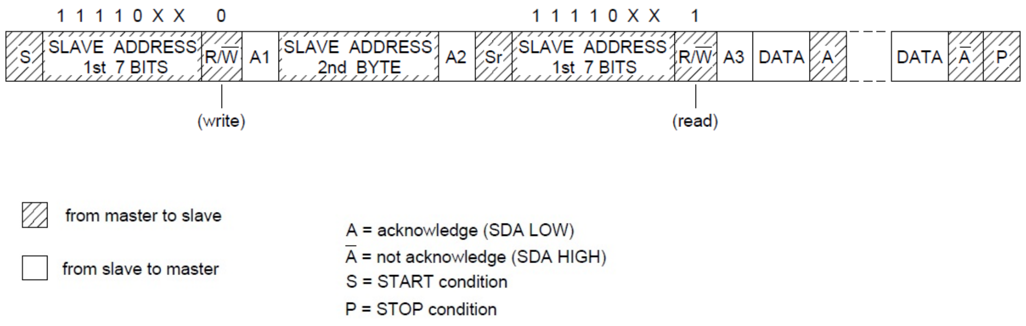


Figure 20 Master-Receiver Protocol

5.13.2.3 Multiple Master Arbitration

The I2C controller bus protocol allows multiple masters to reside on the same bus. If there are two masters on the same I2C-bus, there is an arbitration procedure if both try to take control of the bus at the same time by generating a START condition at the same time. Once a master (for example, a microcontroller) has control of the bus, no other master can take control until the first master sends a STOP condition and places the bus in an idle state. Arbitration takes place on the SDA line, while the SCL line is ‘1’. The master, which transmits a ‘1’ while the other master transmits ‘0’, loses arbitration and turns off its data output stage. The master that lost arbitration can continue to generate clocks until the end of the byte transfer. If both masters are addressing the same slave device, the arbitration could go into the data phase. Figure below illustrates the timing of when two masters are arbitrating on the bus. Because the codes are unique, only one master can win arbitration.

Master arbitration is lost when:

a) Master drives SDA high, but the I2C bus is low

b) Stop condition detected while not requested

Slave doesn’t have arbitration lost scenario.

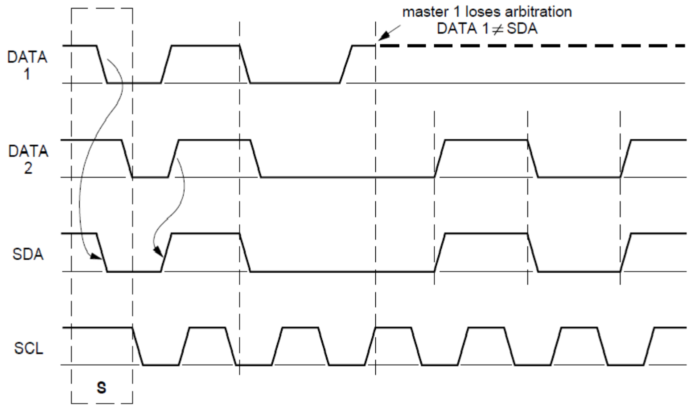


Figure 21 Arbitration Procedure of Two Masters

5.13.3 Function Description

An overview of the I2C module is shown in the figure below.

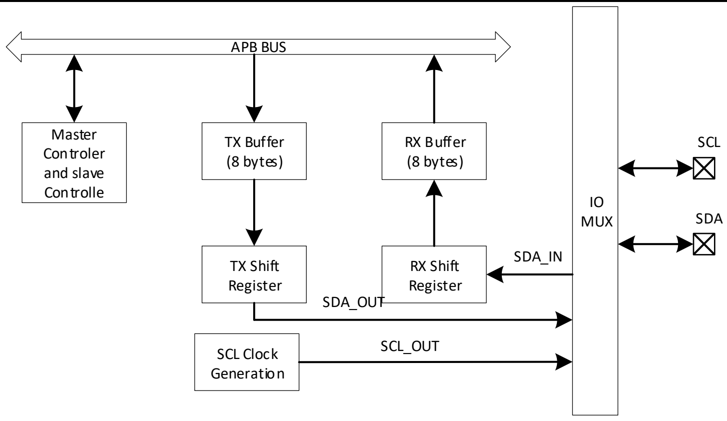


Figure 22 I2C overview

5.13.3.1 Transmission

It can be set as I2C master-transmitter or I2C slave-transmitter. Refer to the “Software Procedure” chapter.

I2C slave-transmitter
When external I2C device requests data from I2C slave, I2C slave controller will ask data from TX FIFO and send data to external I2C device. If the TX FIFO is empty, the transmitter sends NACK, and the external master will terminate the transfer.

I2C master-transmitter
When setting in I2C master-transmitter mode, I2C master controller will ask data from TX FIFO and send data to external I2C device. When the byte has been transmitted, a new byte is loaded into the shift register if available, and transmission continues until master command (I2C_I2CM_CR and I2C0_I2CM_TRANSFER_LEN) is finished. If the TX FIFO is empty, the transmitter terminates the transfer.

TX FIFO last data shift out in master mode
When TX FIFO is empty, the last data may still in the shift buffer and it needs some time to wait for the buffer shift out. Software can check I2CM_IRQ and I2CM_AL to judge whether the last byte has been shifted out (I2CM_IRQ is ‘1’, I2M_AL is ‘0’) or not.

TX FIFO overflow
A byte can be loaded into TX FIFO by writing to I2C_TFIFO_WR_DATA. When writing more bytes to the transmit buffer than the free space there is, the TX FIFO overflow interrupt flag in I2C_INT_STATUS will be set to indicate the overflow.

TX FIFO clear
The TX FIFO and the FIFO counter can be cleared by setting I2C_FIFO_CLR. This will prevent the I2C from transmitting the old data in the buffer and shift register.

5.13.3.2 Reception

It can be set as an I2C master-receiver or I2C slave-receiver. Refer to the “Software Procedure” chapter.

I2C master-receiver

When I2C master-receiver receives bytes from external I2C device, the I2C master will receive data and send data to RX FIFO. Received bytes can be read from the RX FIFO by the register (I2C_RFIFO_RD_DATA). When data becomes available in the RX FIFO, the RX FIFO not empty flag in I2C_INT_STATUS is set. When the RX FIFO becomes near full, I2C master will receive the current byte and transmit NACK to stop the reading. The CPU can read the RFIFO_CNT field in I2C_FIFO_CNT register to see how many bytes received in the RX FIFO.

I2C slave-receiver

When external I2C device sends data to I2C slave, I2C slave will receive data and send data to RX FIFO. Received data can be read from the RX FIFO by the register (I2C_RFIFO_RD_DATA). When data becomes available in the RX FIFO, the RX FIFO not empty flag in I2C_INT_STATUS is set. When the RX FIFO becomes near full, I2C slave will receive the current byte data and transmit NACK, and external I2C master will stop the writing. The CPU can read the RFIFO_CNT field in I2C_FIFO_CNT register to see how many bytes received in the RX FIFO.

The RX FIFO and FIFO counter can be cleared by setting I2C_FIFO_CLR.

5.13.3.3 RX FIFO Timeout

When RX FIFO Timeout is detected, the interrupt TIMEOUT_ERROR is set. The RX FIFO Timeout appears when the time between two RX FIFO read/write exceeds the timeout threshold (I2C_TO_THLD) when RX FIFO is not empty.

5.13.3.4 Master SCL Clock Generation

When in master mode, the SCL frequency is determined by the register I2CM_PRER. Refer to the following examples for I2CM_PRER setting.

- when SCL is 100KHz, I2C_PRER = 16MHz/(5*100KHz) - 1 = 31
- when SCL is 400KHz, I2C_PRER = 16MHz/(5*400KHz) - 1 = 7

5.13.4 Software Procedure

Master initial configuration.

1. Disable I2C master by writing ‘0’ to bit 7 of the register (I2C_I2CM_CFG).
2. Select I2C master mode by writing ‘1’ to bit 18 of the register (I2C_CFG).
3. Write the register (I2C_I2CM_CFG[23:8]) to set the frequency of SCL.
4. Write the register (I2C_I2CM_CFG[6]) to select 7-bit or 10-bit addressing.

5. Write the register (I2C_I2CMS_ADD) to set the I2C master target address.
6. Enable the I2C master by writing ‘1’ to bit 7 of the register (I2C_I2CM_CFG).
7. Write the register (I2C_I2CM_CR).

Master transmit flow by CPU.

1. CPU sends data to TX FIFO.
2. I2C PHY reads TX FIFO data and sends the data out by I2C interface.
3. I2C PHY sends interrupt to CPU when TX FIFO data counter reaches threshold.
4. CPU sends new data and clears interrupt.
5. CPU checks all data are transmitted out (I2CM_IRQ is ‘1’, I2M_AL is ‘0’).

Master receive flow by CPU.

1. I2C PHY receives data from I2C interface and writes data to RX FIFO.
2. I2C PHY sends interrupt to CPU when RX FIFO data count reaches threshold.
3. CPU reads RX FIFO data.
4. CPU clears interrupt.

Slave initial configuration.

1. Select I2C slave mode by writing ‘0’ to bit 18 of the register (I2C_CFG).
2. Write the register (I2C_I2CMS_ADD) to set the slave address.
3. Write the register (I2C_CFG) to specify which type of addressing is supported (7-bit or 10-bit).

Slave receive flow by CPU.

1. I2C PHY receives data from I2C interface and sends data to RX FIFO.
2. I2C PHY sends interrupt to CPU when RX FIFO data count reaches threshold.
3. CPU reads RX FIFO data.
4. CPU clears interrupt.

Slave transmit flow by CPU.

1. CPU sends data to TX FIFO.
2. I2C PHY sends TX FIFO data by I2C interface.
3. I2C PHY sends interrupt to CPU when TX FIFO data count reach threshold.
4. CPU sends new data to TX FIFO.
5. CPU clears interrupt.

DMA transmit flow.

1. DMA configures I2C interface and TX FIFO address.
2. I2C PHY sends TX request to DMA when TX FIFO data counter under threshold.
3. DMA writes data to TX FIFO.
4. TX FIFO loads data to TX shifter register.
5. TX shift register drives data to SDA.
6. Repeat step 2~5 until DMA finishes.

DMA receive flow.

1. DMA configures I2C interface and RX FIFO address.
2. RX shift register receives SDA data from I2C interface.
3. RX shift register loads bytes to RX FIFO.
4. DMA receives RX request when RX FIFO data counter over threshold.
5. DMA reads RX FIFO data until RX FIFO data counter is under threshold.
6. Repeat 2~5 until DMA finishes.

5.13.5 I2C interrupt

I2C interrupt table as shown in below:

Table 119 I2C interrupt

Interrupt Number	Interrupt name	Description
0	TIMEOUT_ERROR	RX timeout error.
1	TX_FIFO_UNTH	TX FIFO data number is under threshold.
2	TX_FIFO_EMPTY	TX FIFO empty
3	TX_FIFO_UDFL	TX FIFO over read when TX FIFO empty
4	TX_FIFO_OVFL	TX FIFO over write when TX FIFO full
5	RX_FIFO_OVTH	RX FIFO over threshold
6	RX_FIFO_FULL	RX FIFO full
7	RX_FIFO_EMPTY_N	RX FIFO not empty
8	RX_FIFO_UDFL	RX FIFO over read when RX FIFO empty
9	I2CM_IRQ	Master transfer completed flag or arbitration is lost
10	I2CS_IR_PHY	I2C SLAVE detect misplace START or STOP
11	I2CS_ADDR_MATCH	I2C SLAVE ADDRESS MATCH

5.13.6 I2C Registers

Table 120 I2C Instance

Base Address	Peripheral	Instance	Description
0x40011000	I2C	I2C	I2C

Table 121 Register Map

Address Offset	Name	Description
0x00	I2C_TFIFO_WR_DATA	I2C TX FIFO Write Register
0x04	I2C_RFIFO_RD_DATA	I2C RX FIFO Read Register
0x08	I2C_INT_STATUS	I2C Interrupt Status
0x0C	I2C_INT_MASK	I2C Interrupt Enable

0x10	I2C_INT_CLR	I2C Interrupt Flag Clear Register
0x14	I2C_I2CM_CR	I2C Master Command Register The command register should be configured before each transfer. For example: write (0xD8), read(0xE8), restart after write(0xDC)
0x18	I2C_CFG	I2C Configure Register
0x1C	I2C_I2CM_CFG	I2C Master Configure Register
0x20	I2C_I2CMS_ADD	I2C Master/Slave Register
0x28	I2C_BYTE_FIFO_CNT	I2C Byte rest and RX FIFO Counter, TX FIFO Counter Read Register
0x2C	I2C_I2CM_TRANSFER_LEN	I2C Master Byte Transfer Length
0x30	I2C_TFIFO_IDLE_DATA	TX Idle Data When I2C TX FIFO EMPTY
0x34	I2C_STATUS	I2C Status Read Register
0x38	I2C_FIFO_CLR	I2C TX/RX FIFO data, FIFO Counter and FIFO Read/Write address Clear

Table 122 I2C_TFIFO_WR_DATA (I2C_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
7:0	I2C _TFIFO_WR_DATA	0x0	W	DMA/CPU write I2C TX FIFO data via this register

Table 123 I2C_RFIFO_RD_DATA (I2C_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
7:0	I2C_RFIFO_RD_DATA	0x0	R	DMA/CPU read I2C RX FIFO data via this register

Table 124 I2C_INT_STATUS (I2C_BASE_ADDR+0x08)

Bit	Field Name	Reset	RW	Description
11	I2CS_ADDR_MATCH	0	R	When slave address is received and it is matched in slave mode 0: invalid 1: valid
10	I2CS_IR_PHY_ERR	0	R	Slave bus error interrupt indicates slave received a START or STOP condition which occurs on the wrong position 0: invalid 1: valid
9	I2CM_IRQ	0	R	Transfer is completed or arbitration is lost in master mode 0: invalid 1: valid
8	I2C_RFIFO_UDFL	0	R	Read RX FIFO when RX FIFO is empty

				0: invalid 1: valid
7	I2C_RFIFO_EMPTY_N	0	R	RX FIFO not empty 0: invalid 1: valid
6	I2C_RFIFO_FULL	0	R	RX FIFO full 0: invalid 1: valid
5	I2C_RFIFO_OVTH	0	R	RX FIFO over threshold 0: invalid 1: valid
4	I2C_TFIFO_OVFL	0	R	Write TX FIFO when TX FIFO is full 0: invalid 1: valid
3	I2C_TFIFO_UDFL	0	R	external I2C device read TX FIFO when TX FIFO is empty 0: invalid 1: valid
2	I2C_TFIFO_EMPTY	0	R	TX FIFO empty 0: invalid 1: valid
1	I2C_TFIFO_UNTH	0	R	TX FIFO under threshold 0: invalid 1: valid
0	I2C_RX_TO	0	R	RX timeout error. The time between two RX FIFO read/write exceeds timeout threshold(I2C_TO_THLD) when RX FIFO is not empty 0: invalid 1: valid

Table 125 I2C_INT_MSAK (I2C_BASE_ADDR+0x0C)

Bit	Field Name	Reset	RW	Description
11	I2CS_ADDR_MATCH_EN	0	RW	Slave address match interrupt enable When slave address is received and it is matched in slave mode 0: disable 1: enable
10	I2CS_IR_PHY_ERR_EN	0	RW	Slave bus error interrupt indicates slave received a START or STOP condition which occurs on the wrong position interrupt enable

				0: disable 1: enable
9	I2CM_IRQ_EN	0	RW	Transfer is completed or arbitration is lost in master mode interrupt enable 0: disable 1: enable
8	I2C_RFIFO_UDFL_EN	0	RW	Read RX FIFO when RX FIFO is empty interrupt enable 0: disable 1: enable
7	I2C_RFIFO_EMPTY_N_EN	0	RW	RX FIFO not empty interrupt enable 0: disable 1: enable
6	I2C_RFIFO_FULL_EN	0	RW	RX FIFO full interrupt enable 0: disable 1: enable
5	I2C_RFIFO_OVTH_EN	0	RW	RX FIFO over threshold interrupt enable 0: disable 1: enable
4	I2C_TFIFO_OVFL_EN	0	RW	Write TX FIFO when TX FIFO is full interrupt enable 0: disable 1: enable
3	I2C_TFIFO_UDFL_EN	0	RW	external I2C device read TX FIFO when TX FIFO is empty
2	I2C_TFIFO_EMPTY_EN	0	RW	TX FIFO empty interrupt enable 0: disable 1: enable
1	I2C_TFIFO_UNTH_EN	0	RW	TX FIFO under threshold interrupt enable 0: disable 1: enable
0	I2C_RX_TO_EN	0	RW	RX timeout error. The time between two RX FIFO read/write exceeds timeout threshold(I2C_TO_THLD) when RX FIFO is not empty interrupt enable

				0: disable 1: enable

Table 126 I2C_INT_CLEAR (I2C_BASE_ADDR+0x10)

Bit	Field Name	Reset	RW	Description
11	I2CS_ADDR_MATCH_CLR	0	W	When slave address is received and it is matched in slave mode interrupt clear 0: invalid 1: valid
10	I2CS_IR_PHY_ERR_CLR	0	W	Slave bus error interrupt indicates slave received a START or STOP condition which occurs on the wrong position interrupt clear 0: invalid 1: valid
9	I2CM_IRQ_CLR	0	W	Transfer is completed or arbitration is lost in master mode interrupt clear 0: invalid 1: valid
8	I2C_RFIFO_UDFL_CLR	0	W	Read RX FIFO when RX FIFO is empty interrupt clear 0: invalid 1: valid
7	I2C_RFIFO_EMPTY_N_CLR	0	W	RX FIFO not empty interrupt clear 0: invalid 1: valid
6	I2C_RFIFO_FULL_CLR	0	W	RX FIFO full interrupt clear 0: invalid 1: valid
5	I2C_RFIFO_OVTH_CLR	0	W	RX FIFO over threshold interrupt clear 0: invalid 1: valid
4	I2C_TFIFO_OVFL_CLR	0	W	Write TX FIFO when TX FIFO is full interrupt clear 0: invalid 1: valid

3	I2C_TFIFO_UDFL_CLR	0	W	External I2C device read TX FIFO when TX FIFO is empty
2	I2C_TFIFO_EMPTY_CLR	0	W	TX FIFO empty interrupt clear 0: invalid 1: valid
1	I2C_TFIFO_UNTH_CLR	0	W	TX FIFO under threshold interrupt clear 0: invalid 1: valid
0	I2C_RX_TO_CLR	0	W	RX timeout error. The time between two RX FIFO read/write exceeds timeout threshold(I2C_TO_THLD) when RX FIFO is not empty interrupt clear 0: invalid 1: valid

Table 127 I2C_I2CM_CR (I2C_BASE_ADDR+0x14)

Bit	Field Name	Reset	RW	Description
7	I2CM_START_CMD	0x0	W	0: invalid 1: a START condition is performed in master mode
6	I2CM_STOP_CMD	0x0	W	0: invalid 1: a STOP condition is performed after the bytes are sent or received in master mode
5	I2CM_READ_CMD	0x0	W	0: invalid 1: a READ is performed in master mode
4	I2CM_WRITE_CMD	0x0	W	0: invalid 1: a WRITE is performed in master mode
3	I2CM_ACK_CMD	0x0	W	0: invalid 1: a ACK is performed after the bytes are sent or received in master mode
2	I2CM_RESTART_CMD	0x0	W	0: invalid 1: a RESTART is performed before the bytes are sent or received in master mode.
1:0	Reserved	0x0	W	Reserved

Table 128 I2C_CFG (I2C_BASE_ADDR+0x18)

Bit	Field Name	Reset	RW	Description

21	I2CS_NACK	0x0	RW	NACK generated by software or hardware after received data in i2c slave mode 0: hardware 1: software
20	I2CS_10BITADDR	0x0	RW	10bit or 7bit addressing mode selection in i2c slave mode
19	Reserved	0x0	RW	Reserved
18	I2C_MODE_SEL	0x0	RW	I2C mode selection 0: slave mode 1:master mode
17:16	Reserved	0x0	RW	Reserved
15:8	I2C_TO_THLD	0x0	RW	I2C RX time out threshold. Unit is (I2CM_PRER +1)*5/16ns
7:4	I2C_RFIFO_THLD	0x0	RW	I2C RX FIFO threshold, range 1~8 max FIFO depth is 8*8 bit
3:0	I2C_TFIFO_THLD	0x0	RW	I2C TX FIFO threshold, range 1~8 max FIFO depth is 8*8 bit

Table 129 I2C_I2CM_CFG (I2C_BASE_ADDR+0x1C)

Bit	Field Name	Reset	RW	Description
23:8	I2CM_PRER	0x0	RW	SCL pre-scale register in master mode when SCL is 100KHz, i2cm_prer = 16MHz/(5*100KHz) - 1 = 31 when SCL is 400KHz, i2cm_prer = 16MHz/(5*400KHz) - 1 = 7 Note: this register is RX FIFO timer out unit
7	I2CM_SCL_CLK_ENABLE	0x0	RW	Enable SCL clock generation in master mode 0: disable 1: enable
6	I2CM_10BITADDR	0x0	RW	10bit or 7bit addressing mode selection in master mode 0: 7bit 1: 10bit
5	I2CM_RESTART_RW_SEL	0x0	RW	READ/WRITE operation selection after restart in master mode 0: write 1:read
4:0	Reserved	0x0	RW	Reserved

Table 130 I2C_I2CMS_ADD (I2C_BASE_ADDR+0x20)

Bit	Field Name	Reset	RW	Description
9:0	I2C_I2CMS_ADD	0x0	RW	I2C target address in i2c master mode, I2C slave address in i2c slave mode

Table 131 I2C_BYTE_FIFO_CNT (I2C_BASE_ADDR+0x28)

Bit	Field Name	Reset	RW	Description
15:8	I2CM_BYTE_REST	0x0	R	Left transfer bytes number in i2c master mode
7:4	I2C_RFIFO_CNT	0x0	R	I2C RX FIFO counter
3:0	I2C_TFIFO_CNT	0x0	R	I2C TX FIFO counter

Table 132 I2C_I2CM_TRANSFER_LEN (I2C_BASE_ADDR+0x2C)

Bit	Field Name	Reset	RW	Description
15:8	I2CM_RESTART_LEN	0x0	RW	Byte length to be transferred after RESTART condition in i2c master mode
7:0	I2CM_START_LEN	0x0	RW	Byte length to be transferred after START Condition in i2c master mode

Table 133 I2C_TFIFO_IDLE_DATA (I2C_BASE_ADDR+0x30)

Bit	Field Name	Reset	RW	Description
7:0	I2C_TFIFO_IDLE_DATA	0x0	RW	Transmit this register (I2C_TFIFO_IDLE_DATA) value to external I2C device when TX FIFO is empty

Table 134 I2C_STATUS (I2C_BASE_ADDR+0x34)

Bit	Field Name	Reset	RW	Description
7	I2CS_RD_OP	0x0	R	I2C transmit data in slave mode, just for debug
6	I2CS_WR_OP	0x0	R	I2C receive data in slave mode, just for debug
5	I2CS_BUSY	0x0	R	I2C slave FSM busy in slave mode, just for debug
4	I2CM_BUSY	0x0	R	I2C master busy this bit becomes high after start condition, this bit becomes low after stop condition in master mode, just for debug
3	I2CM_RXACK	0x0	R	I2C master receive ACK 1: receive NACK 0:receive ACK in master mode, just for

				debug
2	I2CM_AL	0x0	R	arbitration lost in master mode
1	I2CM_TIP	0x0	R	I2C master FSM is in read/write operation in master mode, just for debug
0	I2CM_IRQ_FLAG	0x0	R	Arbitration lost or transfer finished in master mode, just for debug

Table 135 I2C_FIFO_CLR (I2C_BASE_ADDR+0x38)

Bit	Field Name	Reset	RW	Description
0	I2C_FIFO_CLR	0x0	W	I2C TX/RX FIFO data, FIFO counter and FIFO READ/WRITE address clear

5.14 7816

This smart card interface is compatible with the ISO 7816-3 and EMV 4.2 related standards. As a master device, it can transmit data controlled by CPU/DMA to destination card, and receive data to be stored in SRAM.

5.14.1 Feature

1. Support the asynchronous protocols T=0 in accordance with ISO 7816-3
2. Flexible output clock 1MHz, 2MHz, 4MHz
3. Error management at character level for T=0
4. 32-bit counting by ETU clock for time-out counter
5. Power-down mode for reducing current consumption when no activity

5.14.2 Function Description

An overview of the 7816 module is shown in figure below.

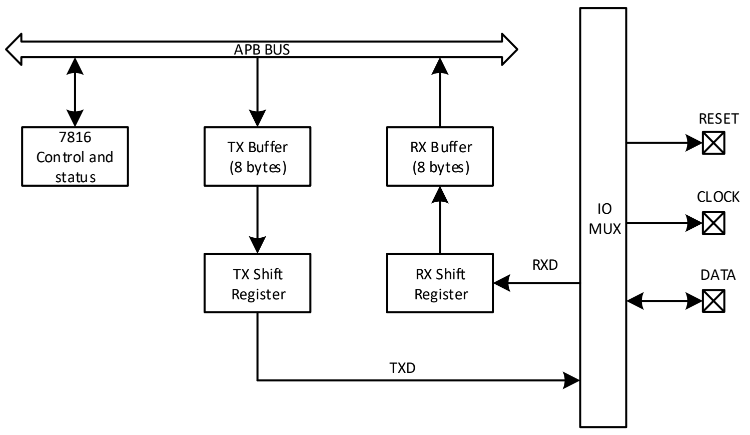


Figure 23 7816 overview

5.14.2.1 Frame Format

The character frame is shown in the figure below. The frame consists of 8 data bits, a start bit and a parity bit. If a parity error is detected by the receiver, it pulls the I/O line low in guard time. It holds the line low for one bit-period before it releases the line. In this case, the guard time is extended by one bit period before a new transmission can start.

1. Before byte transmission, I/O should be set to high
2. Start bit. When I/O change to low, transmission starts
3. Data bits. Eight bits of information, as illustrated from ba to bh
4. Parity bit. The tenth bit bi used for even parity checking
5. Guard time. TX mode, if received data from card is ‘0’, repeat the transmission of the latest byte data. RX mode, if the parity bit of received data is OK, send bit ‘1’ to card.

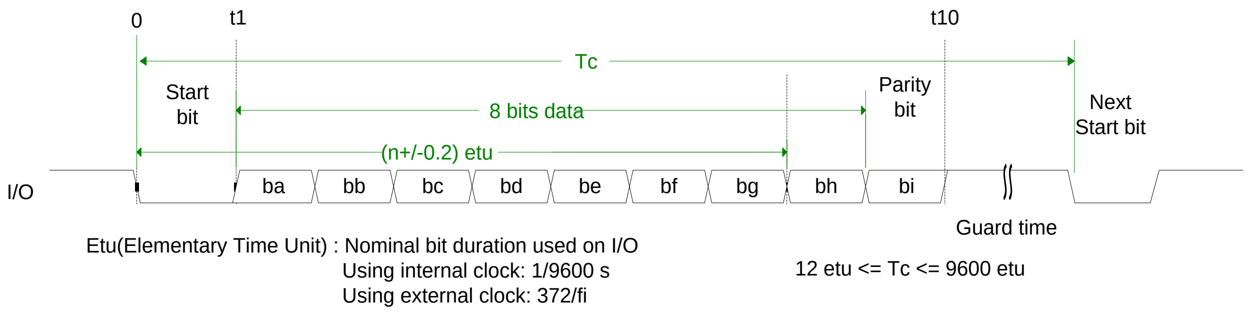


Figure 24 Character frame

There are some timing constrains for the character transmission.

1. Within a character, the time from the leading edge of the start bit to the trailing edge of the nth bit shall be equal to (n+/-0.2) etu.
2. Sampling time shall be less than 0.2 etu.
3. Delay between two consecutive characters (leading edge) is at least 12 etu and can’t exceed 9600 etu.

5.14.2.2 Cold Reset

In order to initiate an interaction with a mechanically connected smartcard, the master device shall activate the electrical circuits in sequence.

1. T0, software triggers internal state machine from idle mode to cold reset mode by setting bit 7816_MODE in 7816_BASIC_CONFIG register. In cold reset state, CLK is active and RST maintains in state L for at least 400 cycles (Tb).
2. T1, internal state machine goes into working mode. RST is set to state H by hardware. Within 400 to 40000 cycles (Tc), I/O should start answer to reset (ACK).

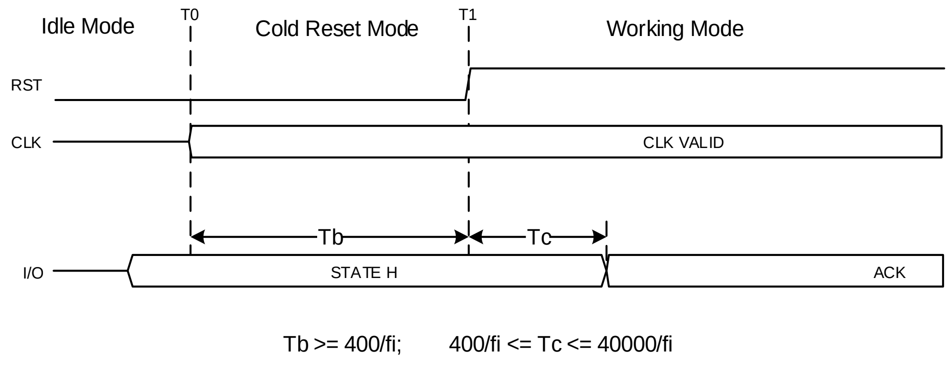


Figure 25 Cold Reset

5.14.2.3 Warm Reset

The master device may warm reset the card at any time during the working mode. The timing of warm reset is shown below.

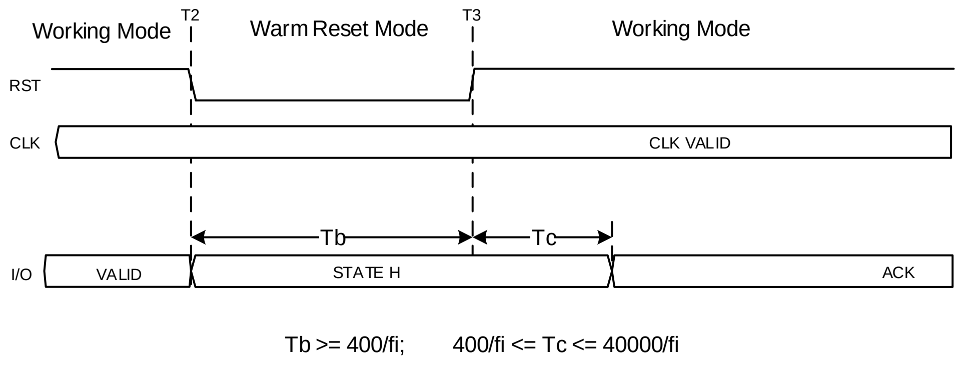


Figure 26 Warm Reset

1. T2, software triggers internal state machine from working mode to warm reset mode by configuring bit 7816_MODE in 7816_BASIC_CONFIG register. RST will be set to state L by hardware.
2. T3, internal state machine goes into working mode. RST is set to state H by hardware. Within 400 to 40000 cycles (Tc), I/O should start answer to reset (ACK).

5.14.2.4 Transmission

The 7816 module drives a synchronous clock to the smart card for communication, and the clock is generated through the internal counter. See the ISO 7816 specification for more info on this clock signal.

The next step of a transmission is storing bytes in the transmit buffer. When the transmission shift register is empty and ready for new data, a frame from the transmit buffer is loaded into the shift register, and transmission begins. When the frame has been transmitted, a new frame is loaded into the shift register if available, and transmission continues. If the transmit buffer is empty, the transmitter goes to an idle state, waiting for a new frame.

A frame can be loaded into the buffer by writing to 7816_TFIFO_WR_DATA. When writing more frames to the transmit buffer than the free space there is, the TX FIFO overflow interrupt flag in 7816_INT_STATUS will be set to indicate the overflow. The data already in the transmit buffer is preserved in this case, and no data is written in.

The transmit buffer and the shift register can be cleared by setting 7816_FIFO_CLR. This will prevent the master from transmitting the old data in the buffer and shift register. After clearing, the buffer is available for new data.

5.14.2.5 Reception

When the receiver is enabled, it actively samples the input looking for a transition from high to low indicating the start baud of a new frame. When a start baud is found, reception of the new frame begins if the receive shift register is empty and ready for new data. When the frame has been received, it is pushed into the receive buffer, making the shift register ready for new frame of data, and the receiver starts looking for another start baud. If the receive buffer is full, the received frame remains in the shift register until more space in the receive buffer is available. If an incoming frame is detected while both the receive buffer and the receive shift register are full, the data in the shift register is overwritten.

Data can be read from the receive buffer 7816_RFIFO_RD_DATA register. When data becomes available in the receive buffer, the RX FIFO not empty flag in 7816_INT_STATUS is set. When the buffer becomes full, RX FIFO overflow interrupt flag in 7816_INT_STATUS is set. To know how many bytes received in the RX buffer, the CPU can read the 7816_RFIFO_CNT field in 7816_FIFO_CNT register.

The receive buffer and the shift register can be cleared by setting 7816_FIFO_CLR. Any frame in receiving will be discarded.

5.14.2.6 Parity Error

When a parity error is detected in an incoming frame, the NAK is generated by hardware. The NAK generated by the receiver is sampled as the stop-bit of the frame. It pulls the line I/O line low after half a stop bit and holds the line low for one bit-period before it releases the line. In this case, the guard time is extended by one bit period before a new transmission can start. The transmitter will automatically retransmit the latest NACK frame. The transmitter will retransmit the frame until it is ACK from the receiver.

5.14.3 Software Procedure

5.14.3.1 CPU Operation Mode

CPU operation mode is shown below figure.

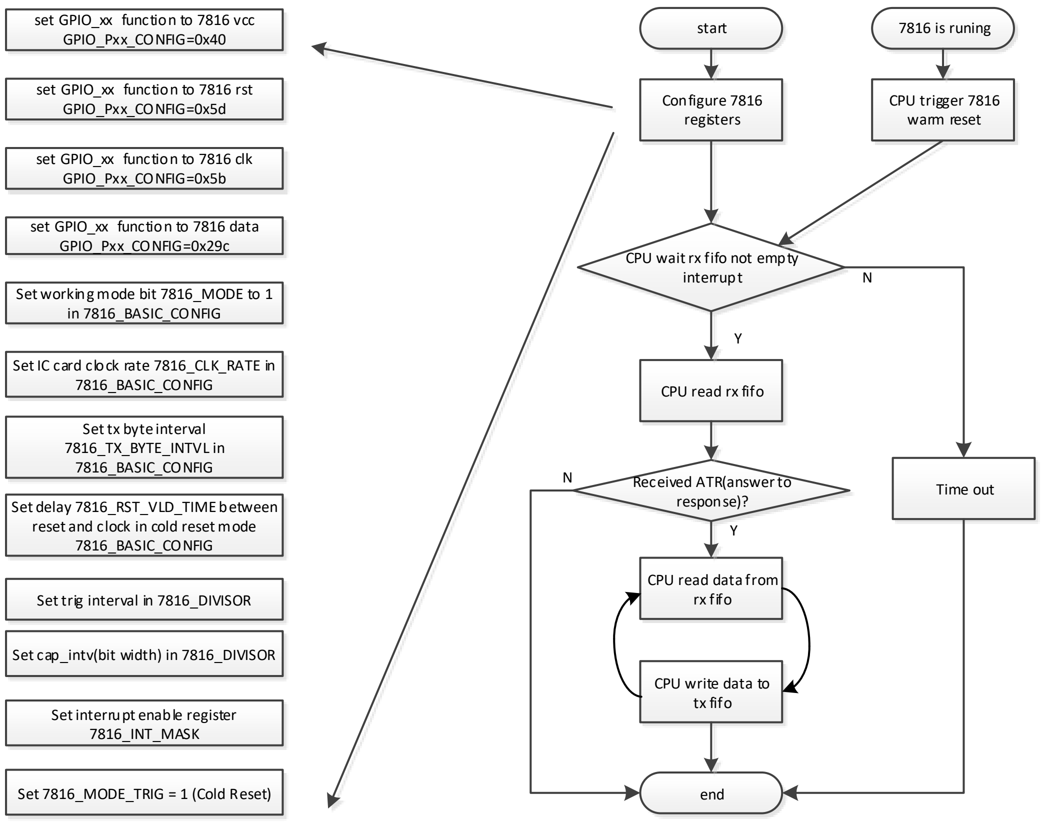

set GPIO_xx function to 7816 vcc GPIO_Pxx_CONFIG=0x40
set GPIO_xx function to 7816 rst GPIO_Pxx_CONFIG=0x5d
set GPIO_xx function to 7816 clk GPIO_Pxx_CONFIG=0x5b
set GPIO_xx function to 7816 data GPIO_Pxx_CONFIG=0x29c
Set working mode bit 7816_MODE to 1 in 7816_BASIC_CONFIG
Set IC card clock rate 7816_CLK_RATE in 7816_BASIC_CONFIG
Set tx byte interval 7816_TX_BYTE_INTVL in 7816_BASIC_CONFIG
Set delay 7816_RST_VLD_TIME between reset and clock in cold reset mode 7816_BASIC_CONFIG
Set trig interval in 7816_DIVISOR
Set cap_intv(bit width) in 7816_DIVISOR
Set interrupt enable register 7816_INT_MASK
Set 7816_MODE_TRIG = 1 (Cold Reset)
start
7816 is runing
Configure 7816 registers
CPU trigger 7816 warm reset
CPU wait rx fifo not empty interrupt
N
Y
CPU read rx fifo
Received ATR(answer to response)?
N
Y
Time out
CPU read data from rx fifo
CPU write data to tx fifo
end

Figure 27 CPU Operation Mode

5.14.3.2 DMA Operation Mode

DMA operation mode is shown below.

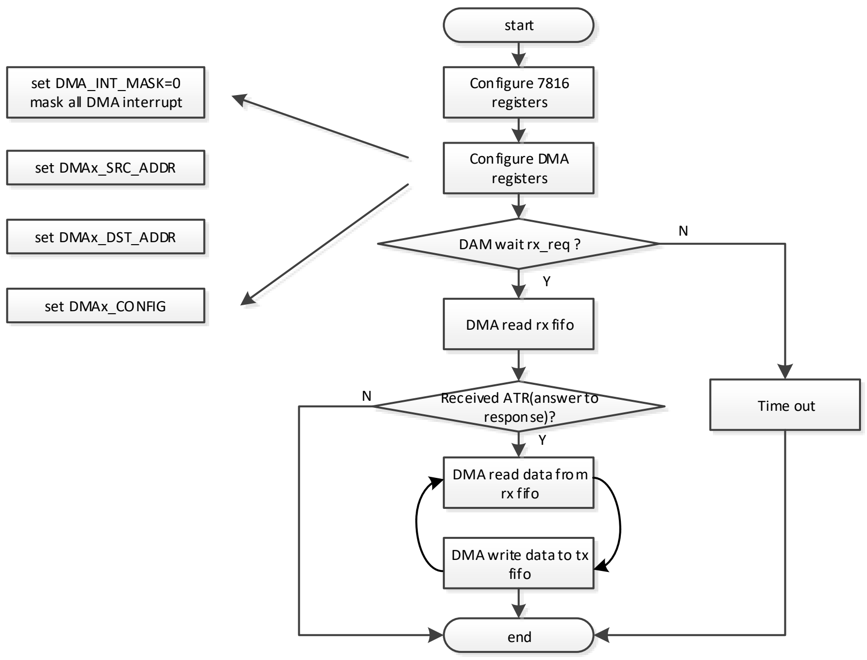


Figure 28 DMA Operation Mode

Note:

1. There are two DMA channels (DMA0-DMA1) used for 7816, and any channel can be configured to 7816 TX or RX.
2. DMAx_CONFIG registers include DMA direction, DMA transmission counter, DMAx_DST_SIZE and DMAx_SRC_SIZE. Refer to DMA registers for more details.

5.14.4 7816 interrupt

7816 interrupt table is shown below.

Table 136 7816 Interrupt

Interrupt Number	Interrupt name	Description
0	TIMEOUT	RX FIFO is not empty, and no further data is received.
1	TX_FIFO_UNDER_THLD	TX FIFO data number is under threshold.
2	TX_FIFO_EMPTY	TX FIFO is empty.
3	RX_FIFO_OVER_THLD	RX FIFO data number is over threshold.
4	RX_FIFO_OVERFLOW	RX FIFO data number is overflow.
5	RX_FIFO_NOT_EMPTY	RX FIFO is not empty.
6	RX_FIFO_UNDERFLOW	RX FIFO data number is under threshold.
7	TX_FIFO_OVERFLOW	TX FIFO data number is overflow.

5.14.4.1 Timeout interrupt

The receive timeout interrupt is asserted when the RX FIFO is not empty, and no further data is received over a programmable timeout period (7816_TO_THLD in 7816_CONFIG register). This mechanism notifies the user to be aware that data is still present in the RX FIFO requiring service. The receive timeout interrupt is cleared when a 1'b1 is written to the corresponding bit of the 7816_INT_CLEAR register.

5.14.4.2 Transmit Interrupt

The transmit interrupt is asserted HIGH when one of the following conditions occurs:

- TX_FIFO_EMPTY
  The interrupt is set when no data written in TX FIFO, or written data leaves the TX FIFO and it becomes empty.
- TX_FIFO_UNDER_THLD
  If the number of characters in the TX FIFO is less than the register (7816_TFIFO_THLD), this bit is asserted.
- TX_FIFO_OVERFLOW
  If the written data in TX FIFO is full (the number is eight), a next write will trigger the bit to high.

5.14.4.3 Receive Interrupt

The receive interrupt is asserted HIGH when one of the following conditions occurs:

- RX_FIFO_NOT_EMPTY
  The interrupt is set when RXD data is received in RX FIFO. It becomes not empty.
- RX_FIFO_UNDERFLOW
  If RX FIFO is empty, a new read command from CPU or DMA will trigger this bit to high.
- RX_FIFO_OVERFLOW
  If RX FIFO is full (the data number is eight), a new write will trigger this bit to high.
- RX_FIFO_OVERF_THLD
  If the number of characters in the RX FIFO is larger than the register (7816_RFIFO_THLD), this bit is asserted.

5.14.5 7816 Registers

Table 137 Instance

Base Address	Peripheral	Instance	Description
0x40012800	7816	7816	Smart Card (ISO7816)

Table 138 7816 Register Map

Address Offset	Name	Description
0x00	7816_TFIFO_WR_DATA	7816 TX FIFO Write Register
0x04	7816_RFIFO_RD_DATA	7816 RX FIFO Read Register

0x08	7816_INT_STATUS	7816 Interrupt Status
0x0C	7816_INT_EN	7816 Interrupt Enable
0x10	7816_INT_CLEAR	7816 Interrupt Flag Clear Register
0x14	7816_BASIC_CONFIG	7816 Basic Configuration Register
0x18	7816_MODE_TRIG	7816 Mode Trigger
0x1C	7816_CONFIG	7816 Configuration Register
0x20	7816_DIVISOR	7816 Data Divisor
0x24	7816_FIFO_CNT	7816 FIFO Counter
0x28	7816_FIFO_CLR	7816 FIFO and Data Clear

Table 139 7816_TFIFO_WR_DATA (7816_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
7:0	7816_TFIFO_WR_DATA	0x0	W	DMA/CPU write 7816 transmit fifo data via this register

Table 140 7816_RFIFO_RD_DATA (7816_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
7:0	7816_RFIFO_RD_DATA	0x0	R	DMA/CPU read 7816 receive fifo data via this register

Table 141 7816_INT_STATUS (7816_BASE_ADDR+0x08)

Bit	Field Name	Reset	RW	Description
7	7816_TX_FIFO_OVERFLOW	0	R	TX FIFO overflow 0: invalid 1: valid
6	7816_RX_FIFO_UNDERFLOW	0	R	RX FIFO underflow 0: invalid 1: valid
5	7816_RX_FIFO_NOT_EMPTY	0	R	RX FIFO not empty 0: invalid 1: valid
4	7816_RX_FIFO_OVERFLOW	0	R	RX FIFO overflow 0: invalid 1: valid
3	7816_RX_FIFO_OVER_THLD	0	R	RX FIFO over threshold 0: invalid 1: valid
2	7816_TX_FIFO_EMPTY	0	R	TX FIFO empty 0: invalid 1: valid
1	7816_TX_FIFO_UNDER_THLD	0	R	TX FIFO under threshold

				0: invalid 1: valid
0	7816_TIMEOUT	0	R	RX timeout error. The time between two RX FIFO read/write exceeds timeout threshold (7816_TO_THLD) when RX FIFO is not empty. 0: invalid 1: valid

Table 142 7816_INT_MASK (7816_BASE_ADDR+0x0C)

Bit	Field Name	Reset	RW	Description
7	7816_TX_FIFO_OVERFLOW_MASK	0	R	TX FIFO overflow interrupt enable 0: disable 1: enable
6	7816_RX_FIFO_UNDERFLOW_MASK	0	R	RX FIFO underflow interrupt enable 0: disable 1: enable
5	7816_RX_FIFO_NOT_EMPTY_MASK	0	R	RX FIFO not empty interrupt enable 0: disable 1: enable
4	7816_RX_FIFO_OVERFLOW_MASK	0	R	RX FIFO overflow interrupt enable 0: disable 1: enable
3	7816_RX_FIFO_OVER_THLD_MASK	0	R	RX FIFO over threshold interrupt enable 0: disable 1: enable
2	7816_TX_FIFO_EMPTY_MASK	0	R	TX FIFO empty interrupt enable 0: disable 1: enable
1	7816_TX_FIFO_UNDER_THLD_MASK	0	R	TX FIFO under threshold interrupt enable 0: disable 1: enable
0	7816_TIMEOUT_MASK	0	R	RX timeout interrupt enable 0: disable 1: enable

Table 143 7816_INT_CLEAR (7816_BASE_ADDR+0x10)

Bit	Field Name	Reset	RW	Description
7	7816_TX_FIFO_OVERFLOW_CLEAR	0	W	TX FIFO overflow interrupt clear 0: invalid 1: clear

6	7816_RX_FIFO_UNDERFLOW_CLEAR	0	W	RX FIFO underflow interrupt clear 0: invalid 1: clear
5	7816_RX_FIFO_NOT_EMPTY_CLEAR	0	W	RX FIFO not empty interrupt clear 0: invalid 1: clear
4	7816_RX_FIFO_OVERFLOW_CLEAR	0	W	RX FIFO overflow interrupt clear 0: invalid 1: clear
3	7816_RX_FIFO_OVER_THLD_CLEAR	0	W	RX FIFO over threshold interrupt clear 0: invalid 1: clear
2	7816_TX_FIFO_EMPTY_CLEAR	0	W	TX FIFO empty interrupt clear 0: invalid 1: clear
1	7816_TX_FIFO_UNDER_THLD_CLEAR	0	W	TX FIFO under threshold interrupt clear 0: invalid 1: clear
0	7816_TIMEOUT_CLEAR	0	W	RX timeout interrupt clear 0: invalid 1: clear

Table 144 7816_BASIC_CONFIG (7816_BASE_ADDR+0x14)

Bit	Field Name	Reset	RW	Description
31:20	7816_RST_VLD_TIME	0x1FF	RW	The time of reset (active low). The clock should toggle during the reset time.
19:6	7816_TX_BYTE_INTVL	0x10	RW	interval between two adjacent TX data It must be set larger than ten ETU number time.
5:3	7816_CLK_RATE	0x0	RW	target device clock frequency 0: 1MHz 1: 2MHz 2~3: 4MHz
2:0	7816_MODE	0x0	RW	working mode command 0: idle 1: cold reset 2: warm reset 3~7: reserved

Table 145 7816_MODE_TRIG (7816_BASE_ADDR+0x18)

Bit	Field Name	Reset	RW	Description
0	7816_MODE_TRIG	0	W	working mode trigger

				0: invalid 1: trigger

Table 146 7816_CONFIG (7816_BASE_ADDR+0x1C)

Bit	Field Name	Reset	RW	Description
19	7816_MSB	0	RW	big-endian mode or little-endian mode selection 0: little-endian 1: big-endian
18	7816_PARITY_MODE	0	RW	parity mode 0: parity odd 1: parity even
17:16	7816_BYTE_SIZE	0x3	RW	transmitted data size 0: 5 bit 1: 6 bit 2: 7 bit 3: 8 bit
15:8	7816_TO_THLD	0xFF	RW	RX time out threshold unit is 1 symbol (byte)
7:4	7816_RFIFO_THLD	0x4	RW	RX FIFO threshold When the data number in rx fifo arrived at RX threshold, interrupt is generated.
3:0	7816_TFIFO_THLD	0x4	RW	TX FIFO threshold When the data number in tx fifo arrived at TX threshold, interrupt is generated.

Table 147 7816_DIVISOR (7816_BASE_ADDR+0x20)

Bit	Field Name	Reset	RW	Description
25:16	7816_TRIG_INTV	0x6	RW	input RXD detected interval used to detect start edge, should be set (CAP_INTV*(16MHz/CLK_RATE))*0.2/7
15:0	7816_CAP_INTV	0x89	RW	data capture interval CAP_INTV = F/D (F: clock rate conversion inter; D: baud rate adjustment integer)

Table 148 7816_FIFO_CNT (7816_BASE_ADDR+0x24)

Bit	Field Name	Reset	RW	Description
7:4	7816_RFIFO_CNT	0x0	R	rx fifo counter
3:0	7816_TFIFO_CNT	0x0	R	tx fifo counter

Table 149 7816_FIFO_CLR (7816_BASE_ADDR+0x28)

Bit	Field Name	Reset	RW	Description

0	7816_FIFO_CLR	0	W	counter (rx and tx) and data (rx and tx) both cleared 0: invalid 1: clear

5.15 QDEC

The QDEC provides buffered decoding of quadrature-encoded sensor signals, which is from mechanical and optical sensors with an optional LED output signal. The sample period and accumulation can be configured to match the application requirements.

5.15.1 Feature

1. supports double transition detection
2. Optional input digital filters
3. Optional LED output for encoder

5.15.2 Function Description

5.15.2.1 Sampling and Decoding

The QDEC decodes the output from an incremental motion encoder by sampling the QDEC phase input pins (A and B). Furthermore, the signed counter value represents the number of steps moved.

The off-chip quadrature encoder is an incremental motion encoder outputting two waveforms, phase A and phase B. The two output waveforms are always 90 degrees out of phase, meaning that one always changes level before the other. The direction of movement is indicated by the signal (A or B) which changes level first. Channels are expected to provide a pulse train with 90 degrees rotation as displayed in the figures below.

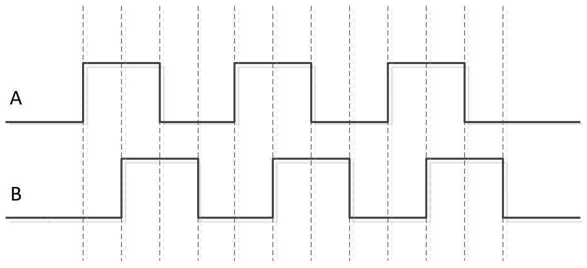


Figure 29 Moving Forward

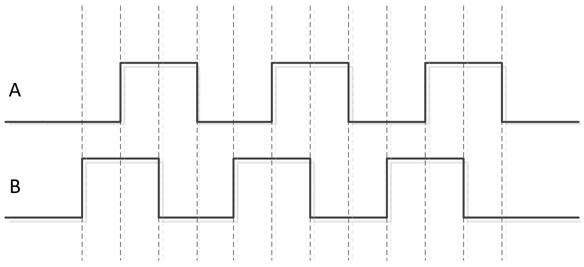


Figure 30 Moving Backwards

Invalid transitions may occur, that is when the two waveforms switch simultaneously. This may occur if the wheel rotates too fast relative to the sample rate set for the decoder as shown in the figures below.

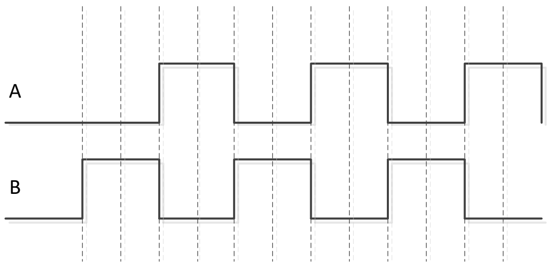


Figure 31 Double Transition Case 1

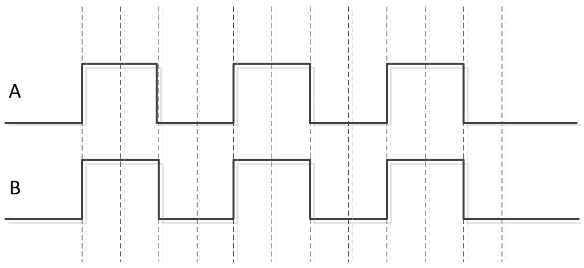


Figure 32 Double Transition Case 2

The QDEC decodes the output from the off-chip encoder by sampling the QDEC phase input pins (A and B) at a fixed rate as specified in the QDEC_SAMPLE_PER register.

If the QDEC_SAMPLE_PER value needs to be changed, the QDEC shall be stopped by setting register QDEC_EN to 0.

When started, the decoder continuously samples the two input waveforms and decodes these by comparing the current sample pair (n) with the previous sample pair (n-1). The decoding of the sample pairs is described in the table below.

Table 150 Sampled Value Encoding

Phase n-1		Phase n		QDEC_SAMPLE_RLT	QDEC_ACC	QDEC_ACC_DBL	Description
A	B	A	B				
0	0	0	0	0	No Change	No Change	No movement
0	0	1	0	1	+1	No Change	Positive direction
0	0	0	1	-1	-1	No Change	Negative direction

0	0	1	1	2	No Change	+1	Double transition
0	1	0	0	1	+1	No Change	Positive direction
0	1	1	0	2	No Change	+1	Double transition
0	1	0	1	0	No Change	No Change	No movement
0	1	1	1	-1	-1	No Change	Negative direction
1	1	0	0	2	No Change	+1	Double transition
1	1	1	0	-1	-1	No Change	Negative direction
1	1	0	1	1	+1	No Change	Positive direction
1	1	1	1	0	No Change	No Change	No movement
1	0	0	0	-1	-1	No Change	Negative direction
1	0	1	0	0	No Change	No Change	No movement
1	0	0	1	2	No Change	+1	Double transition
1	0	1	1	+1	+1	No Change	Positive direction

5.15.2.2 LED output

The LED output follows the sample period. The LED is switched on a given period before sampling and switched off immediately after the inputs are sampled. The period the LED is switched on before sampling is given in the QDEC_LED_PER register and the LED output pin polarity is specified in the QDEC_LED_POL register.

For using off-chip mechanical encoders not requiring a LED, the LED output can be disabled. In this case the QDEC will not acquire access to a LED output pin and the pin can be used for other purposes.

5.15.2.3 Inputs Digital Filters

The two-phase inputs have digital debounce filters. When enabled through the QDEC_DEB_EN register, the filter inputs are sampled at a fixed 16 MHz frequency during the entire sample period (which is specified in the QDEC_SAMPLE_PER register). All of the samples within this sample period are filtered before the input signal is accepted and transferred to the output of the filter.

As a result, only input signal with a steady state longer than the period specified in QDEC_SAMPLE_PER is guaranteed to pass through the filter. Any signal with a steady state shorter than QDEC_SAMPLE_PER will always be suppressed by the filter.

The LED will always be ON when the debounce filters are enabled, as the inputs in this case will be sampled continuously.

5.15.2.4 Accumulators

The quadrature decoder contains two accumulator registers, QDEC_ACC and QDEC_ACC_DBL, that accumulate respectively valid motion sample values and invalid sample values (double transitions).

The QDEC_ACC register accumulates with one valid value (1/-1) every period (QDEC_SAMPLE_PER). A QDEC_ACC_OVERFLOW interrupt will be generated if the QDEC_ACC exceeds the threshold register (QDEC_SAMPLE_ACC).

The accumulator QDEC_ACC_DBL accumulates the number of detected double transitions since the previous clearing of the QDEC_ACC_DBL register.

The accumulator QDEC_ACC_NO_CHANGE accumulates the number of detected no movement transitions, and it will generate ACC_NO_CHANGE_OVERFLOW interrupt when it exceeds the threshold register (QDEC_SAMPLE_ACC_NO_CHANGE).

The QDEC_ACC and QDEC_ACCDBL registers can be cleared by the QDEC_CLR.

5.15.3 Software Procedure

QDEC working flow is shown in figure below.

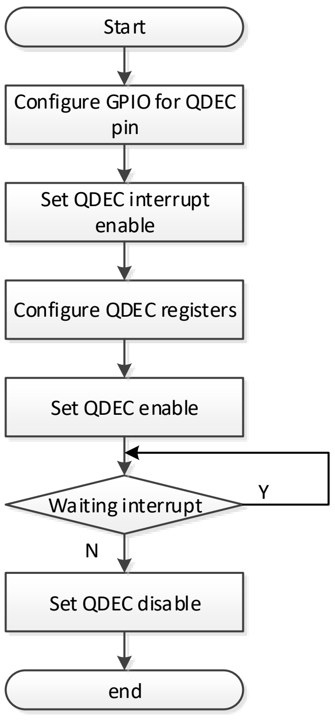


Figure 33 QDEC working flow

Note:

1. Configured QDEC registers include
   - QDEC_SAMPLE_PER, the period of sampling the QDEC phase input pins (A and B)
   - QDEC_SAMPLE_ACC, the threshold of accumulating the valid transitions
   - QDEC_SAMPLE_ACC_DBL, the threshold of accumulating the valid transitions
   - QDEC_SAMPLE_ACC_NO_CHANGE, the threshold of accumulating the no movement transitions
2. Waiting interrupt
   - If received a positive or a negative direction interrupt, CPU reads QDEC_ACC register to get the number of a valid transitions.
   - If received a sleep interrupt, CPU can go into sleep status.

5.15.4 QDEC Interrupt

QDEC interrupt table is shown below.

Table 151 QDEC Interrupt

Interrupt Number	Interrupt name	Description
0	POS_DIR	Detect input phases is positive direction.
1	NEG_DIR	Detect input phases is negative direction.
2	DOUBLE_TRANS	Double transition detected
3	ACC_OVERFLOW	Accumulated the valid transition is overflow
4	ACC_DBL_OVERFLOW	Accumulated the double transition is overflow
5	ACC_NO_CHANGE_OVERFLOW	Accumulated no movement transition is overflow

5.15.5 QDEC Registers

Table 152 QDEC Instance

Base Address	Peripheral	Instance	Description
0x40011400	QDEC	QDEC	Quadrature Decoder

Table 153 QDEC Register Map

Address Offset	Name	Description
0x00	QDEC_INT_STATUS	QDEC Interrupt Status
0x04	GPIO_INT_EN	QDEC Interrupt Enable
0x08	GPIO_INT_CLEAR	QDEC Interrupt Flag Clear Register
0x0C	QDEC_CTRL	QDEC Control
0x10	QDEC_LED_POL	QDEC LED Output Polarity
0x14	QDEC_LED_PER	QDEC LED Period
0x18	QDEC_SAMPLE_PER	QDEC Sample Period
0x1C	QDEC_SAMPLE_ACC	QDEC Valid Transition Sample Accumulator

		Threshold
0x20	QDEC_SAMPLE_ACC_DBL	QDEC Double Transition Accumulator Threshold
0x24	QDEC_ACC_NO_CHANGE	QDEC No Movement Accumulator Threshold
0x28	QDEC_SAMPLE_RLT	QDEC Last Motion Status
0x2C	QDEC_CLR	QDEC Register Clear
0x30	QDEC_ACC	QDEC Valid Transition Sample Accumulator
0x34	QDEC_ACC_DBL	QDEC Double Transition Accumulator
0x38	QDEC_ACC_NO_CHANGE	QDEC No Movement Accumulator

Table 154 QDEC_INT_STATUS (QDEC_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
5	QDEC_NO_CHANGE_OVERFLOW	0	R	Accumulated no movement transition is overflow 0: invalid 1: valid
4	QDEC_ ACC_DBL_OVERFLOW	0	R	Accumulated the double transition is overflow 0: invalid 1: valid
3	QDEC_ ACC_OVERFLOW	0	R	Accumulated the valid transition is overflow 0: invalid 1: valid
2	QDEC_ DOUBLE_TRANS	0	R	Double transition detected 0: invalid 1: valid
1	QDEC_NEG_DIR	0	R	Detect input phases is negative direction 0: invalid 1: valid
0	QDEC_POS_DIR	0	R	Detect input phases is positive direction 0: invalid 1: valid

Table 155 QDEC_INT_EN (QDEC_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
5	QDEC_NO_CHANGE_OVERFLOW_EN	0	RW	Accumulated no movement transition overflow interrupt enable 0: disable 1: enable
4	QDEC_ ACC_DBL_OVERFLOW_EN	0	RW	Accumulated the double transition overflow interrupt enable

				0: disable 1: enable
3	QDEC_ ACC_OVERFLOW_EN	0	RW	Accumulated the valid transition overflow interrupt enable 0: disable 1: enable
2	QDEC_ DOUBLE_DETECT_EN	0	RW	Double transition detected interrupt enable 0: disable 1: enable
1	QDEC_NEG_DIR_EN	0	RW	Detect input phases negative direction interrupt enable 0: disable 1: enable
0	QDEC_POS_DIR_EN	0	RW	Detect input phases positive direction interrupt enable 0: disable 1: enable

Table 156 QDEC_INT_CLEAR (QDEC_BASE_ADDR+0x08)

Bit	Field Name	Reset	RW	Description
5	QDEC_NO_CHANGE_OVERFLOW_CLEAR	0	W	Accumulated no movement transition overflow interrupt enable 0: disable 1: enable
4	QDEC_ ACC_DBL_OVERFLOW_CLEAR	0	W	Accumulated the double transition overflow interrupt enable 0: disable 1: enable
3	QDEC_ ACC_OVERFLOW_CLEAR	0	W	Accumulated the valid transition overflow interrupt enable 0: disable 1: enable
2	QDEC_ DOUBLE_DETECT_CLEAR	0	W	Double transition detected interrupt enable 0: disable 1: enable
1	QDEC_NEG_DIR_CLEAR	0	W	Detect input phases negative direction interrupt enable 0: disable 1: enable
0	QDEC_POS_DIR_CLEAR	0	W	Detect input phases positive direction

				interrupt enable 0: disable 1: enable

Table 157 QDEC_CTRL (QDEC_BASE_ADDR+0x0C)

Bit	Field Name	Reset	RW	Description
2	QDEC_DIR_POL	0	RW	QDEC direction detection 0: positive direction is 00/01/11/10/00 1: positive direction is 00/10/11/01/00
1	QDEC_DEB_EN	0	RW	input debounce filters enable control 0: disable 1: enable
0	QDEC_EN	0	RW	QDEC module enable 0: disable 1: enable

Table 158 QDEC_LED_POL (QDEC_BASE_ADDR+0x10)

Bit	Field Name	Reset	RW	Description
0	QDEC_LED_POL	0	RW	LED output pin polarity 0: LED active on output pin low 1: LED active on output pin high

Table 159 QDEC_LED_PER (QDEC_BASE_ADDR+0x14)

Bit	Field Name	Reset	RW	Description
20:0	QDEC_LED_PER	0xFF	RW	time period the LED is switched on, unit is 62.5ns

Table 160 QDEC_SAMPLE_PER (QDEC_BASE_ADDR+0x18)

Bit	Field Name	Reset	RW	Description
3:0	QDEC_SAMPLE_PER	0x0	RW	QDEC sample period 0: 128us 1: 256us 2: 512us 3: 1024us 4: 2048us 5: 4096us 6: 8192us 7: 16384us 8: 32768us 9: 65536us 10: 131072us 11~15: reserved

Table 161 QDEC_SAMPLE_ACC (QDEC_BASE_ADDR+0x1C)

Bit	Field Name	Reset	RW	Description
7:0	QDEC_SAMPLE_ACC	0xFF	RW	register (QDEC_ACC) sampling threshold value

Table 162 QDEC_SAMPLE_ACC_DBL (QDEC_BASE_ADDR+0x20)

Bit	Field Name	Reset	RW	Description
7:0	QDEC_SAMPLE_ACC_DBL	0xFF	RW	register (QDEC_ACC_DBL) sampling threshold value

Table 163 QDEC_SAMPLE_ACC_NO_CHANGE (QDEC_BASE_ADDR+0x24)

Bit	Field Name	Reset	RW	Description
7:0	QDEC_SAMPLE_ACC_NO_CHANGE	0xFF	RW	register (QDEC_ACC_NO_CHANGE) sampling threshold value

Table 164 QDEC_SAMPLE_RLT (QDEC_BASE_ADDR+0x28)

Bit	Field Name	Reset	RW	Description
2:0	QDEC_SAMPLE_RLT	0x0	R	last motion sample 000: no movement 001: positive direction 010: double transition detected 111: negative direction other: invalid

Table 165 QDEC_CLR (QDEC_BASE_ADDR+0x2C)

Bit	Field Name	Reset	RW	Description
2	QDEC_ACC_NO_CHANGE_CLR	0	W	register (QDEC_ACC_NO_CHANGE) clear 0: invalid 1: clear
1	QDEC_ACC_DBL_CLR	0	W	register (QDEC_ACC_DBL) clear 0: invalid 1: clear
0	QDEC_ACC_CLR	0	W	register (QDEC_ACC) clear 0: invalid 1: clear

Table 166 QDEC_ACC (QDEC_BASE_ADDR+0x30)

Bit	Field Name	Reset	RW	Description
7:0	QDEC_ACC	0x0	R	counter for accumulating the valid transitions bit[8] is a signed bit

				bit[8]=0, indicates a positive direction bit[8]=1, indicates a negative direction bit[7:0] is a counter

Table 167 QDEC_ACC_DBL (QDEC_BASE_ADDR+0x34)

Bit	Field Name	Reset	RW	Description
7:0	QDEC_ACC_DBL	0x0	R	counter for accumulating the number of detected double transitions

Table 168 QDEC_ACC_NO_CHANGE (QDEC_BASE_ADDR+0x38)

Bit	Field Name	Reset	RW	Description
7:0	QDEC_ACC_NO_CHANGE	0x0	R	counter for accumulating the time of no change

5.16 Audio

There are three types of audio input interface, digital microphone (DMIC), analog microphone (AMIC) and I2S (Inter-IC Sound). The audio interface and data path are as below block diagrams.

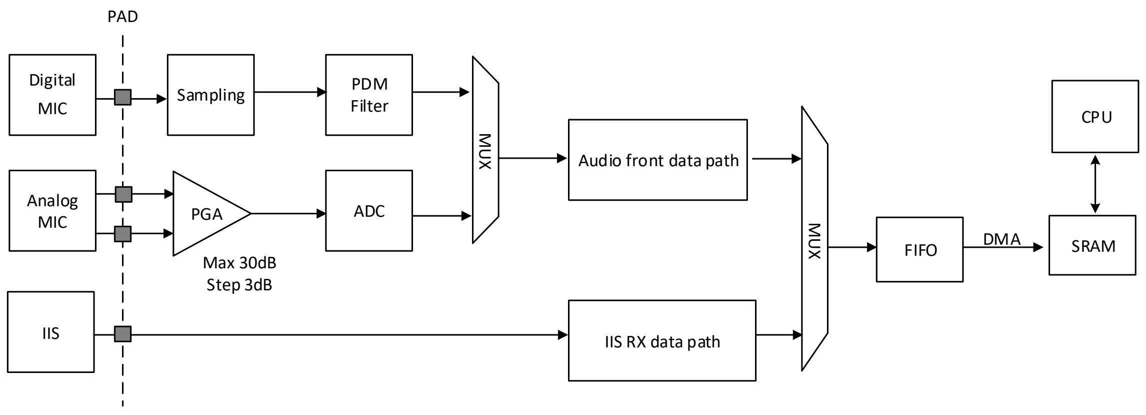


Figure 34 Audio Interface and Data Path Overview

5.16.1 Feature

1. Support PDM (pulse density modulation) mode
2. Support external analog MIC with internal audio ADC
3. Support I2S (Inter-IC Sound) mode
4. Support hardware decimation filter
5. Support hardware DC removal algorithm
6. Received Data to SRAM through DMA

7. Support 0dB~30dB analog PGA
8. Support digital PGA
9. Audio microphone bias voltage adjustable

5.16.2 Audio Interface Mode

There are three types of audio interface, and it's selected by the registers below.

Table 169 Audio Interface Mode

Audio interface mode	PMD_MODE	PDM_EN
I2S mode	0	x
ADC mode	1	0
PDM mode	1	1

5.16.3 PDM Interface

The pulse density modulation (PDM) module enables input of pulse density modulated signals from external audio device, for example, digital microphone. The PDM module generates the PDM clock and supports single-channel data input. This interface can be used to receive audio sample streams from external audio ADC output device.

PDM_CLK frequency is fixed at 2MHz, and it's shared with I2S_MCLK. PDM_DATA input pin is shared with I2S_DATA. GPIO IO mux needs to be set ready before PDM works. By default, bit from the PDM microphone is sampled on PDM_CLK rising edge (delay 3 cycle of 16M clock domain), and it can be configured to take falling edge by register PDM_SAMPLE_EDGE.

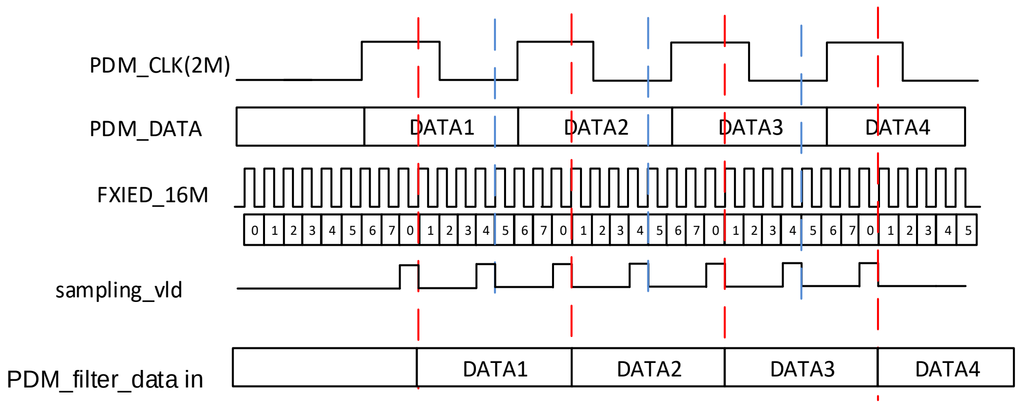


Figure 35 PDM data format

5.16.4 Audio ADC

The audio ADC is a differential successive analog-to-digital converter. It can connect with external analog MIC.

Feature

1. 16bit audio ADC, 74dB THD+N
2. Single-ended (P0) or differential (P0, P1) analog MIC input
3. Configurable 0dB~30dB analog PGA with 3dB step
4. Provide BIAS for external MIC through P4
5. Optional BIAS reference voltage filter capacitance through P2

By default, PGA is configured in differential mode, and single-ended mode can be set via register AUDIO_PGA_SINGLE_EN. MIC reference voltage can be configured via register MIC_BIAS_REF_SEL. When used as MIC input, MIC BIAS or BIAS reference voltage filter capacitance, the corresponding IO (P0, P1, P2, P4) should be set in high-z state (IE=0, OE=0, PU=0, PD=0). In single-ended mode, P1 can be used as GPIO.

5.16.5 I2S Interface

This interface can be used to receive audio sample streams from external audio ADC output device.

Feature

1. Master/slave mode configurable
2. I2S/LJF/DSP format configurable
3. Frame length configurable (16/20/24/32bit)
4. Master clock frequency configurable (2/4/8/16MHz)
5. Sampling frequency configurable (8/16/32KHz)
6. Receive function only

5.16.5.1 Function Description

The I2S interface consists of the signals shown in Figure below. The master provides the clock signals to the slave, including Word Clock (WCLK) and Bit Clock (BCLK). Audio data is transferred serially on the data line, SDATA. An optional master clock (MCLK) signal can be generated by MXD2656A1, and MCLK can be used as the main clock for external audio codec.

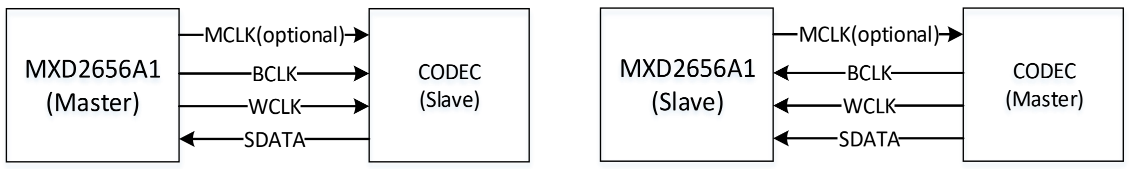


Figure 36 I2S interface signals

5.16.5.1.1 Clock Configuration

The register (MASTER_SLAVE) selects an internal or external BCLK/WCLK source for the I2S module. The WCLK source (internal or external) can be inverted using the register (WCLK_INV).

In master mode

- WCLK = BCLK / 32
- BCLK = XO16M / 512 or 1024 or 2048 (see register FM_NUM)

In master or slave mode

- MCLK = XO16M / 1 or 2 or 4 or 8 (see register MCLK_DIV)

NOTE: Data is sampled at the rising edge of BCLK and updated at the falling edge of BCLK (when SMPL_EDGE is set to '1').

5.16.5.2 Serial Interface Formats

I2S Format

In I2S format, the MSB of the left channel is valid at the second rising edge of the bit clock after the falling edge of the word clock. Similarly, the MSB of the right channel is valid at the second rising edge of the bit clock after the rising edge of the word clock (when WCLK_INV is set to '0').

The I2S interface automatically captures the data bit in the WCLK period.

- If the sample resolution is higher than the number of bits per WCLK period, the samples are truncated.
- If the sample resolution is lower than the number of bits per WCLK period, the samples are zero padded.

The maximum number of bits per word is specified using the DATA_LENGTH register. The number of BCLK cycles in a phase must be equal to or higher than this number.

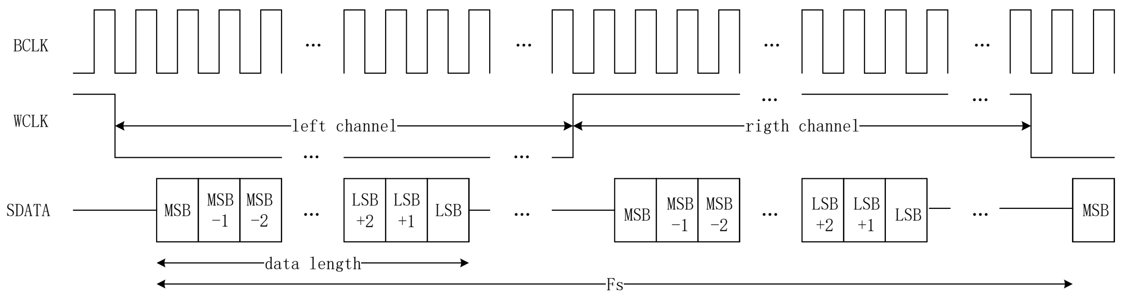


Figure 37 I2S format

LJF (Left Justified Format)

Left-justified format is similar to I2S format, and the only difference is that the MSB of the left channel is valid at the rising edge of the bit clock following the rising edge of the word clock. Similarly, the MSB of the right channel is valid at the rising edge of the bit clock following the falling edge of the word clock (when WCLK_INV is set to '1').

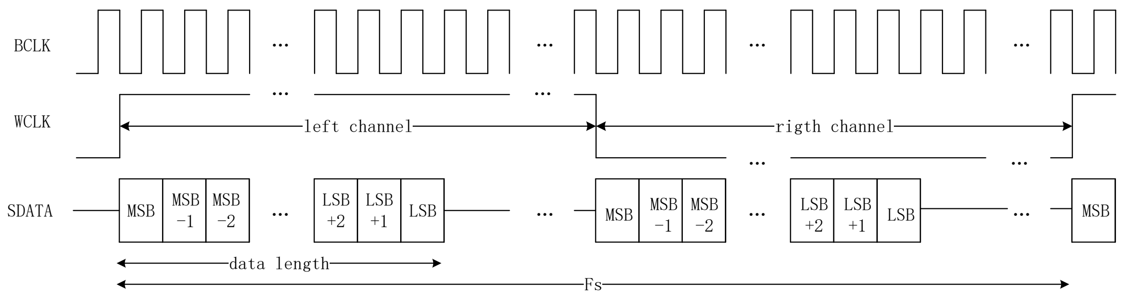


Figure 38 LJF format

DSP-A Format

DSP-a format is a single-phase format, where WCLK is high for one BCLK period. There is an optional IDLE period at the end of the clock phase between the last data channel and the next WCLK period; logical ‘0’ is output during this period.

The number of BCLK cycles in the phase must be equal to or larger than twice of the word length, as specified in the DATA_LENGTH register.

The number of BCLK periods between a WCLK rise edge and MSB of the first word in a channel is 0.

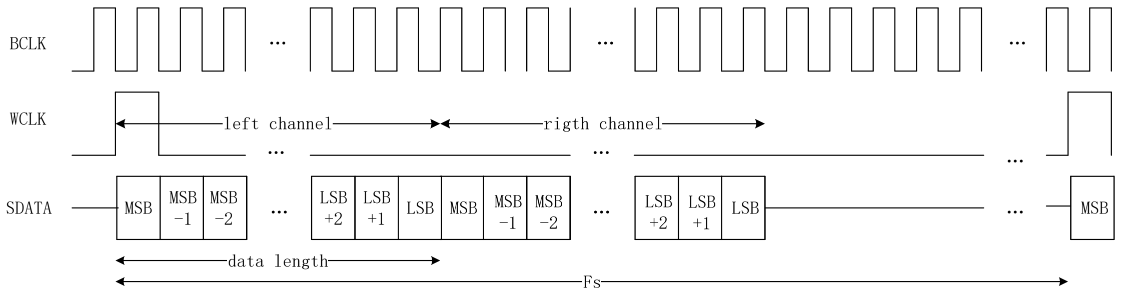


Figure 39 DSP-A format

DSP-B Format

DSP-B format is similar to DSP-A format, and the only difference is that the number of BCLK periods between a WCLK rise edge and MSB of the first word in a channel is 1.

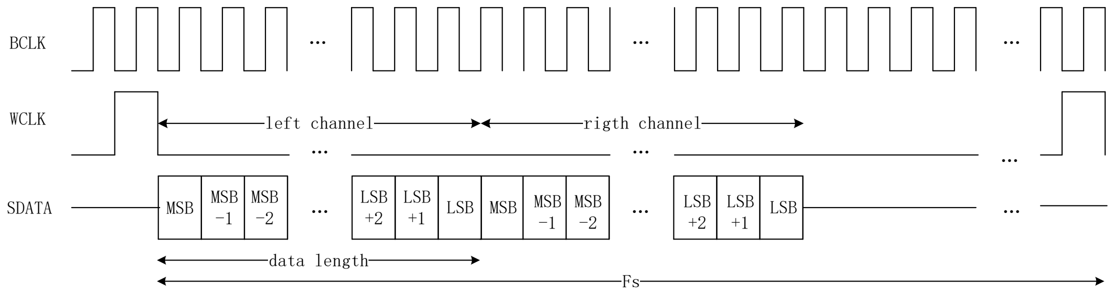


Figure 40 DSP-B format

5.16.6 Audio Front Data Path

Audio front data path processor includes DC Removal (DCR), CIC down-sample filter, compensation filter and strong low pass filter.

- DCR is used to remove DC of input audio signal.
- CIC filter is used to decimate the input audio signal and +/-0.03dB ripple could be obtained after its corresponding compensation filter.
- Efficient low-cost SLPF (strong low pass filter) is with 6.8~8.0KHz transition band.

In order to avoid too big DC in the input audio signal especially for analog MIC, DCR (DC removal) operation is added. DCR input data source can be selected from ADC or PDM filter. In ADC mode, input data stream is 16MHz sampling rate, and in PDM mode it’s 4MHz sampling rate. SLPF output sampling rate is always 16KHz.

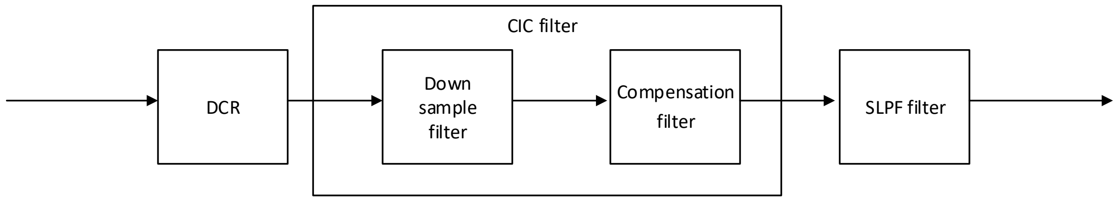


Figure 41 Audio Front Data Path

DCR supports four modes as below, and DCR should be set bypass mode for PDM input.

Table 170 DCR mode configuration

DCR mode	CIC_DCR_BYPASS	CIC_DC_HOST_EN	DC_SETTLE_EN	description
Bypass mode	1	x	x	DCR bypass
CPU set mode	0	1	x	DC value set by register CIC_DC_HOST_VALUE
Hold mode	0	0	1	DC calculation time set by register DC_SETTLE_EXP. After settle time, DC is hold.
Always on	0	0	0	DC always calculates

5.16.7 Typical Characteristics

Amplitude-frequency characteristics of CIC filters after compensation is shown in the figure below.

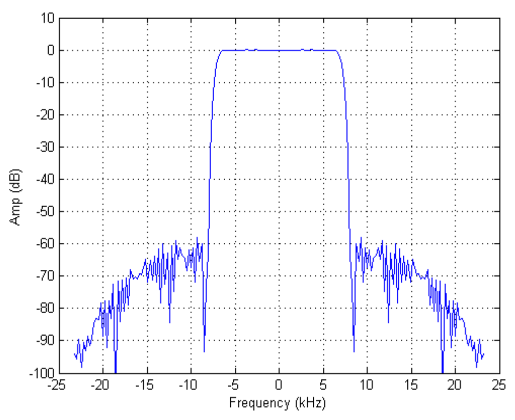


Figure 42 CIC frequency response

MICBIAS(P4) output voltage vs. VDDR as figure below.

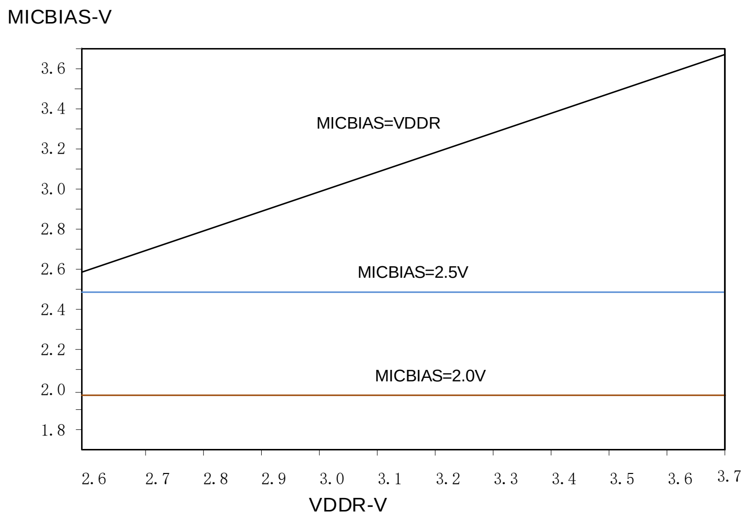


Figure 43 MICBIAS output voltage vs. VDDR

5.16.8 Interrupts

The audio module has four interrupt sources, three from the I2S module and one from the CIC. Interrupt source table is shown below.

Table 171 CIC interrupt

Interrupt Number	Interrupt Name	Description
0	I2S_WCLK_TIMEOUT	I2S WCLK TIME OUT
1	I2S_WCLK_ERR	I2S WCLK ERROR
2	I2S_BUS_ERR	I2S INPUT BUFFER OVERFLOW
3	CIC_FIFO_OVER_FLOW	CIC FIFO IS UNDER OVERFLOW

I2S_WCLK_TIMEOUT

It’s set when the sample stamp generator does not detect a positive WCLK edge for 65535 clock periods. This signalizes that the internal or external BCLK and WCLK generator source has been disabled. The bit is sticky and may only be cleared by software (by writing '1' to I2S_WCLK_TIMEOUT_CLR).

I2S_WCLK_ERR

It’s set when:

- An unexpected WCLK edge occurs during the data delay period of a phase. Note unexpected WCLK edges during the word and idle periods of the phase are not detected.
- In I2S/LJF mode, when two WCLK edges are less than 4 BCLK cycles apart.
- In DSP-A/DSP-B mode, when a WCLK pulse occurs before the last channel.

This error requires a complete restart since word synchronization has been lost. The bit is sticky and may only be cleared by software (by writing '1' to I2S_WCLK_ERROR_CLR).

I2S_BUS_ERR

It’s set when a DMA operation is not completed in time (audio input buffer overflow). The bit is sticky and may only be cleared by software (by writing '1' to BUS_ERROR_CLR). Note that DMA initiated transactions to illegal address will not trigger an interrupt. The response to such transactions is undefined.

CIC_FIFO_OVER_FLOW

It's set when a DMA operation is not completed in time (CIC output FIFO overflow). The bit is sticky and may only be cleared by software (by writing '1' to CIC_FIFO_OVER_FLOW_CLR). Note that DMA initiated transactions to illegal address will not trigger an interrupt. The response to such transactions is undefined.

5.16.9 Software Procedure

5.16.9.1 PDM Mode

The PDM work mode procedure.

1. Configure GPIO with PDM_DATA (share SDATA) and PDM_CLK (share I2S_MCLK) pins.
2. Configure CIC module soft reset=0, and hold
3. Configure CIC as PDM mode,
4. Configure DMA interface
5. Enable CIC module clock
6. Release CIC soft reset, and audio front data processor works as PDM mode

5.16.9.2 ADC Mode

The ADC work mode procedure.

1. Configure CIC module soft reset=0, and hold
2. Initial analog for ADC mode:
    a) Configure P4 to high-z, used to BIAS for external MIC
    b) Configure ADC_EN and MIC_BIAS reference voltage
    c) Configure P0/P1 to high-z. used as analog MIC data input port
    d) Configure PGA gain
3. Configure CIC as ADC mode
4. Configure DMA interface
5. Enable CIC module clock
6. Release CIC soft reset, and audio front data processor works as ADC mode

5.16.9.3 I2S Mode

The I2S work mode procedure.

1. Set up and configure required SDATA and clock pins (refer to GPIO registers).
2. Configure the master/slave mode.

3. Configure BCLK, WCLK (sample frequency) and MCLK audio clocks.
4. Configure the serial audio interface format (DATA_FORMAT) and length (DATA_LENGTH).
5. Configure the DMA interface.
6. Configure the channel (CH_MASK) to start receive.

5.16.10 Audio Register

Table 172 CIC Instance

Base Address	Peripheral	Instance	Description
0x4001B000	PMU	PMU	Power Management Unit
0x40021000	I2S	I2S	Inter-IC Sound
0x40022000	CIC	CIC	Cascaded Integrator-Comb Filter

5.16.10.1 Audio Analog Register

Table 173 Audio Analog Register Map

Address Offset	Name	Description
0xBC	AUDIO_ADC_CFG	Audio ADC Configure Register

Table 174 AUDIO_ADC_CFG (PMU_BASE_ADDR +0xBC)

Bit	Field Name	Reset	RW	Description
9:7	RESERVED	0x0	RW	Reserved for test
6	AUDIO_PGA_SINGLE_EN	0	RW	Audio PGA single end select: 0b: Single end 1b:difference end
5:3	RESERVED	0x0	RW	Reserved for test
2:0	MIC_BIAS_REF_SEL	0x0	RW	ADC BIAS output Voltage select: 000b: disable, P4 output high-z 001b: MIC_BIAS output Voltage 2V 010b: MIC_BIAS output Voltage 2.5V 011b: MIC_BIAS output Voltage VDDR 1xxb:reserved

5.16.10.2 I2S Register

Table 175 I2S Register Map

Address Offset	Name	Description
0x00	I2S_FIFO	I2S FIFO output
0x04	I2S_INT_FLAG	I2S interrupt flag

0x08	I2S_INT_SET	I2S interrupt set
0x0c	I2S_INT_CLR	I2S interrupt clear
0x10	I2S_INT_EN	I2S interrupt enable
0x14	I2S_CONFIG	I2S configuration

Table 176 I2S_FIFO (I2S_BASE_ADDR +0x0)

Bit	Field Name	Reset	RW	Description
31:0	I2S_FIFO	0x0	R	DMA/CPU read i2s receive FIFO data via this register

Table 177 I2S_INT_FLAG (I2S_BASE_ADDR +0x4)

Bit	Field Name	Reset	RW	Description
2	I2S_WCLK_TIMEOUT	0x0	R	See Interrupt
1	I2S_WCLK_ERR	0x0	R	See Interrupt
0	I2S_BUS_ERR	0x0	R	See Interrupt

Table 178 I2S_INT_SET (I2S_BASE_ADDR +0x8)

Bit	Field Name	Reset	RW	Description
2	I2S_WCLK_TIMEOUT_SET	0x0	W	I2S_WCLK_TIMEOUT interrupt set 0 : invalid 1 : set
1	I2S_WCLK_ERR_SET	0x0	W	I2S_WCLK_ERR interrupt set 0 : invalid 1 : set
0	I2S_BUS_ERR_SET	0x0	W	I2S_BUS_ERR interrupt set 0 : invalid 1 : set

Table 179 I2S_INT_CLR (I2S_BASE_ADDR +0xC)

Bit	Field Name	Reset	RW	Description
2	I2S_WCLK_TIMEOUT_CLR	0x0	W	I2S_WCLK_TIMEOUT interrupt clear 0 : invalid 1 : clear
1	I2S_WCLK_ERR_CLR	0x0	W	I2S_WCLK_ERR interrupt clear 0 : invalid 1 : clear
0	I2S_BUS_ERR_CLR	0x0	W	I2S_BUS_ERR interrupt clear 0 : invalid 1 : clear

Table 180 I2S_INT_EN (I2S_BASE_ADDR +0x10)

Bit	Field Name	Reset	RW	Description

2	I2S_WCLK_TIMEOUT_EN	0x0	RW	I2S_WCLK_TIMEOUT interrupt enable 0 : disable 1 : enable
1	I2S_WCLK_ERR_EN	0x0	RW	I2S_WCLK_ERR interrupt enable 0 : disable 1 : enable
0	I2S_BUS_ERR_EN	0x0	RW	I2S_BUS_ERR interrupt enable 0 : disable 1 : enable

Table 181 I2S_CONFIG (I2S_BASE_ADDR +0x14)

Bit	Field Name	Reset	RW	Description
18	PDM_SAMPLE_EDGE	0	RW	PDM sample data edge selection 0 : negative edge 1 : positive edge
17	PDM_RATE	0	RW	PDM data rate selection 0 : 1M data rate 1 : 2M data rate
16	AUDIO_MODE	0	RW	Audio mode selection 0 : LJF/I2S/PCM mode 1 : PDM/ADC mode
15	PDM_EN	0	RW	PDM enable 0 : disable 1 : enable
14	MCLK_EN	0	RW	Master clock generate enable 0 : disable 1 : enable
13	XO_32M_N16M	0	RW	External crystal frequency is 16MHz or 32MHz 0 : 16M 1 : 32M
12:11	FS_NUM	0x3	RW	Sample frequency selection 0 : 8k, 1 : 16k 2 : 32KHz 3 : reserved
10	MASTER_SLAVE	0	RW	Master/slave selection 0 : slave 1 : master
9:8	MCLK_DIV	0	RW	Master clock selection 0 : 2MHz 1 : 4MHz 2 : 8MHz

				3 : 16MHz
7:6	CH_MASK	0	RW	Channel selection 0 : disable 1 : mono (left) 2 : mono (right) 3 : stereo
5:4	DATA_LENGTH	0x3	RW	data length selection 0 : 16bit 1 : 20bit 2 : 24bit 3 : 32bit (only I2S/LJF format support)
3:2	DATA_FORMAT	0x1	RW	Data format selection 0 : LJF 1 : I2S 2 : DSP-a 3 : DSP-b
1	WCLK_INV	0	RW	Inverts WCLK source (pad or internal) when set. 0 : not inverted 1 : inverted
0	SMPL_EDGE	0x1	RW	BCLK edge selection to sample DATA and WCLK 0 : BCLK negative edge 1 : BCLK positive edge

5.16.10.3 CIC Register

Table 182 CIC Register Map

Address Offset	Name	Description
0x00	CIC_FIFO	CIC FIFO output Register
0x04	CIC_DOWNSP	CIC filter down-sample rate Register
0x08	CIC_COEFF 0	CIC compensating filter coefficient Register
0x0C	CIC_COEFF 1	CIC compensating filter coefficient Register
0x10	CIC_COEFF 2	CIC compensating filter coefficient Register
0x14	CIC_COEFF 3	CIC compensating filter coefficient Register
0x18	CIC_CONFIG_1	CIC Configuration Register
0x1C	CIC_STA_REG	CIC Status Register
0x20	CIC_CTRL_REG	CIC Control Register
0x24	CIC_CONFIG_2	CIC Configuration Register

Table 183 CIC_FIFO (CIC_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
31:0	CIC_FIFO	0x0	R	CIC FIFO read register

Table 184 CIC_DOWNSP (SPIx_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
8:0	CIC_DOWNSP	0x1F4	RW	CIC filter down sample rate 500: in SRADC mode 125: in PDM mode

Table 185 CIC_COEFF 0 (CIC_BASE_ADDR +0x08)

Bit	Field Name	Reset	RW	Description
8:0	CIC_COEFF 0	0x1FE	RW	CIC compensating filter coefficient 0, singed number

Table 186 CIC_COEFF 1 (CIC_BASE_ADDR +0x0C)

Bit	Field Name	Reset	RW	Description
8:0	CIC_COEFF 1	0x00D	RW	CIC compensation filter coefficient 1, singed number

Table 187 CIC_COEFF 2 (CIC_BASE_ADDR +0x10)

Bit	Field Name	Reset	RW	Description
8:0	CIC_COEFF 2	0x1C7	RW	CIC compensation filter coefficient 2, singed number

Table 188 CIC_COEFF 3 (CIC_BASE_ADDR +0x14)

Bit	Field Name	Reset	RW	Description
8:0	CIC_COEFF 3	0xDC	RW	CIC compensation filter coefficient 3, unsigned number

Table 189 CIC_CONFIG_1 (CIC_BASE_ADDR +0x18)

Bit	Field Name	Reset	RW	Description
20	ADC_POS_SEL	0	RW	SRADC output data sample clock edge selection: 0b: falling edge 1b: raise edge
19	DC_SETTLE_EN	0	RW	DC(direct current) calculate mode 0b: DC calculation always on 1b: DC calculation time refer to register DC_SETTLE_EXP
18:16	DC_SETTLE_EXP	0x0	RW	DC calculation time =2^(2+dc_settle_exp) 000b: 4ms 001b: 8ms 010b: 16ns 011b: 32ms 100b: 64ms 101b: 128ms 110b: 256ms 111b: 512ms
15	RESERVED	0	RW	Reserved bit

14	CIC_SLPF_BYPASS_EN	0	RW	Strong low pass filter bypass enable 0b: not bypass 1b: bypass
13:12	CIC_MODE_SEL	0x2	RW	CIC FIFO data source selection 00b: reserved 01b: from PDM bypass CIC 10b: from PDM not bypass CIC 11b: from SRADC not bypass CIC
11	CIC_ADC_ORG_SEL	0	RW	SRADC original code selection 0b: complement code 1b: original code
10	RESERVED	0	RW	Reserved bit
9	CIC_DCR_BYPASS	0	RW	CIC DCR(DC removal) bypass control 0b: not bypass 1b: bypass
8:4	CIC_A_RSHIFT	0x0	RW	CIC down sample filter DGC(digital gain control) coefficient coefficient = 2^(CIC_A_RSHIFT)
3:0	CIC_DCR_N	0xC	RW	DCR calculation coefficient

Table 190 CIC_STA_REG (CIC_BASE_ADDR +0x1C)

Bit	Field Name	Reset	RW	Description
30	CIC_FIFO_SOURCE_FLAG	0	R	CIC FIFO data source selection 0b: from I2S 1b: from PDM/ADC
29	CIC_DC_CALCULATION_FLG	0	R	DCR calculation working flag 0b: working done 1b: in working
28	CIC_FIFO_OVERFLOW_FLG	0	R	FIFO overflow interrupt status
27:0	CIC_DC_VAULE	0x0	R	DCR DC value, singed

Table 191 CIC_CTRL_REG (CIC_BASE_ADDR +0x20)

Bit	Field Name	Reset	RW	Description
2	CIC_DCR_START	0	W	CIC dc calculation trigger 0b: not trigger 1b: trigger
1	CIC_FIFO_OVER_FLOW_CLR	0	W	FIFO overflow interrupt clear trigger 0b: invalid 1b: clear
0	RESERVED	0	W	Reserved bit

Table 192 CIC_CONFIG_2 (CIC_BASE_ADDR +0x24)

Bit	Field Name	Reset	RW	Description

14:4	CIC_DC_HOST_VALUE	0x0	RW	DC host value, signed
3	CIC_DC_HOST_EN	0	RW	DC value host enable 0b: disable, DCR uses hardware automatic calculated value 1b: enable, DC value used CIC_DC_HOST_VALUE
2:1	CIC_RESERVED	0x3	RW	Reserved bit
0	CIC_INT_FIFO_OVER_FLOW	0	RW	CIC FIFO is over flow interrupt enable flag 0b: disable 1b: enable

5.17 Timer

The timer includes timer0 and general purpose timer1~4 as illustrated below.

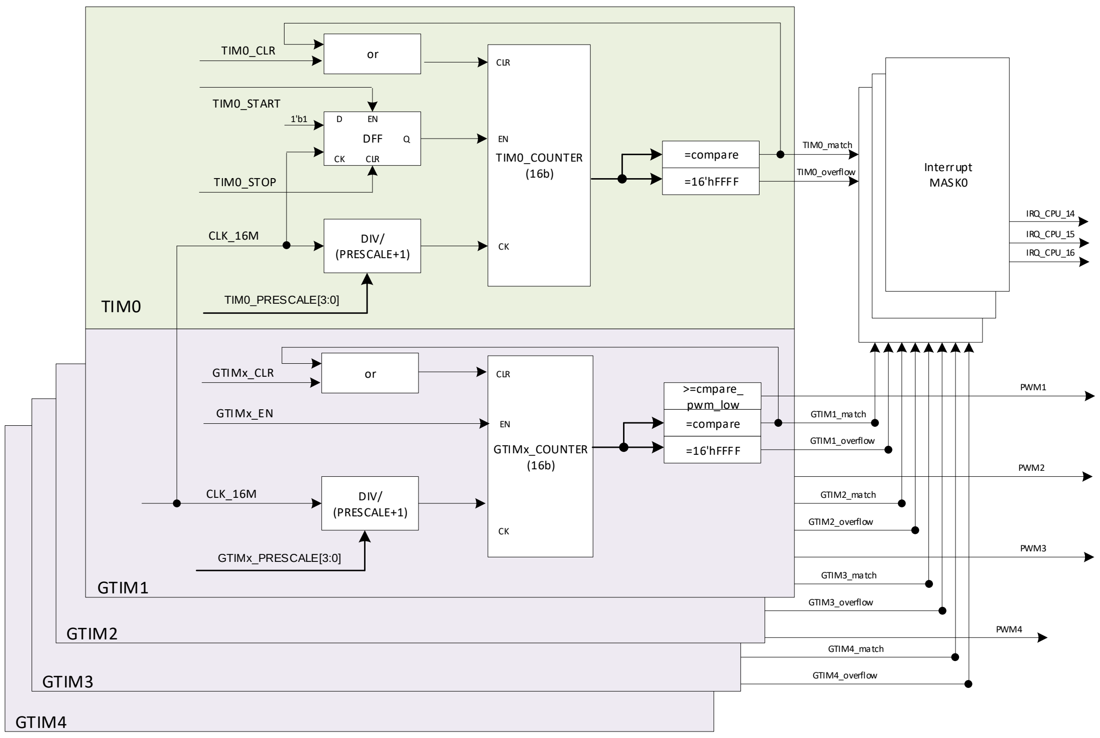


Figure 44 Timer Block Diagram

5.17.1 Feature

Timer 0

1. 16-bit counter with 4-bit pre-scale
2. The tick can be 1MHz~16MHz
3. Counter mode
4. Interrupt mode. Generate matching interrupt or overflow interrupt

General Purpose Timer 1 ~ Timer 4

1. 16-bit counter with 4-bit pre-scale
2. The tick can be 1MHz~16MHz
3. Counter mode
4. Interrupt mode. Generate matching interrupt or overflow interrupt
5. PWM mode. Frequency range 16Hz~8MHz. Duty cycle 0%~100%. Waveform smooth switch for new parameters

5.17.2 Function Description

The 16-bit general purpose timer is incremented by 1 every 16MHz/8MHz/4MHz/2MHz/1MHz cycle, which depends on the register (GPT_GTIM_PRESCALE_CTRL). Timer0 supports counter and interrupt modes. Timer 1 ~ timer 4 supports additional PWM mode. Timer 1 and timer 2 supports IR mode.

The timer0~4 counter will count from 0 to 0xffff to generate an overflow interrupt when the register COMPARE is zero. The timer0~4 counter will count from 0 to COMPARE to generate the matching interrupt when the register COMPARE is not zero.

5.17.3 PWM Function

The PWM signal can be changed by the registers (GTIMx_PWM_POL, GTIMx_COMPARE and GTIMx_PWM_LOW) when counter reaches the value of GTIM1_COMPARE. The registers (GTIMx_PWM_POL, GTIMx_COMPARE and GTIMx_LOW) are automatically loaded when the PWM counter completes the previous counting, so smooth waveform transition is guaranteed when the parameters change.

High level duty cycle is (1-(low/compare+1))*100%.

Duty cycle is 100% when GTIM1_PWM_LOW = 0 and GTIM1_COMPARE > 0, or when GTIM1_PWM_LOW = (GTIM1_COMPARE+1) and GTIM1_COMPARE > 0.

Duty cycle is 0% when GTIM1_PWM_LOW >= (GTIM1_COMPARE+1).

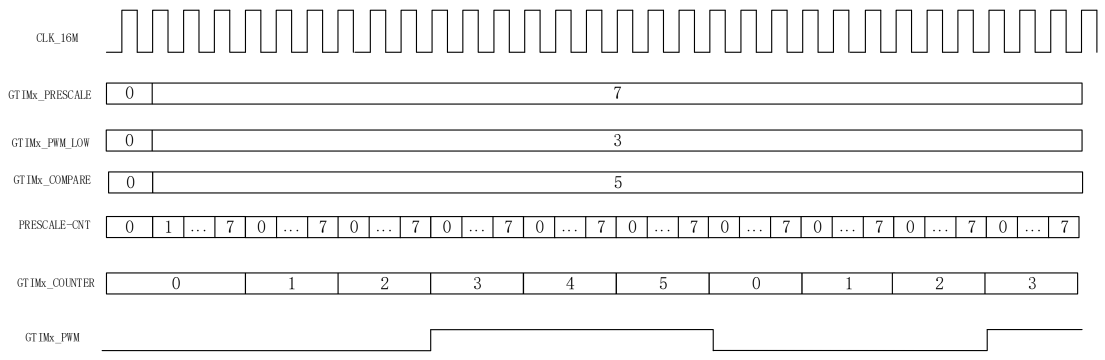


Figure 45 PWM Generation

5.17.4 IR Function

IR is generated by timer1 PWM and timer2 PWM. Timer 1 PWM is the IR carrier signal, and timer 2 PWM is the IR code. Timer 1 parameters is always the same. Timer 2 parameters needs to be dynamically changed to generate the IR code.

Hardware loads the configured new parameter registers (GTIM2_COMPARE/GTIM2_PWM_LOW) when the current counter matches. In the matching interrupt handler, software sets the next new parameters. With such hardware and software co-work mechanism, arbitrary IR code waveform can be generated continuously.

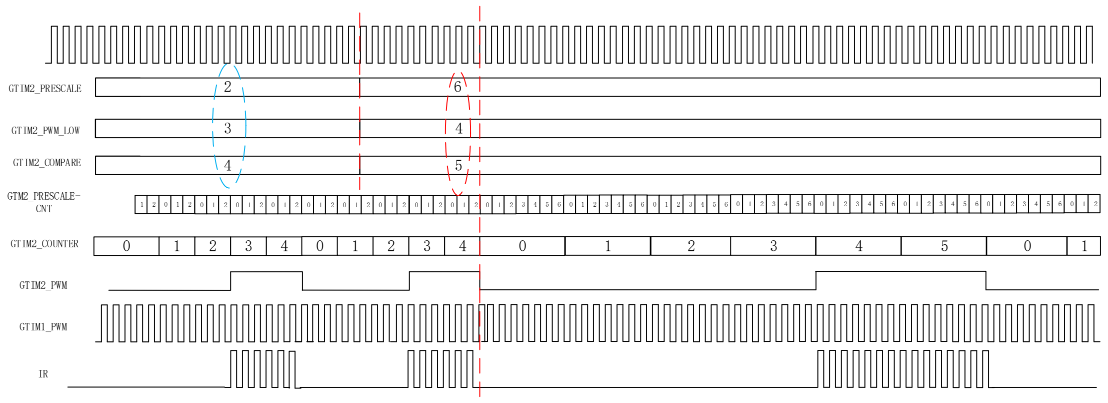


Figure 46 IR Generation

5.17.5 Software Procedure

Timer0 initial flow.

1. Set GPT_INT_MASK and GPT_INT_EN0/EN1/EN2
2. Set timer0 pre-scale by GPT_TIM0_CTR
3. Set GPT_TIM0_COMPARE
4. Set TIM0_START by TIM0_TRIG_CFG

Timer1 ~ timer4 initial flow.

1. Set GPT_INT_MASK and GPT_INT_EN0/EN1/EN2
2. Set pre-scale by GPT_GTIM_PRESCALE_CTRL
3. Set GPT_GTIM1_PWM_COMPARE
4. Set Timer1 enable by GPT_GTIM_CTRL[GTIM1_ENABLE]

PWM initialization flow.

1. Clear timer counter, write GPT_TRIG_CFG
2. Set 16-bit timer compare value. Write the register (GPT_GTIM1_PWM_COMPARE[GTIM1_PWM_LOW], GPT_GTIM1_PWM_COMPARE[GTIM1_COMPARE])

3. Set timer pre-scale register (GPT_TIM_PRESCALE_CTRL)
4. Select the target GPIO
5. Enable timer and enable PWM through the register (GPT_GTIM_CTRL)

PWM new parameter set flow.

1. Set new compare and duty register (GPT_GTIM1_PWM_COMPARE)
2. Set new pre-scale register (GPT_TIM_PRESCALE_CTRL). Hardware guarantees PWM new waveform smooth transition

IR flow.

1. Clear timer1/2 counter, write GPT_TRIG_CFG
2. Set 16-bit timer1 compare value, write the register (GTIM1_PWM_LOW, GTIM1_COMPARE)
3. Set 16-bit timer2 compare value, write the register (GTIM2_PWM_LOW, GTIM2_COMPARE)
4. Set timer1/2 pre-scale value by GPT_TIM_PRESCALE_CTRL
5. Select target GPIO
6. Enable timer1 and timer2 and enable timer1 PWM and timer2 PWM by GPT_GTIM_CTRL
7. Set timer 2 new parameter
8. Dynamically set timer 2 next new parameter in the interrupt handler

5.17.6 Timer0~4 Interrupt

Timer0~4 interrupt table is shown below.

Table 193 Timer0~4 Interrupt

Interrupt Number	Interrupt name	Description
9	TIM4_OVERFLOW_INT	Timer4 reaches 16'hffff, then generate overflow interrupt
8	TIM4_MATCHING_INT	Timer4 reaches the value of register (GTIM4_COMPARE), then generate matching interrupt
7	TIM3_OVERFLOW_INT	Timer3 reaches 16'hffff, then generate overflow interrupt
6	TIM3_MATCHING_INT	Timer3 reaches the value of register (GTIM3_COMPARE), then generate matching interrupt
5	TIM2_OVERFLOW_INT	Timer2 reaches 16'hffff, then generate overflow interrupt
4	TIM2_MATCHING_INT	Timer2 reaches the value of register (GTIM2_COMPARE), then generate matching interrupt
3	TIM1_OVERFLOW_INT	Timer1 reaches 16'hffff, then generate overflow interrupt
2	TIM1_MATCHING_INT	Timer1 reaches the value of register (GTIM1_COMPARE), then generate matching interrupt
1	TIM0_OVERFLOW_INT	Timer0 reaches 16'hffff, then generate overflow interrupt
0	TIM0_MATCHING_INT	Timer0 reaches the value of register (TIM0_COMPARE), then generate matching interrupt

5.17.7 Timer0~4 Registers

Table 194 Timer0~4 Instance

Base Address	Peripheral	Instance	Description
0x40015000	GPT	GPT	General Purpose Timer

Table 195 Timers Register Map

Address Offset	Name	Description
0x00	GPT_INT_STATUS	GPT Interrupt Status
0x04	GPT_INT_MASK	GPT Interrupt Enable
0x08	GPT_INT_CLEAR	GPT Interrupt Flag Clear Register
0x0C	GPT_TIM0_CTR	Timer0 Pre-scaler Register
0x10	GPT_TIM0_TRIG_CFG	Timer0 Trigger Register
0x14	GPT_TIM0_COMPARE	Timer0 Compare Register
0x18	GPT_TIM0_CNT	Timer0 Counter
0x1C	GPT_GTIM_CTRL	Timer1~4 Enable Register
0x20	GPT_TRIG_CFG	Timer1~4 Counter Clear Trigger Register
0x24	GPT_GTIM_PRESCALE_CTRL	Timer1~4 Prescaler Register
0x28	GPT_GTIM_CNT_READ_SEL	Timer1~4 Counter Read Selection
0x2C	GPT_GTIM_CNT	Timer1~4 Counter
0x30	GPT_GTIM1_PWM_COMPARE	Timer1 Compare/PWM Register
0x34	GPT_GTIM2_PWM_COMPARE	Timer2 Compare/PWM Register
0x38	GPT_GTIM3_PWM_COMPARE	Timer3 Compare/PWM Register
0x3C	GPT_GTIM4_PWM_COMPARE	Timer4 Compare/PWM Register
0x40	GPT_INT_EN0	Timer0~4 Interrupt Selection for IRQ_CPU_14
0x44	GPT_INT_EN1	Timer0~4 Interrupt Selection for IRQ_CPU_15
0x48	GPT_INT_EN2	Timer0~4 Interrupt Selection for IRQ_CPU_16

Table 196 GPT_INT_STATUS (GPT_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
9	TIM4_OVERFLOW_INT	0	R	Timer4 overflow interrupt
8	TIM4_MATCHING_INT	0	R	Timer4 matching interrupt
7	TIM3_OVERFLOW_INT	0	R	Timer3 overflow interrupt
6	TIM3_MATCHING_INT	0	R	Timer3 matching interrupt
5	TIM2_OVERFLOW_INT	0	R	Timer2 overflow interrupt
4	TIM2_MATCHING_INT	0	R	Timer2 matching interrupt

3	TIM1_OVERFLOW_INT	0	R	Timer1 overflow interrupt
2	TIM1_MATCHING_INT	0	R	Timer1 matching interrupt
1	TIM0_OVERFLOW_INT	0	R	Timer0 overflow interrupt
0	TIM0_MATCHING_INT	0	R	Timer0 matching interrupt

Table 197 GPT_INT_EN (GPTT_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
9	TIM4_OVERFLOW_INTEN	0	RW	Timer4 overflow interrupt enable 0: disable 1: enable
8	TIM4_MATCHING_INTEN	0	RW	Timer4 matching interrupt enable 0: disable 1: enable
7	TIM3_OVERFLOW_INTEN	0	RW	Timer3 overflow interrupt enable 0: disable 1: enable
6	TIM3_MATCHING_INTEN	0	RW	Timer3 matching interrupt enable 0: disable 1: enable
5	TIM2_OVERFLOW_INTEN	0	RW	Timer2 overflow interrupt enable 0: disable 1: enable
4	TIM2_MATCHING_INTEN	0	RW	Timer2 matching interrupt enable 0: disable 1: enable
3	TIM1_OVERFLOW_INTEN	0	RW	Timer1 overflow interrupt enable 0: disable 1: enable
2	TIM1_MATCHING_INTEN	0	RW	Timer1 matching interrupt enable 0: disable 1: enable
1	TIM0_OVERFLOW_INTEN	0	RW	Timer0 overflow interrupt enable 0: disable 1: enable
0	TIM0_MATCHING_INTEN	0	RW	Timer0 matching interrupt enable 0: disable 1: enable

Table 198 GPT_INT_CLEAR (GPT_BASE_ADDR+0x08)

Bit	Field Name	Reset	RW	Description
9	TIM4_OVERFLOW_INTCLR	0	W	Timer4 overflow interrupt clear 0: invalid

				1: clear
8	TIM4_MATCHING_INTCLR	0	W	Timer4 matching interrupt clear 0: invalid 1: clear
7	TIM3_OVERFLOW_INTCLR	0	W	Timer3 overflow interrupt clear 0: invalid 1: clear
6	TIM3_MATCHING_INTCLR	0	W	Timer3 matching interrupt clear 0: invalid 1: clear
5	TIM2_OVERFLOW_INTCLR	0	W	Timer2 overflow interrupt clear 0: invalid 1: clear
4	TIM2_MATCHING_INTCLR	0	W	TIMER2 matching interrupt clear 0: invalid 1: clear
3	TIM1_OVERFLOW_INTCLR	0	W	Timer1 overflow interrupt clear 0: invalid 1: clear
2	TIM1_MATCHING_INTCLR	0	W	Timer1 matching interrupt clear 0: invalid 1: clear
1	TIM0_OVERFLOW_INTCLR	0	W	Timer0 overflow interrupt clear 0: invalid 1: clear
0	TIM0_MATCHING_INTCLR	0	W	Timer0 matching interrupt clear 0: invalid 1: clear

Table 199 GPT_TIM0_CTR (GPT_BASE_ADDR+0x0C)

Bit	Field Name	Reset	RW	Description
3:0	TIM0_PRESCALE	0x0	RW	Timer0 pre-scale register Pre-scale for timer0 counter frequency frequency(timer0) = 16MHz/(TIM0_PRESCALE +1)

Table 200 GPT_TIM0_TRIG_CFG (GPT_BASE_ADDR+0x10)

Bit	Field Name	Reset	RW	Description
2	TIM0_CNT_CLR	0	W	Timer0 counter clear
1	TIM0_STOP	0	W	Timer0 counter stop 0:invalid 1:stop

0	TIM0_START	0	W	Timer0 counter start 0:invalid 1:start

Table 201 TIM0_COMPARE (GPT_BASE_ADDR+0x14)

Bit	Field Name	Reset	RW	Description
15:0	TIM0_COMPARE	0x0	RW	Timer0 compare register Used to generate matching interrupt when timer0 counter reaches register (TIM0_COMPARE) Note: This register sets to zero when generating overflow interrupt timer0 period = (TIM0_PRESCALE +1)*(TIM0_COMPARE+1)/16MHz

Table 202 GPT_TIM0_CNT (GPT_BASE_ADDR+0x18)

Bit	Field Name	Reset	RW	Description
15:0	GPT_TIM0_CNT	0x0	R	Timer0 counter

Table 203 GPT_GTIM_CTRL (GPT_BASE_ADDR+0x1C)

Bit	Field Name	Reset	RW	Description
13	GTIM4_PWM_POL	0	RW	Timer4 PWM polarity 0: PWM is low during idle state and low time state 1: PWM is high during idle state and low time state
12	GTIM3_PWM_POL	0	RW	Timer3 PWM polarity 0: PWM is low during idle state and low time state 1: PWM is high during idle state and low time state
11	GTIM2_PWM_POL	0	RW	Timer2 PWM polarity 0: PWM is low during idle state and low time state 1: PWM is high during idle state and low time state
10	GTIM1_PWM_POL	0	RW	Timer1 PWM polarity 0: PWM is low during idle state and low time state 1: PWM is high during idle state and low time state
9	PWM4_ENABLE	0	RW	Timer4 PWM output enable

				0: disable 1: enable Note: Because timer4 is used to generate PWM output, GTIM4_ENABLE must be 1 too.
8	PWM3_ENABLE	0	RW	Timer3 PWM output enable 0: disable 1: enable Note: Because timer3 is used to generate PWM output, GTIM3_ENABLE must be 1 too.
7	PWM2_ENABLE	0	RW	Timer2 PWM output enable 0: disable 1: enable Note: Because Time2 is used to generate PWM output, GTIM2_ENABLE must be 1 too.
6	PWM1_ENABLE	0	RW	Timer1 PWM output enable 0: disable 1: enable Note: Because Timer1 is used to generate PWM output, GTIM1_ENABLE must be 1 too.
5	IR_POL	0	RW	IR polarity 0: IR is high during idle state and low time state 1: IR is low during idle state and low time state
4	IR_ENABLE	0	RW	IR function enable 0: disable 1: enable Note: Because Timer1 and Timer2 are used to generate IR output, GTIM1_ENABLE and GTIM2_ENABLE must be 1 too.
3	GTIM4_ENABLE	0	RW	General purpose timer4 enable 0: disable 1: enable
2	GTIM3_ENABLE	0	RW	General purpose timer3 enable 0: disable 1: enable
1	GTIM2_ENABLE	0	RW	General purpose timer2 enable 0: disable 1: enable
0	GTIM1_ENABEL	0	RW	General purpose timer1 enable 0: disable 1: enable

Table 204 GPT_TRIG_CFG (GPT_BASE_ADDR+0x20)

Bit	Field Name	Reset	RW	Description
3	GTIM4_CNT_CLR	0	W	General purpose timer4 counter clear 0: invalid 1: clear
2	GTIM3_CNT_CLR	0	W	General purpose timer3 counter clear 0: invalid 1: clear
1	GTIM2_CNT_CLR	0	W	General purpose timer2 counter clear 0: invalid 1: clear
0	GTIM1_CNT_CLR	0	W	General purpose timer1 counter clear 0: invalid 1: clear

Table 205 GPT_GTIM_PRESCALE_CTRL (GPT_BASE_ADDR+0x24)

Bit	Field Name	Reset	RW	Description
15:12	GTIM4_PRESCALE	0x2	RW	Pre-scaler for general purpose timer4 frequency: Frequency (timer4) = 16MHz/(GTIM4_PRESCALE +1)
11:8	GTIM3_PRESCALE	0x2	RW	Pre-scaler for general purpose timer3 frequency: Frequency (timer3) = 16MHz/(GTIM3_PRESCALE +1)
7:4	GTIM2_PRESCALE	0x2	RW	Pre-scaler for general purpose timer2 frequency: Frequency (timer2) = 16MHz/(GTIM2_PRESCALE +1)
3:0	GTIM1_PRESCALE	0x2	RW	Pre-scaler for general purpose timer1 frequency: Frequency (timer1) = 16MHz/(GTIM1_PRESCALE +1)

Table 206 GPT_GTIM_CNT_READ_SEL (GPT_BASE_ADDR+0x28)

Bit	Field Name	Reset	RW	Description
2:0	GPT_GTIM_CNT_READ_SEL	0x0	RW	General purpose timer1~timer4 counter selection 1: timer1 counter 2: timer2 counter 3: timer3 counter 4: timer4 counter

				5~7: timer1 counter

Table 207 GPT_GTIM_CNT (GPT_BASE_ADDR+0x2C)

Bit	Field Name	Reset	RW	Description
15:0	GPT_GTIM_CNT	0x0	R	Counter value depends on register (GPT_GTIM_CNT_READ_SEL) selection

Table 208 GPT_GTIM1_PWM_COMPARE (GPT_BASE_ADDR+0x30)

Bit	Field Name	Reset	RW	Description
31:16	GTIM1_PWM_LOW	0xF	RW	Define the cycle Timer1 PWM enters high time state. If register (GTIM1_PWM_LOW) -1 >= GTIM1_COMPARE, PWM out is always 0 If register (GTIM1_PWM_LOW) = 0, PWM out is always 1
15:0	GTIM1_COMPARE	0xFF	RW	Define the cycle Timer1 PWM enters low time state. It can be used to generate matching interrupt when general purpose timer1 reaches register (GTIM1_COMPARE) Note: If generating overflow interrupt, register (GTIM1_COMPARE) must be 0

Table 209 GPT_GTIM2_PWM_COMPARE (GPT_BASE_ADDR+0x34)

Bit	Field Name	Reset	RW	Description
31:16	GTIM2_PWM_LOW	0xF	RW	Define the cycle Timer2 PWM enters high time state. If register (GTIM2_PWM_LOW) -1 >= GTIM2_COMPARE, PWM out is always 0 If register (GTIM2_PWM_LOW) = 0, PWM out is always 1
15:0	GTIM2_COMPARE	0xFF	RW	Define the cycle Timer2 PWM enters low time state. It can be used to generate matching interrupt when general purpose timer1 reaches register (GTIM2_COMPARE) Note: If generating overflow interrupt, register (GTIM2_COMPARE) must be 0

Table 210 GPT_GTIM3_PWM_COMPARE (GPT_BASE_ADDR+0x38)

Bit	Field Name	Reset	RW	Description
31:16	GTIM3_PWM_LOW	0xF	RW	Define the cycle Timer3 PWM enters high time

				state. If register (GTIM3_PWM_LOW) -1 >= GTIM3_COMPARE, PWM out is always 0 If register (GTIM3_PWM_LOW) = 0, PWM out is always 1
15:0	GTIM3_COMPARE	0xFF	RW	Define the cycle Timer3 PWM enters low time state. It can be used to generate matching interrupt when general purpose timer1 reaches register (GTIM3_COMPARE) Note: If generating overflow interrupt, register (GTIM3_COMPARE) must be 0

Table 211 GPT_GTIM4_PWM_COMPARE (GPT_BASE_ADDR+0x3C)

Bit	Field Name	Reset	RW	Description
31:16	GTIM4_PWM_LOW	0xF	RW	Define the cycle Timer4 PWM enters high time state. If register (GTIM4_PWM_LOW) -1 >= GTIM4_COMPARE, PWM out is always 0 If register (GTIM4_PWM_LOW) = 0, PWM out is always 1
15:0	GTIM4_COMPARE	0xFF	RW	Define the cycle Timer4 PWM enters low time state. It can be used to generate matching interrupt when general purpose timer1 reaches register (GTIM4_COMPARE) Note: If generating overflow interrupt, register (GTIM4_COMPARE) must be 0

Table 212 GPT_INT_EN0 (GPT_BASE_ADDR+0x40)

Bit	Field Name	Reset	RW	Description
9	TIM4_OVERFLOW_INTEN0	0	RW	Timer4 overflow interrupt enable 0: disable 1: enable
8	TIM4_MATCHING_INTEN0	0	RW	Timer4 matching interrupt enable 0: disable 1: enable
7	TIM3_OVERFLOW_INTEN0	0	RW	Timer3 overflow interrupt enable 0: disable 1: enable
6	TIM3_MATCHING_INTEN0	0	RW	Timer3 matching interrupt enable 0: disable

				1: enable
5	TIM2_OVERFLOW_INTEN0	0	RW	Timer2 overflow interrupt enable 0: disable 1: enable
4	TIM2_MATCHING_INTEN0	0	RW	Timer2 matching interrupt enable 0: disable 1: enable
3	TIM1_OVERFLOW_INTEN0	0	RW	Timer1 overflow interrupt enable 0: disable 1: enable
2	TIM1_MATCHING_INTEN0	0	RW	Timer1 matching interrupt enable 0: disable 1: enable
1	TIM0_OVERFLOW_INTEN0	0	RW	Timer0 overflow interrupt enable 0: disable 1: enable
0	TIM0_MATCHING_INTEN0	0	RW	Timer0 matching interrupt enable 0: disable 1: enable

Table 213 GPT_INT_EN1 (GPT_BASE_ADDR+0x44)

Bit	Field Name	Reset	RW	Description
9	TIM4_OVERFLOW_INTEN1	0	RW	Timer4 overflow interrupt enable 0: disable 1: enable
8	TIM4_MATCHING_INTEN1	0	RW	Timer4 matching interrupt enable 0: disable 1: enable
7	TIM3_OVERFLOW_INTEN1	0	RW	Timer3 overflow interrupt enable 0: disable 1: enable
6	TIM3_MATCHING_INTEN1	0	RW	Timer3 matching interrupt enable 0: disable 1: enable
5	TIM2_OVERFLOW_INTEN1	0	RW	Timer2 overflow interrupt enable 0: disable 1: enable
4	TIM2_MATCHING_INTEN1	0	RW	Timer2 matching interrupt enable 0: disable 1: enable
3	TIM1_OVERFLOW_INTEN1	0	RW	Timer1 overflow interrupt enable 0: disable

				1: enable
2	TIM1_MATCHING_INTEN1	0	RW	Timer1 matching interrupt enable 0: disable 1: enable
1	TIM0_OVERFLOW_INTEN1	0	RW	Timer0 overflow interrupt enable 0: disable 1: enable
0	TIM0_MATCHING_INTEN1	0	RW	Timer0 matching interrupt enable 0: disable 1: enable

Table 214 GPT_INT_EN2 (GPT_BASE_ADDR+0x48)

Bit	Field Name	Reset	RW	Description
9	TIM4_OVERFLOW_INTEN2	0	RW	Timer4 overflow interrupt enable 0: disable 1: enable
8	TIM4_MATCHING_INTEN2	0	RW	Timer4 matching interrupt enable 0: disable 1: enable
7	TIM3_OVERFLOW_INTEN2	0	RW	Timer3 overflow interrupt enable 0: disable 1: enable
6	TIM3_MATCHING_INTEN2	0	RW	Timer3 matching interrupt enable 0: disable 1: enable
5	TIM2_OVERFLOW_INTEN2	0	RW	Timer2 overflow interrupt enable 0: disable 1: enable
4	TIM2_MATCHING_INTEN2	0	RW	Timer2 matching interrupt enable 0: disable 1: enable
3	TIM1_OVERFLOW_INTEN2	0	RW	Timer1 overflow interrupt enable 0: disable 1: enable
2	TIM1_MATCHING_INTEN2	0	RW	Timer1 matching interrupt enable 0: disable 1: enable
1	TIM0_OVERFLOW_INTEN2	0	RW	Timer0 overflow interrupt enable 0: disable 1: enable
0	TIM0_MATCHING_INTEN2	0	RW	Timer0 matching interrupt enable 0: disable

				1: enable

5.18 Real Time Counter

The Real-time counter (RTC) module provides a generic, low power timer on the low-frequency clock source (32.768 KHz). The RTC module includes a 40-bit COUNTER, a 12-bit (1/X) pre-scaler, capture/compare registers. The RTC can be used to wakeup PMU and generate an interrupt.

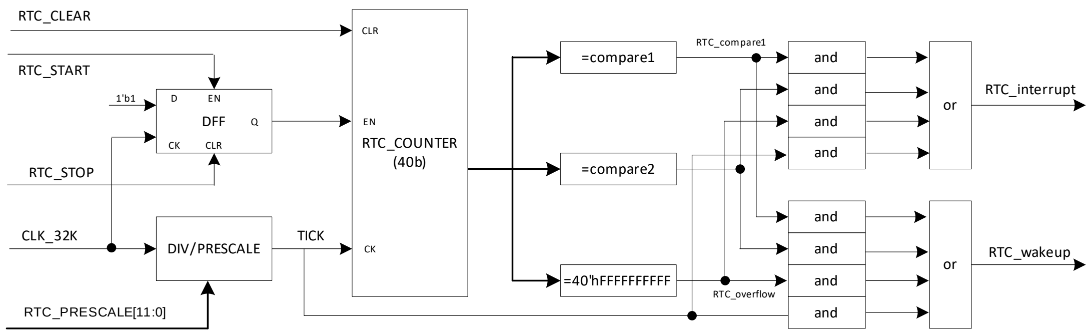


Figure 47 RTC Function Block

5.18.1 Feature

1. 40 bit Real Time Counter with 12 bit pre-scale with 32.768KHz working clock frequency
2. 2-channel compare and overflow events
3. Generate wakeup to PMU
4. Generate interrupt to CPU

5.18.2 Function Description

Tick

RTC tick frequency = 32.768 KHz / (RTC_PRESCALE+1). RTC counts by '1' every tick.

Overflow

RTC provides one quick overflow mechanism for software test. Through writing the register (RTC_TRIGOVRFLW), it sets the counter value to 0xFFFFFFFFF0, and overflow interrupt occurs when the counter overflows from 0xFFFFFFFFFF to 0.

Compare

There are two compare registers. The registers needs 3 cycles of 32.768 KHz to be valid. When setting a compare register, the following behavior should be noticed.

- COMPARE occurs when a register (RTC_COMP0/1) is N and the COUNTER value transitions from N to N+1.

- If a register (RTC_COMP0/1) is N and the COUNTER value is N when the START task is set, this will not trigger a COMPARE event.
- If the COUNTER is N, writing N+3 to the register (RTC_COMP0/1) is guaranteed to trigger a COMPARE event at N+3.
- If the COUNTER is N, writing N or N+1 or N+2 to a register (RTC_COMP0/1) may not trigger a COMPARE event.
- If the COUNTER is N and the current register (RTC_COMP0/1) value is (N+1) or (N+2) or (N+3), when a new register (RTC_COMP0/1) value is written, a match may trigger on the previous register (RTC_COMP0/1) value before the new value takes effect. If the current register (RTC_COMP0/1) value is greater than N+2 when the new value is written, there will be no event generated by the old setting.

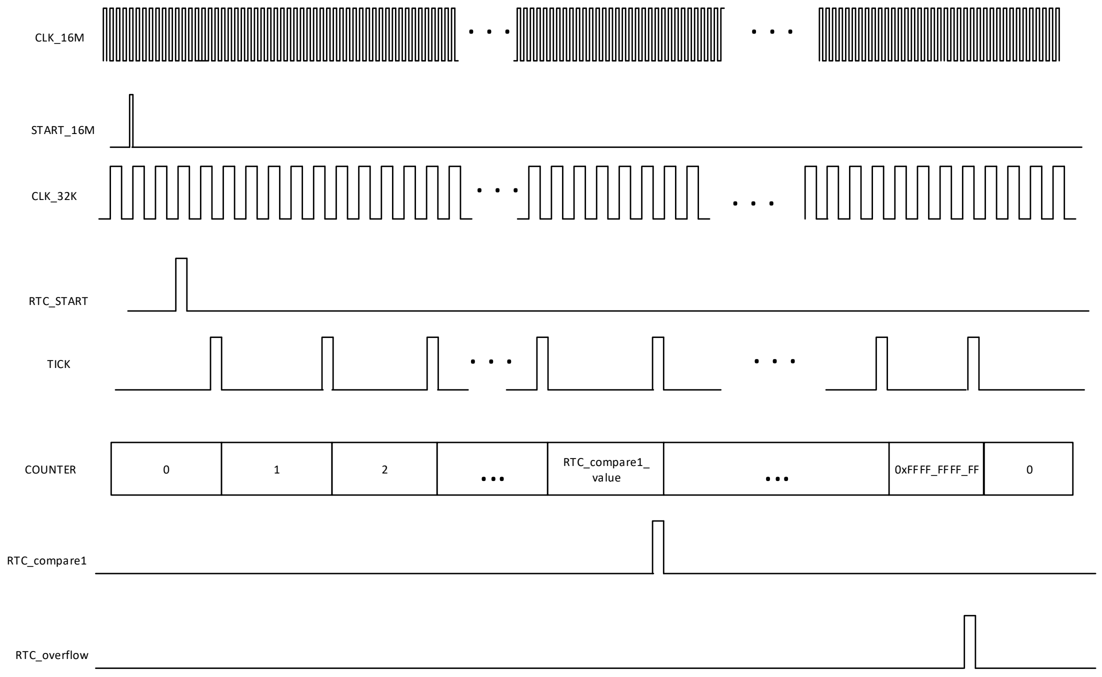


Figure 48 RTC Timing diagram

5.18.3 Signal Synchronous Time

RTC works in 32.768KHz clock domain. The register event trigger signals are in system 16M clock domain, so these signals need 3 cycles of 32.768KHz to take effect in 32.768KHz domain. The table below lists all these signals.

Table 215 Signal List Crossing Clock Domains

Signal Name	Signal Description
RTC_START_TRIG	RTC counter start trigger
RTC_STOP_TRIG	RTC counter start trigger
RTC_TRIGOVRFLW	RTC quick overflow counter set Used to generate overflow interrupt quickly.
RTC_CNT_CLR	RTC counter clear trigger
RTC_WAKEUP_MASK_TRIG	Load event for register (RTC_INT_MASK[7:4]) to take effect in 32KHz clock working domain

RTC_PRESCALE_TRIG	Load event for register (RTC_PRESCALE) to take effect in 32KHz clock working domain
RTC_COMP1_TRIG	Load event for register (RTC_COMP1) to take effect in 32KHz clock working domain
RTC_COM0_TRIG	Load event for register (RTC_COMP0) to take effect in 32KHz clock working domain

Since the registers (RTC_COMP0/1 and RTC_PRESCALE) need 3 cycles of the 32.768 KHz. Software can read back RTC_SET_FLG to see whether the synchronous is done. If done, writing the registers (RTC_COMP0_SET_CLR, RTC_COMP1_SET_CLR and RTC_PRESCALE_SET_CLR) can clear RTC_SET_FLG.

5.18.4 RTC Counter Read

To read the RTC Counter register, CPU needs to trigger register RTC_RD_FRZ first. When the read register value is N, the internal counter value may be N or N+1 (RTC tick transition may occur during reading).

5.18.5 Software Procedure

RTC flow.

1. Clear RTC counter by RTC_CNT_CLR. Wait 3 cycles of 32.768 KHz.
2. Set 40bit RTC compare value through register (RTC_COMP0/1)
3. Set 12bit RTC pre-scale value through register (RTC_PRESCALE)
4. Set RTC_TRIG_CFG to load pre-scale and compare value to 32.768 KHz domain (RTC_PRESCALE_TRIG, RTC_COMP0_TRIG and RTC_COMP1_TRIG). Wait 3 cycles of 32.768 KHz.
5. Start RTC by RTC_START_TRIG. Wait 3 cycles of 32.768 KHz.
6. Wait RTC interrupt
7. Write 1 to RTC_RD_FRZ to freeze 40 bits RTC read counter. Read RTC_CNT

5.18.6 RTC interrupt

RTC interrupt table is shown below.

Table 216 RTC interrupt

Interrupt Number	Interrupt name	Description
3	RTC_OVERFLOW_INT	RTC reaches 40'hffffffffff, then generate overflow interrupt
2	RTC_INCREMENT_INT	RTC increment event
1	RTC_COMP1_INT	RTC reaches the value of register (RTC_COMP1)
0	RTC_COMP0_INT	RTC reaches the value of register (RTC_COMP0)

5.18.7 RTC Registers

Table 217 RTC Instance

Base Address	Peripheral	Instance	Description
0x40016000	RTC	RTC	Real Timer Counter

Table 218 RTC Register Map

Address Offset	Name	Description
0x00	RTC_INT_STATUS	RTC Interrupt Status
0x04	RTC_INT_MASK	RTC Interrupt Enable
0x08	RTC_INT_CLR	RTC Interrupt Flag Clear Register
0x0C	RTC_TRIG_CFG	RTC Trigger Register
0x10	RTC_PRESCALE	RTC Pre-scale Register
0x14	RTC_COMP0_HI	RTC Compare0[39:32] Register
0x18	RTC_COMP0_LOW	RTC Compare0[31:0] Register
0x1C	RTC_COMP1_HI	RTC Compare1[39:32] Register
0x20	RTC_COMP1_LOW	RTC Compare1[31:0] Register
0x24	RTC_TRIG_COMP	RTC Compare Trigger Register
0x28	RTC_CNT_HI	RTC Counter[39:32]
0x2C	RTC_CNT_LOW	RTC Counter[31:0]
0x30	RTC_SET_FLG	RTC Set Done Flag Register

Table 219 RTC_INT_STATUS (RTC_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
3	RTC_OVERFLOW_INT	0	R	RTC overflow interrupt, RTC reaches 40’hffffffffff, then generate overflow interrupt. 0: invalid 1: valid
2	RTC_TICK_INT	0	R	counter increment event 0: invalid 1: valid
1	RTC_COMP1_INT	0	R	counter reaches register (RTC_COMP1) 0: invalid 1: valid
0	RTC_COMP0_INT	0	R	counter reaches register (RTC_COMP0) 0: invalid 1: valid

Table 220 RTC_INT_MASK (RTC_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
3	RTC_OVERFLOW_INT_MSK	0	RW	RTC overflow interrupt enable

				0: disable 1: enable
2	RTC_TICK_INT_MSK	0	RW	counter increment event interrupt enable 0: disable 1: enable
1	RTC_COMP1_INT_MSK	0	RW	RTC COMP1 interrupt enable 0: disable 1: enable
0	RTC_COMP0_INT_MSK	0	RW	RTC COMP0 interrupt enable 0: disable 1: enable

Table 221 RTC_INT_CLEAR (WDT_BASE_ADDR+0x08)

Bit	Field Name	Reset	RW	Description
3	RTC_OVERFLOW_INT_CLR	0	W	RTC overflow interrupt clear 0: invalid 1: clear
2	RTC_TICK_INT_CLR	0	W	RTC Interrupt clear 0: invalid 1: clear
1	RTC_COMP1_INT_CLR	0	W	RTC COMP1 interrupt clear 0: invalid 1: clear
0	RTC_COMP0_INT_CLR	0	W	RTC COMP0 interrupt clear 0: invalid 1: clear

Table 222 RTC_TRIG_CFG (RTC_BASE_ADDR+0x0C)

Bit	Field Name	Reset	RW	Description
4	RTC_RD_FRZ	0x0	W	RTC 40bit counter read trigger 0: invalid 1: trigger
3	RTC_TRIGOVRFLW	0	W	RTC trigger overflow RTC quick overflow counter set 0: invalid 1: 1: RTC counter be set to 0xfffffffff0 Note: Used to generate overflow interrupt quickly. This register need 2~3cycles of 32KHz time to be valid.
2	RTC_CNT_CLR	0x0	W	RTC counter clear trigger 0: invalid

				1: clear This register need 2~3cycles of 32KHz time to be valid.
1	RTC_STOP_TRIG	0x0	W	RTC counter start trigger 0: invalid 1: stop This register need 2~3cycles of 32KHz time to be valid.
0	RTC_START_TRIG	0x0	W	RTC counter start trigger 0: invalid 1: start This register need 2~3cycles of 32KHz time to be valid.

Table 223 RTC_PTESCALE (RTC_BASE_ADDR+0x10)

Bit	Field Name	Reset	RW	Description
11:0	RTC_PRESCALE	0x0	RW	Pre-scaler for RTC frequency fRTC = 32768Hz/(PRESCALER +1)

Table 224 RTC_COMP0_HI (RTC_BASE_ADDR+0x14)

Bit	Field Name	Reset	RW	Description
7:0	RTC_COMP0_HI	0xFF	RW	Register (RTC_COMP0) bit [39:32] Used to generate COMP0 interrupt when RTC counter reaches register (RTC_COMP0)

Table 225 RTC_COMP0_LOW (RTC_BASE_ADDR+0x18)

Bit	Field Name	Reset	RW	Description
31:0	RTC_COMP0_LOW	0xFFFFFFFF	RW	Register (RTC_COMP0) bit[31:0]

Table 226 RTC_COMP1_HI (RTC_BASE_ADDR+0x1C)

Bit	Field Name	Reset	RW	Description
11:0	RTC_COMP1_HI	0xFF	RW	Register (RTC_COMP1) bit [39:32] Used to generate COMP1 interrupt when RTC counter reaches register (RTC_COMP1)

Table 227 RTC_COMP1_LOW (RTC_BASE_ADDR+0x20)

Bit	Field Name	Reset	RW	Description
11:0	RTC_COMP1_LOW	0xFFFFFFFF	RW	Register (RTC_COMP1) bit[31:0]

Table 228 RTC_TRIG_COMP (RTC_BASE_ADDR+0x24)

Bit	Field Name	Reset	RW	Description
6	RTC_PRESCALE_SET_CLR	0x0	W	register (RTC_SET_FLG [2]) clear

				0: invalid 1:clear
5	RTC_COMP1_SET_CLR	0x0	W	register (RTC_SET_FLG [1]) clear 0: invalid 1:clear
4	RTC_COMP0_SET_CLR	0x0	W	register (RTC_SET_FLG [0]) clear 0: invalid 1:clear
3	RTC_WAKEUP_MASK_TRIG	0x0	W	Load event for register (RTC_INT_MASK [7:4]) to take effect in 32KHz clock working domain 0: invalid 1: load enable
2	RTC_PRESCALE_TRIG	0x0	W	Load event for register (RTC_PRESCALE) to take effect in 32KHz clock working domain 0: invalid 1: load enable
1	RTC_COMP1_TRIG	0x0	W	Load event for register (RTC_COMP1) to take effect in 32KHz clock working domain 0: invalid 1: load enable
0	RTC_COM0_TRIG	0x0	W	Load event for register (RTC_COMP0) to take effect in 32KHz clock working domain 0: invalid 1: load enable

Table 229 RTC_CNT_HI (RTC_BASE_ADDR+0x28)

Bit	Field Name	Reset	RW	Description
11:0	RTC_CNT_HI	0x0	R	RTC counter bit [39:32] Note: Trigger RTC_RD_FRZ before reading RTC_CNT_HI and RTC_CNT_LOW

Table 230 RTC_CNT_LOW (RTC_BASE_ADDR+0x2C)

Bit	Field Name	Reset	RW	Description
11:0	RTC_CNT_LOW	0x0	R	RTC counter bit [31:0]

Table 231 RTC_SET_FLG (RTC_BASE_ADDR+0x30)

Bit	Field Name	Reset	RW	Description
2	RTC_PRESCALE_SET_FLG	0x0	R	RTC pre-scale value to RTC 32KHz domain succeed flag
1	RTC_COMP1_SET_FLG	0x0	R	RTC 40bit COMP1 value to RTC 32KHz domain succeed flag
0	RTC_COMP0_SET_FLG	0x0	R	RTC 40bit COMP0 value to RTC 32KHz domain

				succeed flag

5.19 Watchdog Timer

The watchdog timer is a 32-bit timer that can be used to detect an unexpected execution sequence caused by a software run-away, and it can generate a full system reset or a Non-Maskable Interrupt (NMI).

5.19.1 Feature

1. 32 bits counter up to 131082 second with 30.52us step.
2. Non-Maskable Interrupt (NMI) or watchdog reset.
3. NMI interrupt is 16*30.52us ahead of watchdog reset, and it can be selected masked or not.

5.19.2 Function Description

The 32 bits watchdog timer is incremented by 1 every 32.768KHz cycle. The timer value can be configured through the register (WDT_COMPARE) which is set to 0xff at reset.

During write access the WDT_CTRL bits must be ‘0’. This provides extra filtering for a software run-away writing all ones to the WDT_CTRL. The timer is enabled or disabled by the register (WDT_CTRL).

If watchdog timer reaches (WDT_COMPARE-16), it generates a NMI interrupt. If Watchdog timer reaches the values of WDT_COMPARE, it generates a watchdog reset.

The NMI handler must write WDT_SERVICE_ TRI] to prevent the generation of a watchdog reset after receiving NMI. The watchdog reset is one of the system reset sources and resets the whole device, including setting the WDT_CNT_VALUE register to 0xff. The Cortex-M0 will start executing again from address zero while watchdog reset.

5.19.3 Software Procedure

Watchdog enable flow.

1. Clear timer counter (WDT_TRIG_CFG [WDT_CNT_CLR]) when PCLK and 32.768KHz clock on.
2. Set compare value (WDT_COMPARE).
3. Trigger compare value loading through setting WDT_TRIG_CFG [WDT_COMP_TRIG] to ‘1’.
4. Watchdog enable trigger through setting WDT_CTRL to 0x0.
5. Trigger watchdog enable through setting WDT_TRIG_CFG [WDT_WEN_TRIG] to ‘1’
6. Software should waits 3 cycles of 32.768 KHz to wait the signals to take effect in 32.768KHz domain. The signals include the above clear, compare value loading and watchdog enable trigger.
7. Software needs to service watchdog in period before watchdog reset. Software must service watchdog if

receiving NMI interrupt

Watchdog disable flow for hibernate or sleep mode:

1. Service watchdog timer through setting WDT_SERVICE_TRIG.
2. Wait 3 cycles of 32.768KHz to wait the service to take effect in 32.768KHz domain.
3. Disable watchdog timer interrupt through setting WDT_INT_MASK.
4. Disable CPU interrupts.
5. To disable watchdog timer, set WDT_CTRL[6:0] to 0x1 ~0x7F first.
6. Watchdog disable trigger through setting WDT_TRIG_CFG [WDT_WEN_TRIG] to ‘1’.
7. Wait 3 cycles of 32.768 KHz to wait the signal (watchdog disable trigger) to take effect in 32.768KHz domain.
8. Software goes to hibernate or sleep through “WFI” instruction.

5.19.4 Watchdog Interrupt

Watchdog interrupt table is shown below.

Table 232 Watchdog Interrupt

Interrupt Number	Interrupt name	Description
1	WDT_OVERFLOW_INT	Watchdog timer reaches 32’hfffffff0, then generate overflow interrupt
0	WDT_MATCHING_INT	Watchdog timer reaches the value of register (WDT_COMPARE–16), then generate matching interrupt

5.19.5 Watchdog timer Registers

Table 233 Watchdog Instance

Base Address	Peripheral	Instance	Description
0x40018000	WDT	WDT	Watchdog timer

Table 234 Watchdog Register Map

Address Offset	Name	Description
0x00	WDT_INT_STATUS	WDT Interrupt Status
0x04	WDT_INT_MASK	WDT Interrupt Enable
0x08	WDT_INT_CLEAR	WDT Interrupt Flag Clear Register
0x0C	WDT_TRIG_CFG	WDT Trigger Register
0x10	WDT_CTRL	WDT Enable Register
0x14	WDT_COMPARE	WDT Compare Register
0x18	WDT_CNT_VALUE	WDT Timer Counter

Table 235 WDT_INT_STATUS (WDT_BASE_ADDR+0x00)

Bit	Field Name	Reset	RW	Description
1	WDT_OVERFLOW_INT	0	R	WDT overflow interrupt, Watchdog timer reaches 32'hfffffff0, then generate overflow interrupt. 0: invalid 1: valid
0	WDT_MATCHING_INT	0	R	WDT matching interrupt, Watchdog timer reaches the value of register (WDT_COMPARE - 16), then generate matching interrupt. 0: invalid 1: valid

Table 236 WDT_INT_MASK (WDT_BASE_ADDR+0x04)

Bit	Field Name	Reset	RW	Description
1	WDT_OVERFLOW_INT_MSK	0	RW	WDT overflow interrupt enable 0: disable 1: enable
0	WDT_MATCHING_INT_MSK	0	RW	WDT matching interrupt enable 0: disable 1: enable

Table 237 WDT_INT_CLEAR (WDT_BASE_ADDR+0x08)

Bit	Field Name	Reset	RW	Description
1	WDT_OVERFLOW_INT_CLR	0	W	WDT overflow interrupt clear 0: invalid 1: clear
0	WDT_MATCHING_INT_CLR	0	W	WDT matching interrupt clear 0: invalid 1: clear

Table 238 WDT_TRIG_CFG (WDT_BASE_ADDR+0x0C)

Bit	Field Name	Reset	RW	Description
4	WDT_WEN_TRIG	0x0	W	WDT enable register trigger Load event for register (WDT_CTRL) to take effect in 32KHz clock working domain. The signal needs 3 cycles of 32KHz to be valid 0: invalid 1: load enable
3	WDT_TRIGOVRFLW	0	W	WDT trigger overflow Watchdog timer trigger event. The signal needs 3 cycles of 32KHz to be valid

				0: invalid 1: Watchdog timer be set to 0xffffffe0 Note: used to generate overflow interrupt quickly
2	WDT_COMP_TRIG	0	W	Watchdog compare register trigger Load event for register (WDT_COMPARE) to take effect in 32KHz clock working domain. The signal needs 3 cycles of 32KHz to be valid. 0: invalid 1: load enable
1	WDT_SERVICE_TRIG	0x0	W	WDT service trigger Watchdog timer service event. The signal need 3 cycles of 32KHz to be valid 0: invalid 1: software service to restart watchdog timer
0	WDT_CNT_CLR	0x0	W	WDT counter clear trigger 0:invalid 1:clear

Table 239 WDT_CTRL (WDT_BASE_ADDR+0x10)

Bit	Field Name	Reset	RW	Description
6:0	WDT_CTRL	0x0	RW	WDT timer enable 0:enable 1~127:disable

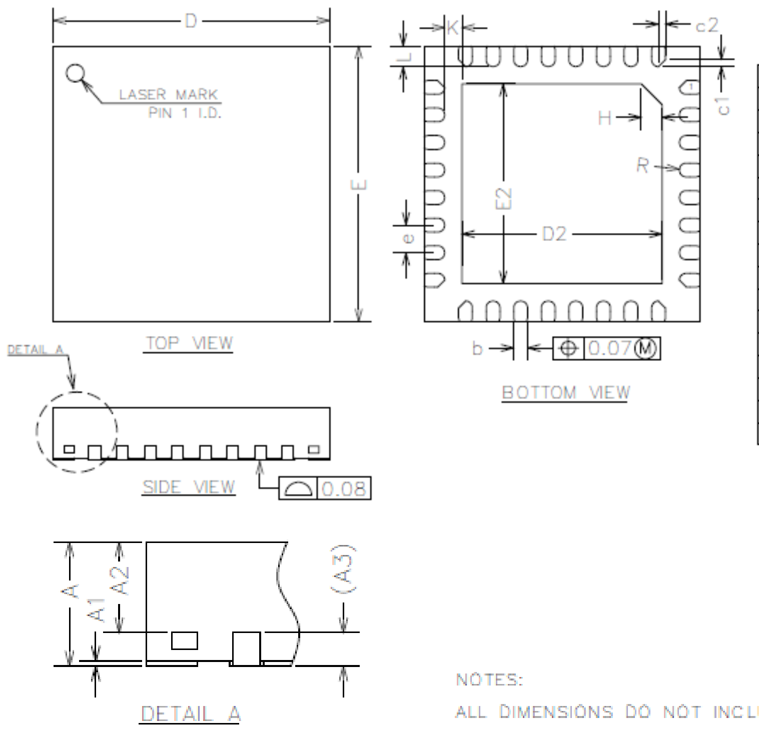
Table 240 WDT_COMPARE (WDT_BASE_ADDR+0x14)

Bit	Field Name	Reset	RW	Description
31:0	WDT_COMPARE	0x0	RW	WDT compare register used to generate matching interrupt and reset Note: Watchdog timer reaches the value of register (WDT_COMPARE) - 16, then generate matching interrupt Watchdog timer reaches the value of register (WDT_COMPARE), then generate reset. If generating overflow interrupt, this register must be zero

Table 241 WDT_CNT_VALUE (WDT_BASE_ADDR+0x18)

Bit	Field Name	Reset	RW	Description
31:0	WDT_CNT_VALUE	0x0	R	WDT timer counter

6 Mechanical Information



COMMON DIMENSIONS
(UNITS OF MEASURE=MILLIMETER)

SYMBOL	MIN	NOM	MAX
A	0.70	0.75	0.80
A1	0	0.02	0.05
A2	0.50	0.55	0.60
A3		0.20REF	
b	0.15	0.20	0.25
D	3.90	4.00	4.10
E	3.90	4.00	4.10
D2	2.80	2.90	3.00
E2	2.80	2.90	3.00
e	0.30	0.40	0.50
H		0.30REF	
K		0.25REF	
L	0.25	0.30	0.35
R	0.09	–	–
c1	–	0.10	–
c2	–	0.10	–

NOTES:
ALL DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS.

Figure 49 QFN32 Package Outline Drawing

MXD2656A1 Datasheet

Room 202, No. 50 Bo Xia Road,

Zhangjiang, Shanghai, China (201203)

TEL: +86-21-61006488

FAX: +86-21-61009682

E-MAIL: info@maxscend.com

WEB: www.maxscend.com