ISD91500

Technical Reference Manual

Multi-Algorithm Voice Processor

With Headphone Driver

The information described in this document is the exclusive intellectual property of
Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of *ISD91500* series microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

Table of Contents

List of Tables

List of Figures

1 General Description

ISD91500 series is an audio system-on-chip (SoC) integrated with high-quality audio features optimized for low power, audio record and playback.

ISD91500 can run up to 49 MHz with embedded ARM® Cortex™-M0 32-bit microcontroller core and 64K-byte non-volatile flash memory which has additional 6KB boot loader flash and 20K-byte embedded SRAM. The audio features include microphone input ADC and stereo DAC with headphone ouput driver. ISD91500 also comes with a wide variety of peripherals, such as Timers, Watchdog Timer (WDT), Peripheral Direct Memory Access (PDMA), Serial interfaces (UART, SPI, I2C and I2S), USB, PWM modulators, GPIO, SDADC, SARADC, DAC and Low Voltage Detector.

ISD91500 series supports a rich set of power saving modes including Deep Power Down (DPD) mode drawing less than 2uA. A micro-power 10KHz oscillator enables the device to periodically wake up from deep power down to check other events.

Typical applications include USB headsets, Gaming controllers, USB/I2S audio bridge and voice communication systems.

2 Features

- **Core**
  - ARM® Cortex™-M0 core running up to 49 MHz
  - One 24-bit system tick timer for operating system support
  - Single-cycle 32-bit hardware multiplier
  - NVIC (Nested Vetctor Interrupt Controller) for the interrupt inputs, each with 4-levels of priority
  - Serial Wire Debug (SWD) supports with 2 watchpoints/4 breakpoints
- **Power Management**
  - Wide operating voltage range from 1.7V to 3.6V (VCCD voltage level equal to VCCA)
  - Power Management Unit (PMU) providing different levels of power control
  - Deep Power Down (DPD) mode with specific register retention for lowest power state (typically <2uA)
  - Wake-up from DPD via LIRC timed operation
  - Standby Power Down(STOP) mode with RAM retention for lowest power state (typically <15uA)
  - Wake-up from STOP via any GPIOs or WDT interrupts
  - Fast wake-up mechanism with 20us CPU running instruction
- **Flash EPROM Memory**
  - 64KB Flash EPROM for program code and data storage
  - Additional 6KB of flash configured as boot sector for ISP loader
  - Support ISP and ICP code update
  - 512B page erase for embedded flash
  - Configurable boundary to delineate code and data flash
  - Support 2-wire In-circuit Programming (ICP) update from SWD ICE interface

- **SRAM Memory**
  - 20KB embedded SRAM
- **Clock Control**
  - 48 MHz/49.152MHz configurable internal high speed RC oscillator (HIRC) for system operation
  - 10 kHz internal low speed RC oscillator (LIRC) for Watchdog Timer and wake-up operation
  - 4~24.576 MHz external high speed crystal oscillator (HXT) for precise timing operation
  - One PLL sourced from HIRC, HXT, and XCLK input
  - Clock Doubler(XCLK) minimum input frequency 512KHz
  - Clock failure detection for high speed external crystal oscillator
- **GPIO**
  - Up to 50 GPIOs individually configurable as three I/O modes
    - Input with pull-up option
    - Push-Pull output
    - Open-Drain output
  - Schmitt trigger and slew rate selectable by quad
  - GPIO configureable as interrupt/wake-up source with edge/level setting
  - GPA group power level follow VCCD level and GPB/GPC/GPD group follow VCCPST level
  - Support 5V tolerance
- **Companding**
  - Compatible with CCITT G.726 and G.726 AnnexA
  - 8-bit PCM or 16-bit linear interfacing
  - 4 kinds ADPCM rates : 40, 32, 24, 16 kbps
  - 8-bit PCM u-law or A-law
  - Support full-duplex for encoding and decoding with different sampling rates
  - 8-level FIFO with interrupt
- **SARADC**
  - 12-bit SAR ADC with 12-bit output
  - 10-bits accuracy guaranteed
  - 12-ch external single-ended input
  - 200K SPS under VCCA (2.5V~3.6V)
  - DMA support for min CPU intervention
- **Audio Analog to Digital Converter**
  - Mono Sigma-Delta ADC with configurable decimation filter and 16 bit output
  - 16-level word FIFO data buffer with mono 32/24/16/8 bits data width
  - Programmable gain amplifier with 32 steps from -12 to 34.5dB in 1.5dB steps
  - Boost gain stage of 26dB, giving maximum total gain of 60.5dB
  - Input from optional gain dedicated from MIC pin
  - Optional 8 level output voltage (50%, 60%, 75%, 90% of VCCA, 1.2V, 1.7V, 2.0V, 2.4V)
  - Support FIFO threshold setting for interrupt
  - Sample rate 8K-48KHz

- DMA support for minimal CPU intervention

- **24bit Stereo Audio DAC with Headphone Output (Class-AB)**
  - -75dB Total Harmonic Distortion (THD+N) performance
  - 90dB Signal-to-Noise (SNR) performance
  - 20mW per channel drive capability into 32Ω load @3.3V
  - 16-level 2-word FIFO data buffer with stereo 32/24/16/8 bits data width
  - Programmable digital volume control from -80dB to 6dB
  - Programmable analog headphone output volume control from -57dB to 6dB
  - Support sample rates from 8 KHz – 48 KHz
  - DMA support for minimal CPU intervention

- **UART**
  - Two UART ports with flow control (TX, RX, CTS and RTS)
  - 16 byte FIFO for receive and transmit data payloads
  - Programmable baud-rate generator max up to 4M (additional 1000000/1843200/3250000)
  - Programmable receiver buffer trigger level
  - DMA support for minimal CPU intervention

- **I2C**
  - Up to two I2C controllers
  - Master/Slave up to 1Mbit/s
  - Bidirectional data transfer between masters and slaves
  - Multi-master bus (no central master)
  - Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
  - Serial clock synchronization with different bit rates to communicate via one serial bus
  - Serial clock synchronization configurable as handshake mechanism to suspend and resume serial transfer
  - Programmable clock allowing versatile rate control
  - Support multiple address recognition (four slave address with mask option)

- **PDMA**
  - 8-channel DMAs support data transfer between SRAM and peripherals of SARADC, SDADC, DAC, SPI0, I2S, UART0/1

- **Timers**
  - Three timers with 8-bit pre-scalar and 16-bit resolution
  - Counter auto reloaded
  - Timer1 can be IR carrier generator

- **Watch Dog Timer**
  - Multiple clock sources
  - 8 selectable time-out period from micro seconds ~ seconds (depends on clock source)
  - WDT can wake up power down/sleep

- Interrupt or reset selectable on watchdog time-out

- **PWM/Capture Timer**
  - 2 sets supporting up to 8 individual PWM outputs
  - Built-in two sets of one 16-bit timer and four 16-bit comparators supporting up to 8 single-ended PWM outputs or 4 complementary paired PWM outputs
  - The PWM generator equipped with a clock source selector, a clock divider, an 8-bit pre-scalar for each
  - PWM set and Dead-Zone generator for complementary paired PWM
  - PWM interrupt synchronous to PWM period
  - 16-bit digital capture timers (shared with PWM timers) providing rising/falling capture inputs
  - Capture interrupt

- **SPI**
  - 2 sets of SPI interface
  - SPI Clock up to 25 MHz
  - SPI data rate in Quad mode of 100 Mbps
  - SPI master/slave mode
  - MSB or LSB first data transfer
  - 2 slave/device select lines when used in master mode
  - 8-level 32-bit FIFO
  - DMA support
  - Quad/Dual SPI support

- **I^2S**
  - 1 set of I2S interface
  - Master mode and Slave mode
  - Capable of handling 8, 16, 24 and 32 bits data sizes in each audio channel
  - Monaural and stereo audio data
  - I2S protocols: Philips standard, MSB-justified, and LSB-justified data format
  - PCM protocols: PCM standard, MSB-justified, and LSB-justified data format
  - Interrupt requests generated when buffer levels cross a programmable boundary
  - Support two 16-level FIFO data buffers, one for transmitting and the other for receiving
  - Support two PDMA requests, one for transmitting and the other for receiving

- **USB 2.0 Device Controller**
  - Compliant with USB 2.0 full-speed specification
  - 1 interrupt vector with 4 different interrupt events (NEVWK, VBUSDET, USB and BUS)
  - Support Control/Bulk/Interrupt/Isochronous transfer type
  - Support suspend function when no bus activity existing for 3 ms
  - 12 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 1k bytes buffer size
  - Remote wake-up capability

- **Brown-Out Detector (BOD)**
  - with 11 levels: 1.8/1.9/2.0/2.1/2.2/2.4/2.6/2.8/3.0/3.1/3.4V with VCCD detection
  - Support Brown-out Interrupt or Reset option
- **Low Voltage Reset: 1.6V with VCCD detection**
- **Operating Temperature: -40℃~85℃**
- **Package**
  - All Green package (RoHS)
  - LQFP64
  - QFN48

3 Part Information and Pin Configuration

3.1 LQFP 64-Pin Diagram

3.1.1 I91535ADI

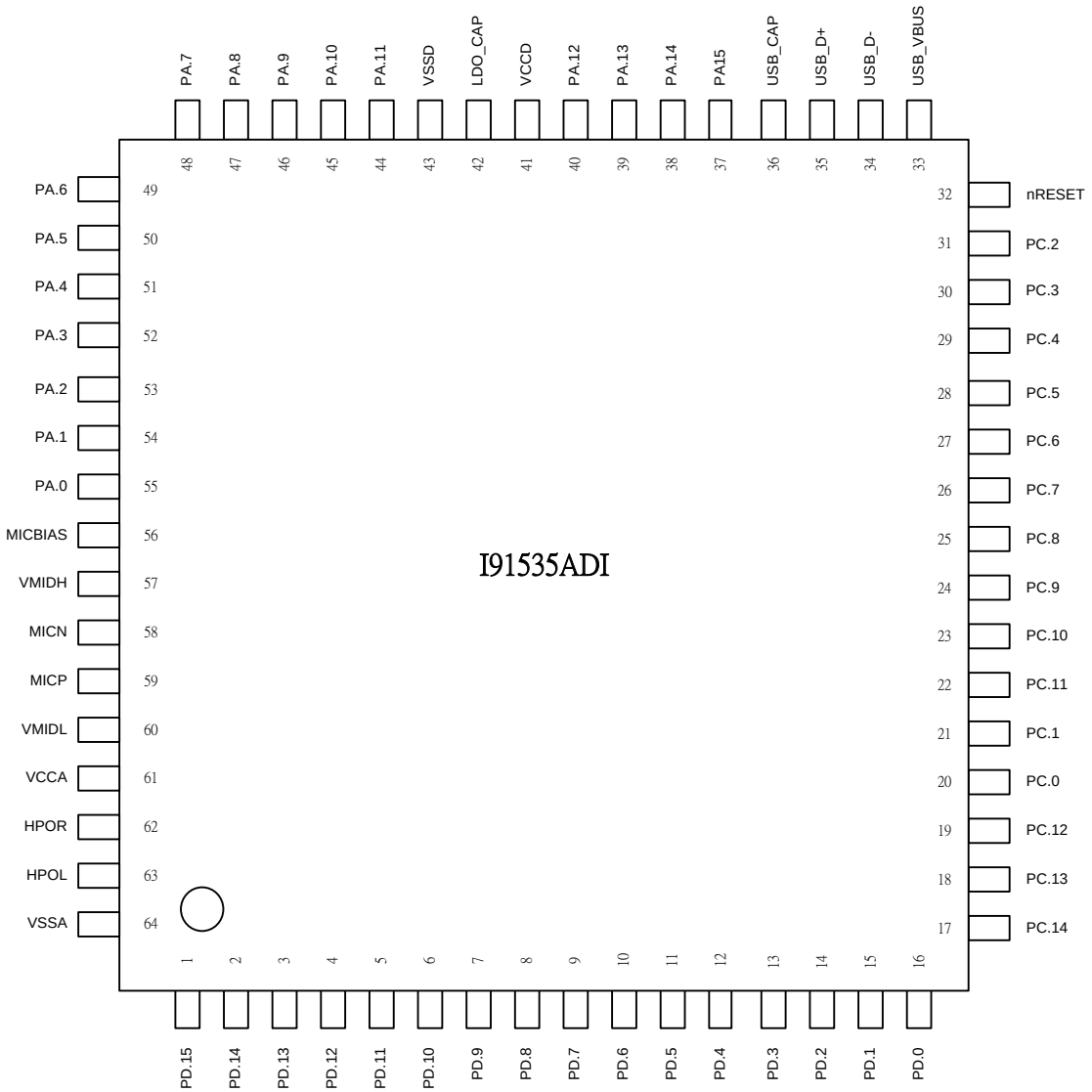


Figure 3.1-1 I91535ADI Pin Diagram

3.1.2 I91535H02DI

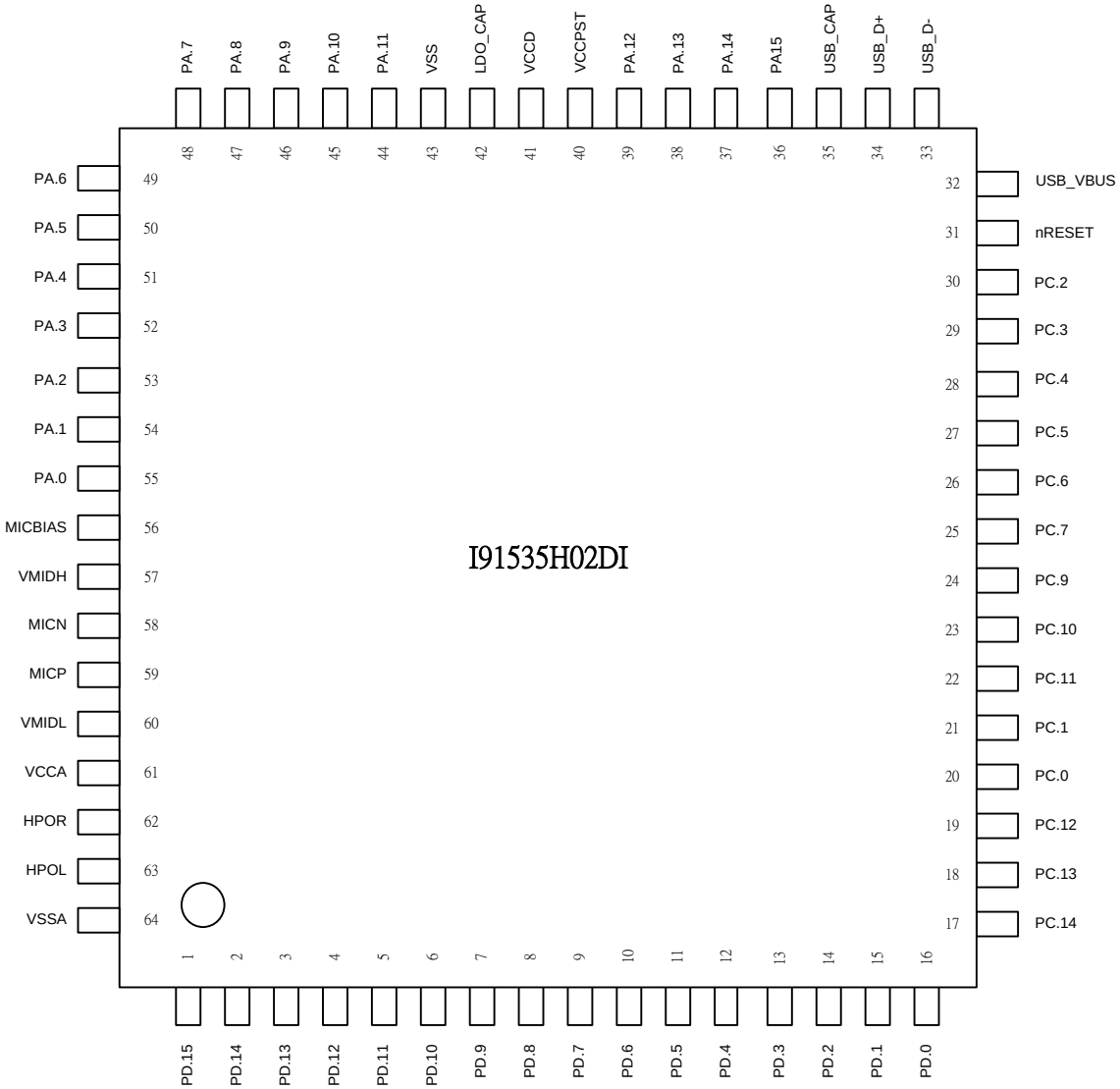


Figure 3.1-2 LQFP64 Type2 Pin Diagram

3.2 QFN 48-Pin Diagram

3.2.1 I91535AQI

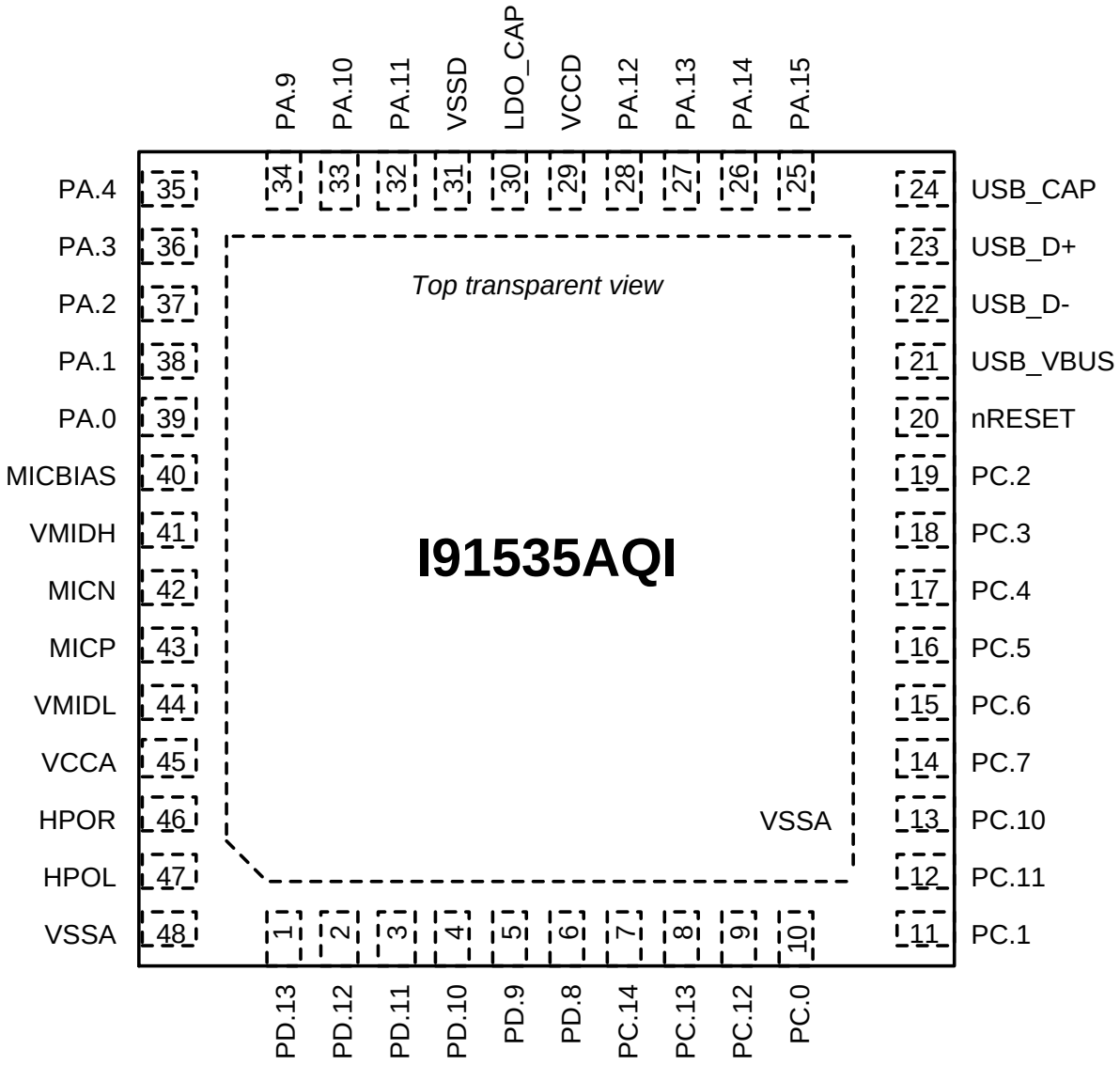


Figure 3.2-1 QFN48 Type1 Pin Diagram

3.2.2 I91535H02QI

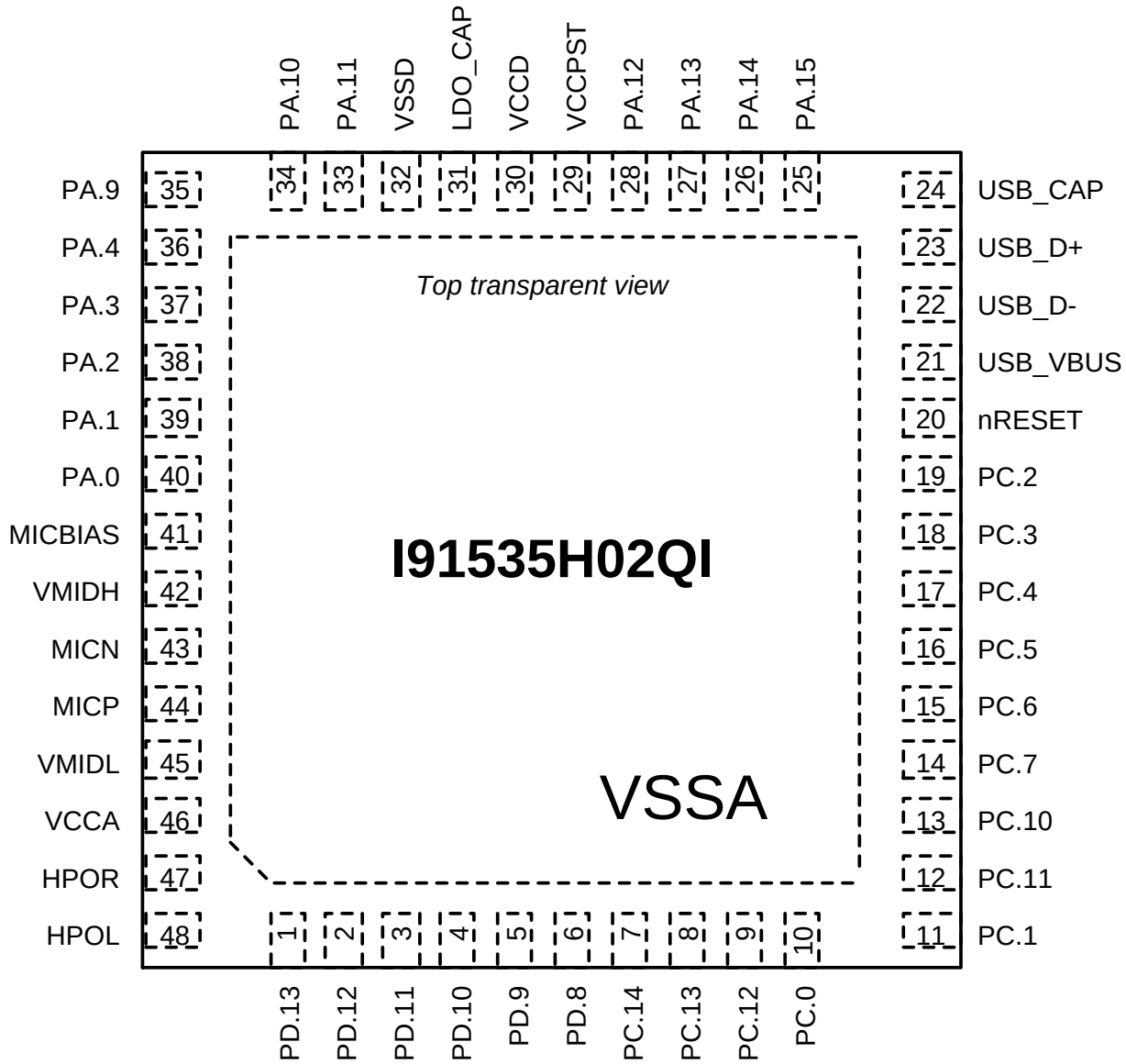


Figure 3.2-2 QFN48 Type2 Pin Diagram

3.3 Pin/Pad Description

3.3.1 LQFP64

I91535 ADI Pin No.	I91535 H02DI Pin No.	Name	Type	Alt CFG	Description
1	1	PD.15	**I/O**	0	General purpose input/output pin; port D, bit15
		I2C0_SDA	**O**	1	I2C0 data input/output pin
		SPI1_MOSI1	**O**	3	SPI1 Master Out Slave In 1
2	2	PD.14	**I/O**	0	General purpose input/output pin; port D, bit14
		I2C0_SCL	**I/O**	1	I2C0 clock pin
		SPI0_SS1	**O**	2	SPI0 Slave Select 1
		SPI1_MISO1	**I/O**	3	SPI1 Master In Slave Out 1
3	3	PD.13	**I/O**	0	General purpose input/output pin; port D, bit13
		I2C1_SDA	**I/O**	1	I2C1 data input/output pin
		SPI0_MISO1	**I/O**	2	SPI0 Master In Slave Out 1
		ICE_DAT	**I/O**	3	SWD Interface, Serial Data
4	4	PD.12	**I/O**	0	General purpose input/output pin, port D, bit12
		I2C1_SCL	**I/O**	1	I2C1 clock pin
		SPI0_MOSI1	**I/O**	2	SPI0 Master Out Slave In 1
		ICE_CLK	**I/O**	3	SWD Interface, Serial Clock
5	5	PD.11	**I/O**	0	General purpose input/output pin, port D, bit11
		PWM1_ch3	**I/O**	1	PWM1 channel 3 output
		SPI0_MOSI0	**I/O**	2	SPI0 Master Out Slave In 0
		I2C1_SDA	**I/O**	3	I2C1 data input/output pin
6	6	PD.10	**I/O**	0	General purpose input/output pin, port D, bit10
		PWM1_ch2	**I/O**	1	PWM1 channel 2 output
		SPI0_CLK	**I/O**	2	SPI0 Serial Clock
		I2C1_SCL	**I/O**	3	I2C1 clock pin
7	7	PD.9	**I/O**	0	General purpose input/output pin, port D, bit9
		PWM1_ch1	**I/O**	1	PWM1 channel 1 output
		SPI0_MISO0	**I/O**	2	SPI0 Master In Slave Out 0

I91535 ADI Pin No.	I91535 H02DI Pin No.	Name	Type	Alt CFG	Description
		UART0_RX	**I**	3	UART0 Receiver Serial In
8	8	PD.8	**I/O**	0	General purpose input/output pin, port D, bit8
		PWM1_ch0	**I/O**	1	PWM1 channel 0 output
		SPI0_SS0	**I/O**	2	SPI0 Slave Select 0
		UART0_TX	**O**	3	UART0 Transmitter Serial Out
9	9	PD.7	**I/O**	0	General purpose input/output pin, port D, bit7
		PWM0_ch3	**I/O**	1	PWM0 channel 3 output
		SPI1_MISO0	**I/O**	3	SPI1 Master In Slave Out 0
10	10	PD.6	**I/O**	0	General purpose input/output pin, port D, bit6
		PWM0_ch2	**I/O**	1	PWM0 channel 2 output
		SPI1_CLK	**I/O**	3	SPI1 Serial Clock
11	11	PD.5	**I/O**	0	General purpose input/output pin, port D, bit5
		PWM0_ch1	**I/O**	1	PWM0 channel 1 output
		SPI1_MOSI0	**I/O**	3	SPI1 Master Out Slave In 0
12	12	PD.4	**I/O**	0	General purpose input/output pin, port D, bit4
		PWM0_ch0	**I/O**	1	PWM0 channel 0 output
		CAP0	**I**	2	Capture input based on PWM0 timer
		I2S0_DI	**I**	3	I2S0 Data input pin
13	13	PD.3	**I/O**	0	General purpose input/output pin, port D, bit3
		UART1_RX	**I**	1	UART1 Receiver Serial In
		PWM0_ch3	**I**	2	PWM0 channel 3 output
		I2S0_DO	**O**	3	I2S0 Data output pin
14	14	PD.2	**I/O**	0	General purpose input/output pin, port D, bit2
		UART1_TX	**O**	1	UART1 Transmitter Serial Out
		PWM0_ch2	**I**	2	PWM0 channel 2 output
		I2S0_BCLK	**I/O**	3	I2S0 Bit Clock pin
15	15	PD.1	**I/O**	0	General purpose input/output pin, port D, bit1
		UART1_nRTS	**O**	1	UART1 Request To Send Output
		PWM0_ch1	**I**	2	PWM0 channel 1 output

I91535 ADI Pin No.	I91535 H02DI Pin No.	Name	Type	Alt CFG	Description
		I2S0_LRCK	**I/O**	3	I2S0 left right channel clock pin
16	16	PD.0	**I/O**	0	General purpose input/output pin, port D, bit0
		UART1_nCTS	**I**	1	UART1 Clear To Send Input
		PWM0_ch0	**I/O**	2	PWM0 channel 0 output
		I2S0_MCLK	**O**	3	I2S0 Master Clock Output pin
17	17	PC.14	**I/O**	0	General purpose input/output pin, port C, bit14
		SPI0_SS0	**I/O**	1	SPI0 Slave Select 0
		I2S0_DI	**I**	2	I2S0 Data input pin
		UART1_RX	**I**	3	UART1 Receiver Serial In
18	18	PC.13	**I/O**	0	General purpose input/output pin, port C, bit13
		SPI0_MISO0	**I/O**	1	SPI0 Master In Slave Out 0
		I2S0_DO	**O**	2	I2S0 Data output pin
		UART1_TX	**O**	3	UART1 Transmitter Serial Out
19	19	PC.12	**I/O**	0	General purpose input/output pin, port C, bit12
		SPI0_CLK	**I/O**	1	SPI0 Serial Clock
		I2S0_BCLK	**I/O**	2	I2S0 Bit Clock pin
		UART1_nRTS	**O**	3	UART1 Request To Send Output
20	20	PC.0	**I/O**	0	General purpose input/output pin, port C, bit0
		XT1_OUT	**O**	1	External 4~24.576 MHz(high speed) crystal output pin
		I2C0_SCL	**I/O**	3	I2C0 clock pin
21	21	PC.1	**I/O**	0	General purpose input/output pin, port C, bit1
		XT1_IN	**I**	1	External 4~24.576 MHz(high speed) crystal input pin
		I2C0_SDA	**I/O**	3	I2C0 data input/output pin
22	22	PC.11	**I/O**	0	General purpose input/output pin, port C, bit11
		SPI0_MOSI0	**I/O**	1	SPI0 Master Out Slave In 0
		I2S0_LRCK	**I/O**	2	I2S0 left right channel clock pin
		UART1_nCTS	**I**	3	UART1 Clear To Send Input
23	23	PC.10	**I/O**	0	General purpose input/output pin, port C, bit10
		SPI0_MOSI1	**I/O**	1	SPI0 Master Out Slave In 1

I91535 ADI Pin No.	I91535 H02DI Pin No.	Name	Type	Alt CFG	Description
		I2S0_MCLK	**O**	2	I2S0 Master Clock Output pin
		MCLKI	**I**	3	External Clock Input
24	24	PC.9	**I/O**	0	General purpose input/output pin, port C, bit9
		SPI0_MISO1	**I/O**	1	SPI0 Master In Slave Out 1
		PWM1_ch3	**O**	3	PWM1 channel 3 output
		PWM1_ch2	**O**	3	PWM1 channel 2 output
25		PC.8	**I/O**	0	General purpose input/output pin, port C, bit8
		SPI0_SS1	**O**	1	SPI0 Slave Select 1
		I2S0_MCLK	**O**	2	I2S0 Master Clock Output pin
		PWM1_ch2	**O**	3	PWM1 channel 2 output
26	25	PC.7	**I/O**	0	General purpose input/output pin, port C, bit7
		I2C0_SDA	**I/O**	1	I2C0 data input/output pin
		SPI1_SS1	**O**	2	SPI1 Slave Select 1
		PWM1_ch1	**O**	3	PWM1 channel 1 output
27	26	PC.6	**I/O**	0	General purpose input/output pin, port C, bit6
		I2C0_SCL	**I/O**	1	I2C0 clock pin
		SPI1_SS0	**I/O**	2	SPI1 Slave Select 0
		PWM1_ch0	**O**	3	PWM1 channel 0 output
28	27	PC.5	**I/O**	0	General purpose input/output pin, port C, bit5
		UART0_RX	**I**	1	UART0 Receiver Serial In
		I2S0_DI	**I**	2	I2S0 Data input pin
		PWM0_ch3	**O**	3	PWM0 channel 3 output
29	28	PC.4	**I/O**	0	General purpose input/output pin, port C, bit4
		UART0_TX	**O**	1	UART0 Transmitter Serial Out
		I2S0_DO	**O**	2	I2S0 Data output pin
		PWM0_ch2	**O**	3	PWM0 channel 2 output
30	29	PC.3	**I/O**	0	General purpose input/output pin, port C, bit3
		UART0_nRTS	**O**	1	UART0 Request To Send Output
		I2S0_BCLK	**I/O**	2	I2S0 Bit Clock pin

I91535 ADI Pin No.	I91535 H02DI Pin No.	Name	Type	Alt CFG	Description
		PWM0_ch1	**O**	3	PWM0 channel 1 output
31	30	PC.2	**I/O**	0	General purpose input/output pin, port C, bit2
		UART0_nCTS	**I**	1	UART0 Clear To Send Input
		I2S0_LRCK	**I/O**	2	I2S0 left right channel clock pin
		PWM0_ch0	**O**	3	PWM0 channel 0 output
32	31	nRESET	**I**		Reset input, low active, internal pull-high
33	32	USB_VBUS	**P**		Power supply from USB or HUB
34	33	USB_D-	**A**		USB differential signal D-
35	34	USB_D+	**A**		USB differential signal D+
36	35	USB_CAP	**P**		USB Internal regulator output decoupling pin. A 1uF cap returning to VSSD must be placed on it.
37	36	PA.15	**I/O**	0	General purpose input/output pin, port A, bit15
		SPI0_SS0	**I/O**	1	SPI0 Slave Select 0
		UART1_RX	**I**	2	UART1 Receiver Serial In
38	37	PA.14	**I/O**	0	General purpose input/output pin, port A, bit14
		SPI0_MISO0	**I/O**	1	SPI0 Master In Slave Out 0
		UART1_TX	**O**	2	UART1 Transmitter Serial Out
39	38	PA.13	**I/O**	0	General purpose input/output pin, port A, bit13
		SPI0_CLK	**I/O**	1	SPI0 Serial Clock
		UART0_nRTS	**O**	3	UART0 Request To Send Output
40	39	PA.12	**I/O**	0	General purpose input/output pin, port A, bit12
		SPI0_MOSI0	**I/O**	1	SPI0 Master Out Slave In 0
		I2C1_SDA	**I/O**	2	I2C1 data input/output pin
		UART0_nCTS	**I**	3	UART0 Clear To Send Input
	40	VCCPST	**P**		Digital IO Power Supply(Include GPB,GPC,GPD)
41	41	VCCD	**P**		Main Digital Power Supply(Also include GPA IO power)
42	42	LDOCAP	**P**		Regulator output decoupling pin for core logic. A 1uF cap returning to VSSD must be placed on it.
43	43	VSSD	**P**		Digital ground
44	44	PA.11	**I/O**	0	General purpose input/output pin, port A, bit11

I91535 ADI Pin No.	I91535 H02DI Pin No.	Name	Type	Alt CFG	Description
		SPI0_MOSI1	I/O	1	SPI0 Master Out Slave In 1
		I2C1_SCL	I/O	2	I2C1 clock pin
		UART0_RX	I	3	UART0 Receiver Serial In
45	45	PA.10	I/O	0	General purpose input/output pin, port A, bit10
		SPI0_MISO1	I/O	1	SPI0 Master In Slave Out 1
		MCLKI	I	2	External Clock Input
		UART0_TX	O	3	UART0 Transmitter Serial Out
46	46	PA.9	I/O	0	General purpose input/output pin, port A, bit9
		SPI0_SS1	O	1	SPI0 Slave Select 1
		UART0_RX	I	2	UART0 Receiver Serial In
		PWM1_ch3	O	3	PWM1 channel 3 output
47	47	PA.8	I/O	0	General purpose input/output pin, port A, bit8
		TMR2	I	1	Timer2 external clock input
		UART0_TX	O	2	UART0 Transmitter Serial Out
		PWM1_ch2	O	3	PWM1 channel 2 output
48	48	PA.7	I/O	0	General purpose input/output pin, port A, bit7
		TMR1	I	1	Timer1 external clock input
		SPI1_SS1	O	2	SPI1 Slave Select 1
		PWM1_ch1	O	3	PWM1 channel 1 output
49	49	PA.6	I/O	0	General purpose input/output pin, port A, bit6
		TMR0	I	1	Timer0 external clock input
		CAP1	I	2	Capture input based on PWM1 timer
		PWM1_ch0	O	3	PWM1 channel 0 output
50	50	PA.5	I/O	0	General purpose input/output pin, port A, bit5
		MCLKI	I	1	External Clock Input
		SPI1_SS0	I/O	3	SPI1 Slave Select 0
51	51	PA.4	I/O	0	General purpose input/output pin, port A, bit4
		I2S0_DI	I	1	I2S0 Data input pin
		UART1_RX	I	2	UART1 Receiver Serial In

I91535 ADI Pin No.	I91535 H02DI Pin No.	Name	Type	Alt CFG	Description
		SPI1_MISO0	**I/O**	3	SPI1 Master In Slave Out 0
52	52	PA.3	**I/O**	0	General purpose input/output pin, port A, bit3
		I2S0_DO	**O**	1	I2S0 Data output pin
		UART1_TX	**O**	2	UART1 Transmitter Serial Out
		SPI1_CLK	**I/O**	3	SPI1 Serial Clock
53	53	PA.2	**I/O**	0	General purpose input/output pin, port A, bit2
		I2S0_BCLK	**I/O**	1	I2S0 Bit Clock pin
		I2C0_SDA	**I/O**	2	I2C0 data input/output pin
		SPI1_MOSI0	**I/O**	3	SPI1 Master Out Slave In 0
54	54	PA.1	**I/O**	0	General purpose input/output pin, port A, bit1
		I2S0_LRCK	**I/O**	1	I2S0 left right channel clock pin
		I2C0_SCL	**I/O**	2	I2C0 clock pin
		SPI1_MOSI1	**I/O**	3	SPI1 Master Out Slave In 1
55	55	PA.0	**I/O**	0	General purpose input/output pin, port A, bit0
		I2S0_MCLK	**O**	1	I2S0 Master Clock Output pin
		IR	**O**	2	IR carrier generator based on Timer1 interval
		SPI1_MISO1	**I/O**	3	SPI1 Master In Slave Out 1
56	56	MICBIAS	**A**		Microphone Bias Output
57	57	VMIDH	**P**		Mid Rail Reference High, Connect 4.7uF to VSSA
58	58	MICN	**A**		Negative Microphone Input
59	59	MICP	**A**		Positive Microphone Input
60	60	VMIDL	**P**		Mid Rail Reference Low, Connect 4.7uF to VSSA
61	61	VCCA	**P**		Main Analog Circuitry Power Supply
62	62	HPOR	**A**		Headphone Right Channel Output
63	63	HPOL	**A**		Headphone Left Channel Output
64	64	VSSA	**P**		Ground for Analog Circuitry

Note1:

I: Digital Input, O: Digital Output, A: Analog pin, P: Power Pin

Note2:

VCCD & VCCA must be supplied same voltage level, and VCCD must larger or equal to VCCPST.
For I91535ADI, VCCD pin also supply all IO power.

3.3.2 QFN48

I91535 AQI Pin No.	I91535 H02QI Pin No.	Name	Type	Alt CFG	Description
1	1	PD.13	**I/O**	0	General purpose input/output pin; port D, bit13
		I2C1_SDA	**I/O**	1	I2C1 data input/output pin
		SPI0_MISO1	**I/O**	2	SPI0 Master In Slave Out 1
		ICE_DAT	**I/O**	3	SWD Interface, Serial Data
2	2	PD.12	**I/O**	0	General purpose input/output pin, port D, bit12
		I2C1_SCL	**I/O**	1	I2C1 clock pin
		SPI0_MOSI1	**I/O**	2	SPI0 Master Out Slave In 1
		ICE_CLK	**I/O**	3	SWD Interface, Serial Clock
3	3	PD.11	**I/O**	0	General purpose input/output pin, port D, bit11
		PWM1_ch3	**I/O**	1	PWM1 channel 3 output
		SPI0_MOSI0	**I/O**	2	SPI0 Master Out Slave In 0
		I2C1_SDA	**I/O**	3	I2C1 data input/output pin
4	4	PD.10	**I/O**	0	General purpose input/output pin, port D, bit10
		PWM1_ch2	**I/O**	1	PWM1 channel 2 output
		SPI0_CLK	**I/O**	2	SPI0 Serial Clock
		I2C1_SCL	**I/O**	3	I2C1 clock pin
5	5	PD.9	**I/O**	0	General purpose input/output pin, port D, bit9
		PWM1_ch1	**I/O**	1	PWM1 channel 1 output
		SPI0_MISO0	**I/O**	2	SPI0 Master In Slave Out 0
		UART0_RX	**I**	3	UART0 Receiver Serial In
6	6	PD.8	**I/O**	0	General purpose input/output pin, port D, bit8
		PWM1_ch0	**I/O**	1	PWM1 channel 0 output
		SPI0_SS0	**I/O**	2	SPI0 Slave Select 0
		UART0_TX	**O**	3	UART0 Transmitter Serial Out
7	7	PC.14	**I/O**	0	General purpose input/output pin, port C, bit14
		SPI0_SS0	**I/O**	1	SPI0 Slave Select 0
		I2S0_DI	**I**	2	I2S0 Data input pin
		UART1_RX	**I**	3	UART1 Receiver Serial In

I91535 AQI Pin No.	I91535 H02QI Pin No.	Name	Type	Alt CFG	Description
8	8	PC.13	**I/O**	0	General purpose input/output pin, port C, bit13
		SPI0_MISO0	**I/O**	1	SPI0 Master In Slave Out 0
		I2S0_DO	**O**	2	I2S0 Data output pin
		UART1_TX	**O**	3	UART1 Transmitter Serial Out
9	9	PC.12	**I/O**	0	General purpose input/output pin, port C, bit12
		SPI0_CLK	**I/O**	1	SPI0 Serial Clock
		I2S0_BCLK	**I/O**	2	I2S0 Bit Clock pin
		UART1_nRTS	**O**	3	UART1 Request To Send Output
10	10	PC.0	**I/O**	0	General purpose input/output pin, port C, bit0
		XT1_OUT	**O**	1	External 4~24.576 MHz(high speed) crystal output pin
		I2C0_SCL	**I/O**	3	I2C0 clock pin
11	11	PC.1	**I/O**	0	General purpose input/output pin, port C, bit1
		XT1_IN	**I**	1	External 4~24.576 MHz(high speed) crystal input pin
		I2C0_SDA	**I/O**	3	I2C0 data input/output pin
12	12	PC.11	**I/O**	0	General purpose input/output pin, port C, bit11
		SPI0_MOSI0	**I/O**	1	SPI0 Master Out Slave In 0
		I2S0_LRCK	**I/O**	2	I2S0 left right channel clock pin
		UART1_nCTS	**I**	3	UART1 Clear To Send Input
13	13	PC.10	**I/O**	0	General purpose input/output pin, port C, bit10
		SPI0_MOSI1	**I/O**	1	SPI0 Master Out Slave In 1
		I2S0_MCLK	**O**	2	I2S0 Master Clock Output pin
		MCLKI	**I**	3	External Clock Input
14	14	PC.7	**I/O**	0	General purpose input/output pin, port C, bit7
		I2C0_SDA	**I/O**	1	I2C0 data input/output pin
		SPI1_SS1	**O**	2	SPI1 Slave Select 1
		PWM1_ch1	**O**	3	PWM1 channel 1 output
15	15	PC.6	**I/O**	0	General purpose input/output pin, port C, bit6
		I2C0_SCL	**I/O**	1	I2C0 clock pin
		SPI1_SS0	**I/O**	2	SPI1 Slave Select 0

I91535 AQI Pin No.	I91535 H02QI Pin No.	Name	Type	Alt CFG	Description
		PWM1_ch0	**O**	3	PWM1 channel 0 output
16	16	PC.5	**I/O**	0	General purpose input/output pin, port C, bit5
		UART0_RX	**I**	1	UART0 Receiver Serial In
		I2S0_DI	**I**	2	I2S0 Data input pin
		PWM0_ch3	**O**	3	PWM0 channel 3 output
17	17	PC.4	**I/O**	0	General purpose input/output pin, port C, bit4
		UART0_TX	**O**	1	UART0 Transmitter Serial Out
		I2S0_DO	**O**	2	I2S0 Data output pin
		PWM0_ch2	**O**	3	PWM0 channel 2 output
18	18	PC.3	**I/O**	0	General purpose input/output pin, port C, bit3
		UART0_nRTS	**O**	1	UART0 Request To Send Output
		I2S0_BCLK	**I/O**	2	I2S0 Bit Clock pin
		PWM0_ch1	**O**	3	PWM0 channel 1 output
19	19	PC.2	**I/O**	0	General purpose input/output pin, port C, bit2
		UART0_nCTS	**I**	1	UART0 Clear To Send Input
		I2S0_LRCK	**I/O**	2	I2S0 left right channel clock pin
		PWM0_ch0	**O**	3	PWM0 channel 0 output
20	20	nRESET	**I**		Reset input, low active, internal pull-high
21	21	USB_VBUS	**P**		Power supply from USB or HUB
22	22	USB_D-	**A**		USB differential signal D-
23	23	USB_D+	**A**		USB differential signal D+
24	24	USB_CAP	**P**		USB Internal regulator output decoupling pin. A 1uF cap returning to VSSD must be placed on it.
25	25	PA.15	**I/O**	0	General purpose input/output pin, port A, bit15
		SPI0_SS0	**I/O**	1	SPI0 Slave Select 0
		UART1_RX	**I**	2	UART1 Receiver Serial In
26	26	PA.14	**I/O**	0	General purpose input/output pin, port A, bit14
		SPI0_MISO0	**I/O**	1	SPI0 Master In Slave Out 0
		UART1_TX	**O**	2	UART1 Transmitter Serial Out

I91535 AQI Pin No.	I91535 H02QI Pin No.	Name	Type	Alt CFG	Description
27	27	PA.13	**I/O**	0	General purpose input/output pin, port A, bit13
		SPI0_CLK	**I/O**	1	SPI0 Serial Clock
		UART0_nRTS	**O**	3	UART0 Request To Send Output
28	28	PA.12	**I/O**	0	General purpose input/output pin, port A, bit12
		SPI0_MOSI0	**I/O**	1	SPI0 Master Out Slave In 0
		I2C1_SDA	**I/O**	2	I2C1 data input/output pin
		UART0_nCTS	**I**	3	UART0 Clear To Send Input
	29	VCCPST	**P**		Digital IO Power Supply(Include GPB,GPC,GPD)
29	30	VCCD	**P**		Main Digital Power Supply(Also include GPA IO power)
30	31	LDO_CAP	**P**		Regulator output decoupling pin for core logic. A 1uF cap returning to VSSD must be placed on it.
31	32	VSSD	**P**		Digital ground
32	33	PA.11	**I/O**	0	General purpose input/output pin, port A, bit11
		SPI0_MOSI1	**I/O**	1	SPI0 Master Out Slave In 1
		I2C1_SCL	**I/O**	2	I2C1 clock pin
		UART0_RX	**I**	3	UART0 Receiver Serial In
33	34	PA.10	**I/O**	0	General purpose input/output pin, port A, bit10
		SPI0_MISO1	**I/O**	1	SPI0 Master In Slave Out 1
		MCLKI	**I**	2	External Clock Input
		UART0_TX	**O**	3	UART0 Transmitter Serial Out
34	35	PA.9	**I/O**	0	General purpose input/output pin, port A, bit9
		SPI0_SS1	**O**	1	SPI0 Slave Select 1
		UART0_RX	**I**	2	UART0 Receiver Serial In
		PWM1_ch3	**O**	3	PWM1 channel 3 output
35	36	PA.4	**I/O**	0	General purpose input/output pin, port A, bit4
		I2S0_DI	**I**	1	I2S0 Data input pin
		UART1_RX	**I**	2	UART1 Receiver Serial In
		SPI1_MISO0	**I/O**	3	SPI1 Master In Slave Out 0
36	37	PA.3	**I/O**	0	General purpose input/output pin, port A, bit3

I91535 AQI Pin No.	I91535 H02QI Pin No.	Name	Type	Alt CFG	Description
		I2S0_DO	**O**	1	I2S0 Data output pin
		UART1_TX	**O**	2	UART1 Transmitter Serial Out
		SPI1_CLK	**I/O**	3	SPI1 Serial Clock
37	38	PA.2	**I/O**	0	General purpose input/output pin, port A, bit2
		I2S0_BCLK	**I/O**	1	I2S0 Bit Clock pin
		I2C0_SDA	**I/O**	2	I2C0 data input/output pin
		SPI1_MOSI0	**I/O**	3	SPI1 Master Out Slave In 0
38	39	PA.1	**I/O**	0	General purpose input/output pin, port A, bit1
		I2S0_LRCK	**I/O**	1	I2S0 left right channel clock pin
		I2C0_SCL	**I/O**	2	I2C0 clock pin
		SPI1_MOSI1	**I/O**	3	SPI1 Master Out Slave In 1
39	40	PA.0	**I/O**	0	General purpose input/output pin, port A, bit0
		I2S0_MCLK	**O**	1	I2S0 Master Clock Output pin
		IR	**O**	2	IR carrier generator based on Timer1 interval
		SPI1_MISO1	**I/O**	3	SPI1 Master In Slave Out 1
40	41	MICBIAS	**A**		Microphone Bias Output
41	42	VMIDH	**P**		Mid Rail Reference High, Connect 4.7uF to VSSA
42	43	MICN	**A**		Negative Microphone Input
43	44	MICP	**A**		Positive Microphone Input
44	45	VMIDL	**P**		Mid Rail Reference Low, Connect 4.7uF to VSSA
45	46	VCCA	**P**		Main Analog Circuitry Power Supply
46	47	HPOR	**A**		Headphone Right Channel Output
47	48	HPOL	**A**		Headphone Left Channel Output
48 49(L/F)	49(L/F)	VSSA	**P**		Ground for Analog Circuitry

Note1:

I: Digital Input, O: Digital Output, A: Analog pin, P: Power Pin

Note2:

VCCD & VCCA must be supplied same voltage level, and VCCD must larger or equal to VCCPST.
For I91535AQI, VCCD pin also supply all IO power.

3.4 Pin Alternate Function

Table 3.4-1 Alternate function table of GPIO

GPIO	Power	ALT =1	I/O of ALT = 1	ALT =2	I/O of ALT = 2	ALT =3	I/O of ALT = 3	Special Modes
GPA0*	VCCD	I2S0_MCLK	O	IROUT	O	SPI1_MISO1	I/O	AIN0
GPA1*	VCCD	I2S0_LRCK	I/O	I2C0_SCL	I/O	SPI1_MOSI1	I/O	AIN1
GPA2*	VCCD	I2S0_BCLK	I/O	I2C0_SDA	I/O	SPI1_MOSI0	I/O	AIN2
GPA3*	VCCD	I2S0_DO	O	UART1_TX	O	SPI1_CLK	I/O	AIN3
GPA4*	VCCD	I2S0_DI	I	UART1_RX	I	SPI1_MISO0	I/O	AIN4
GPA5*	VCCD	MCLKI	I	-		SPI1_SS0	I/O	AIN5
GPA6*	VCCD	TMR0	I	CAP1	I	PWM1_ch0	O	AIN6
GPA7*	VCCD	TMR1	I	SPI1_SS1	I/O	PWM1_ch1	O	AIN7
GPA8*	VCCD	TMR2	I	UART0_TX	O	PWM1_ch2	O	AIN8
GPA9*	VCCD	SPI0_SS1	O	UART0_RX	I	PWM1_ch3	O	AIN9
GPA10*	VCCD	SPI0_MISO1	I/O	MCLKI	I	UART0_TX	O	AIN10
GPA11*	VCCD	SPI0_MOSI1	I/O	I2C1_SCL	I/O	UART0_RX	I	AIN11
GPA12*	VCCD	SPI0_MOSI0	I/O	I2C1_SDA	I/O	UART0_nCTS	I	
GPA13*	VCCD	SPI0_CLK	I/O	-		UART0_nRTS	O	
GPA14*	VCCD	SPI0_MISO0	I/O	UART1_TX	O	-		
GPA15* (wake up)	VCCD	SPI0_SS0	I/O	UART1_RX	I	-		
GPB0*	VCCPST	I2C1_SCL	I/O	-		-		
GPB1*	VCCPST	I2C1_SDA	I/O	-		-		
GPC0*	VCCPST	XT1_OUT	O	-		I2C0_SCL	I/O	
GPC1*	VCCPST	XT1_IN	I	-		I2C0_SDA	I/O	
GPC2*	VCCPST	UART0_nCTS	I	I2S0_LRCK	I/O	PWM0_ch0	O	
GPC3*	VCCPST	UART0_nRTS	O	I2S0_BCLK	I/O	PWM0_ch1	O	
GPC4*	VCCPST	UART0_TX	O	I2S0_DO	O	PWM0_ch2	O	
GPC5*	VCCPST	UART0_RX	I	I2S0_DI	I	PWM0_ch3	O	
GPC6*	VCCPST	I2C0_SCL	I/O	SPI1_SS0	I/O	PWM1_ch0	O	
GPC7*	VCCPST	I2C0_SDA	I/O	SPI1_SS1	I/O	PWM1_ch1	O	
GPC8*	VCCPST	SPI0_SS1	O	I2S0_MCLK	O	PWM1_ch2	O	
GPC9*	VCCPST	SPI0_MISO1	I/O	-		PWM1_ch3	O	

GPC10*	VCCPST	SPI0_MOSI1	I/O	I2S0_MCLK	O	MCLKI	I	
GPC11*	VCCPST	SPI0_MOSI0	I/O	I2S0_LRCK	I/O	UART1_nCTS	I	
GPC12*	VCCPST	SPI0_CLK	I/O	I2S0_BCLK	I/O	UART1_nRTS	O	
GPC13*	VCCPST	SPI0_MISO0	I/O	I2S0_DO	O	UART1_TX	O	
GPC14*	VCCPST	SPI0_SS0	I/O	I2S0_DI	I	UART1_RX	I	
GPC15*	VCCPST	-		MCLKI	I	-		
GPD0*	VCCPST	UART1_nCTS	I	PWM0_ch0	O	I2S0_MCLK	O	
GPD1*	VCCPST	UART1_nRTS	O	PWM0_ch1	O	I2S0_LRCK	I/O	
GPD2*	VCCPST	UART1_TX	O	PWM0_ch2	O	I2S0_BCLK	I/O	
GPD3*	VCCPST	UART1_RX	I	PWM0_ch3	O	I2S0_DO	O	
GPD4*	VCCPST	PWM0_ch0	O	CAP0	I	I2S0_DI	I	
GPD5*	VCCPST	PWM0_ch1	O	-		SPI1_MOSI0	I/O	
GPD6*	VCCPST	PWM0_ch2	O	-		SPI1_CLK	I/O	
GPD7*	VCCPST	PWM0_ch3	O	-		SPI1_MISO0	I/O	
GPD8*	VCCPST	PWM1_ch0	O	SPI0_SS0	I/O	UART0_TX	O	
GPD9*	VCCPST	PWM1_ch1	O	SPI0_MISO0	I/O	UART0_RX	I	
GPD10*	VCCPST	PWM1_ch2	O	SPI0_CLK	I/O	I2C1_SCL	I/O	
GPD11*	VCCPST	PWM1_ch3	O	SPI0_MOSI0	I/O	I2C1_SDA	I/O	
GPD12	VCCPST	I2C1_SCL	I/O	SPI0_MOSI1	I/O	ICE_CLK*	I/O	
GPD13	VCCPST	I2C1_SDA	I/O	SPI0_MISO1	I/O	ICE_DAT*	I/O	
GPD14*	VCCPST	I2C0_SCL	I/O	SPI0_SS1	O	SPI1_MISO1	I/O	
GPD15*	VCCPST	I2C0_SDA	I/O	-		SPI1_MOSI1	I/O	

I: Input, O: Output

Note1: AIN0~AIN11 are SARADC inputs, when using SARADC input please select GPIO input mode and disable pull-up option.

Note2: The suffix * is for the reset state of each GPIO

4 Block Diagram

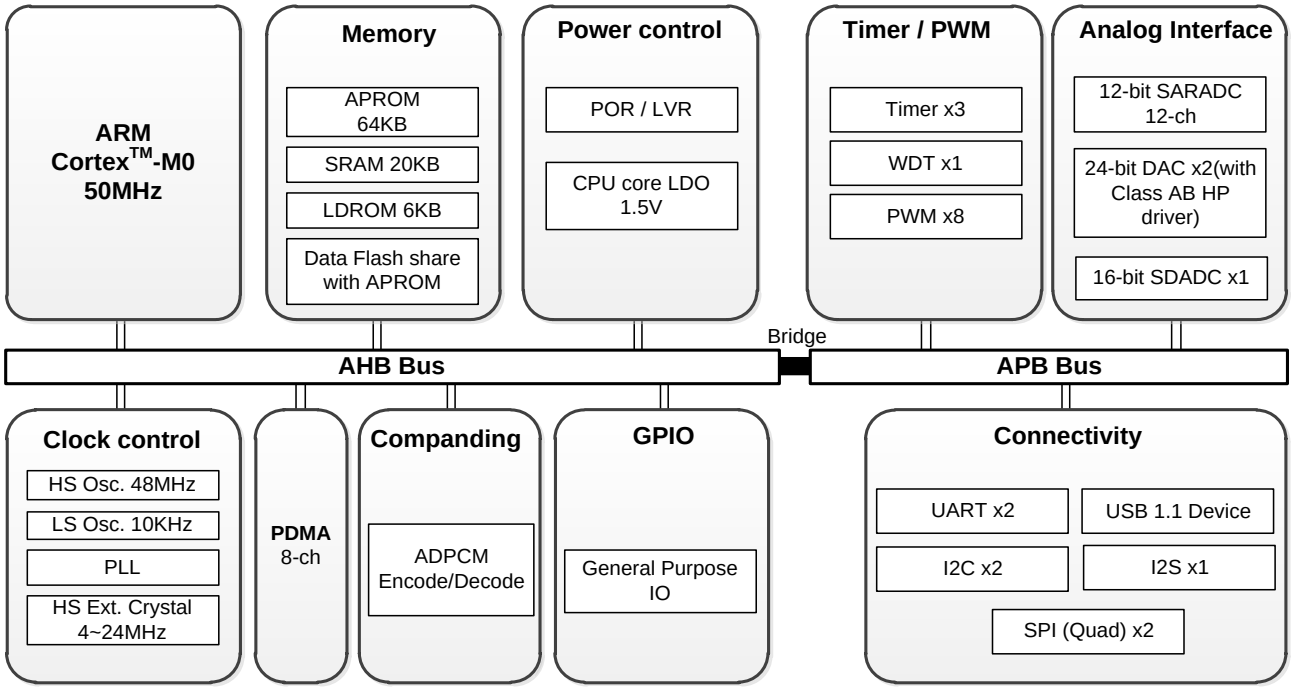


Figure 4-1 Functional Block Diagram 1

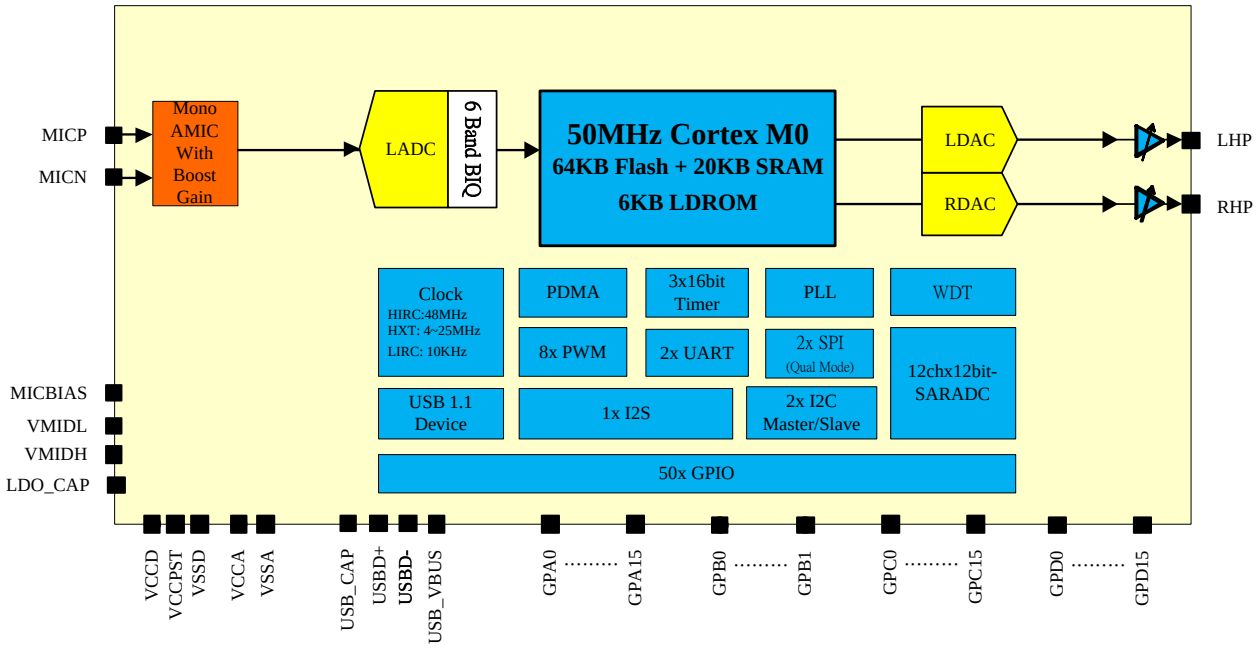


Figure 4-2 Functional Block Diagram 2

5 Functional Description

5.1 ARM® Cortex™-M0 core

The Cortex™-M0 processor is a configurable, multistage, 32-bit RISC processor. It has an AMBA AHB-Lite interface and includes an NVIC component. It also has optional hardware debug functionality. The processor can execute Thumb code and is compatible with other Cortex-M profile processor.

Figure 5.1-1 shows the functional blocks of processor.

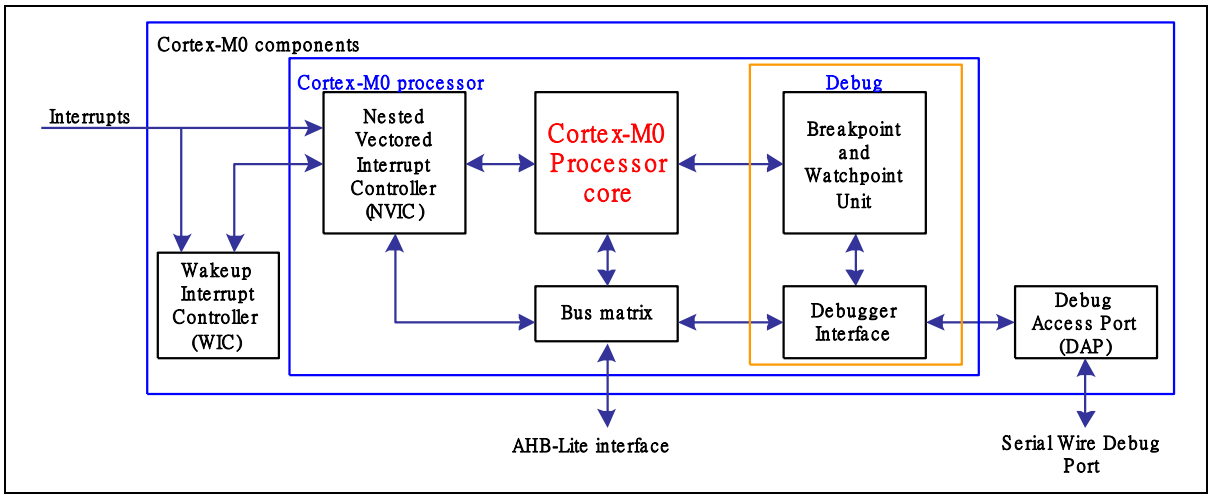


Figure 5.1-1 Functional Block Diagram

The implemented device provides:

- A low gate count processor that features:
  - The ARMv6-M Thumb® instruction set.
  - Thumb-2 technology.
  - ARMv6-M compliant 24-bit SysTick timer.
  - A 32-bit hardware multiplier.
  - The system interface supports little-endian data accesses.
  - The ability to have deterministic, fixed-latency, interrupt handling.
  - Load/store-multiples and multicycle-multiplies that can be abandoned and restarted to facilitate rapid interrupt handling.
  - C Application Binary Interface compliant exception model.
    This is the ARMv6-M, C Application Binary Interface (C-ABI) compliant exception model that enables the use of pure C functions as interrupt handlers.
  - Low power sleep-mode entry using Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or the return from interrupt sleep-on-exit feature.
- NVIC features
  - 32 external interrupt inputs, each with four levels of priority.
  - Dedicated non-Maskable Interrupt (NMI) input.
  - Support for both level-sensitive and pulse-sensitive interrupt lines
  - Wake-up Interrupt Controller (WIC), providing ultra-low power sleep mode support.

- Debug support
  - Four hardware breakpoints.
  - Two watchpoints.
  - Program Counter Sampling Register (PCSR) for non-intrusive code profiling.
  - Single step and vector catch capabilities.
- Bus interfaces
  - Single 32-bit AMBA-3 AHB-Lite system interface that provides simple integration to all system peripherals and memory.
  - Single 32-bit slave port that supports the DAP.

5.2 System Manager

5.2.1 Overview

The following functions are included in system manager section

- System Memory Map
- System management registers for chip and module functional reset and multi-function pin control
- Chip miscellaneous Control Register
- Combined peripheral interrupt source identify

5.2.2 System Memory Map

ISD91500 provides a 4G-byte address space for programmers. The memory locations assigned to each on-chip modules are shown in Table 5.2-1

Table 5.2-1 is Address Space Assignments for On-Chip Modules. The detailed register and memory addressing and programming will be described in the following sections for individual on-chip modules. ISD91500 series only supports little-endian data format.

Table 5.2-1 Address Space Assignments for On-Chip Modules

Address Space	Token	Modules
Flash & SRAM Memory Space		
0x0000_0000 – 0x0000_FFFF	FLASH_BA	Flash Memory Space (64KB)
0x2000_0000 – 0x2000_4FFF	SRAM_BA	SRAM Memory Space (20KB)
AHB Modules Space (0x5000_0000 – 0x5000_FFFF)		
0x5000_0000 – 0x5000_01FF	SYS_BA	System Global Control Registers
0x5000_0200 – 0x5000_02FF	CLK_BA	Clock Control Registers
0x5000_0300 – 0x5000_03FF	INT_BA	Interrupt Multiplexer Control Registers
0x5000_4000 – 0x5000_7FFF	GPIO_BA	GPIO Control Registers
0x5000_9000 – 0x5000_BFFF	PDMA_BA	PDMA Controller Registers

0x5000_C000 – 0x5000_CFFF	FMC_BA	Flash Memory Control Registers
0x5000_E000 – 0x5000_EFFF	CPD_BA	Companding Control Registers
APB Modules Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_40FF	WDT_BA	Watch-Dog Timer Control Registers
0x4001_0000 – 0x4001_0FFF	TMR_BA	Timer0/Timer1/Timer2 Control Registers
0x4002_0000 – 0x4002_0FFF	I2C0_BA	I2C0 Control Registers
0x4002_1000 – 0x4002_1FFF	I2C1_BA	I2C1 Control Registers
0x4003_0000 – 0x4003_0FFF	SPI0_BA	SPI0 Serial Interface Control Registers
0x4003_1000 – 0x4003_1FFF	SPI1_BA	SPI1 Serial Interface Control Registers
0x4004_0000 – 0x4004_0FFF	PWM0_BA	PWM0 Control Registers
0x4005_0000 – 0x4005_0FFF	PWM1_BA	PWM1 Control Registers
0x4006_0000 – 0x4006_0FFF	UART0_BA	UART0 Control Registers
0x4006_1000 – 0x4006_1FFF	UART1_BA	UART1 Control Registers
0x4007_0000 – 0x4007_0FFF	DAC_BA	DAC Control Registers
0x4008_0000 – 0x4008_3FFF	ANA_BA	Analog Block Control Registers
0x4009_0000 – 0x4009_0FFF	I2S0_BA	I2S0 Control Registers
0x400B_1000 – 0x400B_1FFF	BIQ_BA	BIQ Control Registers
0x400C_0000 – 0x400C_0FFF	USBD_BA	USB Device Control Registers
0x400D_0000 – 0x400D_FFFF	SDADC_BA	Analog-Digital-Converter(SDADC) Control Registers
0x400E_0000 – 0x400E_0FFF	SARADC_BA	SARADC Control Registers
System Control Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SYSTICK_BA	System Timer Control Registers
0xE000_E100 – 0xE000_ECFF	SCS_BA	NVIC Control Registers
0xE000_ED00 – 0xE000_ED8F	SYSINFO_BA	System Control Registers

5.2.3 System Manager Control Registers

Register	Offset	R/W	Description	Reset Value
SYS Base Address: **SYS_BA = 0x5000_0000**				
SYS_PDID	SYS_BA+0x00	R	Product Identifier Register	0xXXXX_XXXX
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Source Register	0x0000_0XXX
SYS_IPRST0	SYS_BA+0x08	R/W	IP Reset Control Resister0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	IP Reset Control Resister1	0x0000_0000
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-Out Detector Control Register	0x000X_00XX
SYS_GPA_MFP	SYS_BA+0x20	R/W	GPIO PA Multiple Alternate Functions and Input Type Control Register	0x0000_0000
SYS_GPB_MFP	SYS_BA+0x24	R/W	GPIO PB Multiple Alternate Functions and Input Type Control Register	0x0000_0000
SYS_GPC_MFP	SYS_BA+0x28	R/W	GPIO PC Multiple Alternate Functions and Input Type Control Register	0x0000_0000
SYS_GPD_MFP	SYS_BA+0x2C	R/W	GPIO PD Multiple Alternate Functions and Input Type Control Register	0x0F00_0000
SYS_GPIO_INTP	SYS_BA+0x40	R/W	GPIO input type and slew rate Control	0xFFFF_03FF
SYS_GPA_PULL	SYS_BA+0x44	R/W	PA.15 ~ PA.0 Pull Resistance Control Register	0x0000_0000
SYS_GPA_HR	SYS_BA+0x48	R/W	PA.15 ~ PA.0 Pull Resistance Select Control Register	0x0000_FFFF
SYS_GPA_IEN	SYS_BA+0x4C	R/W	PA.15 ~ PA.0 Digital and Analog Input Buffer Control Register	0x0000_0000
SYS_GPB_PULL	SYS_BA+0x54	R/W	PB.1 ~ PB.0 Pull Resistance Control Register	0x0000_0000
SYS_GPB_HR	SYS_BA+0x58	R/W	PB.1 ~ PB.0 Pull Resistance Select Control Register	0x0000_0003
SYS_GPB_IEN	SYS_BA+0x5C	R/W	PB.1 ~ PB.0 Digital Input Buffer Control Register	0x0000_0000
SYS_GPC_PULL	SYS_BA+0x64	R/W	PC.15 ~ PC.0 Pull Resistance Control Register	0x0000_0000

SYS_GPC_HR	SYS_BA+0x68	R/W	PC.15 ~ PC.0 Pull Resistance Select Control Register	0x0000_FFFF
SYS_GPC_IEN	SYS_BA+0x6C	R/W	PC.15 ~ PC.0 Digital Input Buffer Control Register	0x0000_0000
SYS_GPD_PULL	SYS_BA+0x74	R/W	PD.15 ~ PD.0 Pull Resistance Control Register	0x0000_0000
SYS_GPD_HR	SYS_BA+0x78	R/W	PD.15 ~ PD.0 Pull Resistance Select Control Register	0x0000_FFFF
SYS_GPD_IEN	SYS_BA+0x7C	R/W	PD.15 ~ PD.0 Digital Input Buffer Control Register	0x0000_0000
SYS_IMGMAP3	SYS_BA+0xF0	R	MAP3 Data Image Register	0xXXXX_XXXX
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control Register	0x0000_0000
SYS_OSCTRIM	SYS_BA+0x110	R/W	Internal oscillator trim register	0xXXXX_XXXX
SYS_OSC10K	SYS_BA+0x114	R/W	10KHz Oscillator and Bias trim register	0xXXXX_XXXX
SYS_OSC_TRIMn n=0,1,2	SYS_BA+0x118+0x04 *n	R/W	Oscillator Frequency Adjustment control register	0xXXXX_XXXX
SYS_IRCTCTL	SYS_BA+0x130	R/W	HIRC Trim Control Register	0x0000_0000
SYS_IRCTIEN	SYS_BA+0x134	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000
SYS_IRCTISTS	SYS_BA+0x138	R/W	HIRC Trim Interrupt Status Register	0x0000_0000
SYS_IRCTCKRF	SYS_BA+0x13C	R/W	HIRC Trim Clock Reference Frequency Register	0x0000_0020
SYS_BGAPTRIM	SYS_BA+0x140	R/W	Bandgap Trim Control Register	0x0000_000X
SYS_UCID[n] N=0,1,2,3	SYS_BA+0x150+0x04 *n	R	Specified ID register for library and customized feature checking	0xXXXX_XXXX

Product Identifier Register (SYS_PDID)

This register provides specific read-only information for software to identify this chip.

Register	Offset	R/W	Description	Reset Value
SYS_PDID	SYS_BA+0x00	R	Product Identifier Register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
IMG2[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
IMG2[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IMG2[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IMG2[7:0]							

Bits	Description	
[31:0]	**IMG2**	**Product Identifier** Data in MAP2 of information block are copied to this register after power on. MAP2 is used to store part number defined by Nuvoton.

System Reset Source Register (SYS_RSTSTS)

This register provides specific information for software to identify this chip's reset source from last operation.

Register	Offset	R/W	Description	Reset Value
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Source Register	0x0000_0XXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					PORWK	TIMWK	PINWK
7	6	5	4	3	2	1	0
Reserved	PMURSTF	Reserved	BODRF	LVRF	WDTRF	PINRF	PORF

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**PORWK**	**Wakeup from DPD From POR** The device was woken from Deep Power Down by a Power On Reset. 0= No wakeup from POR. 1= The device was issued a wakeup from DPD by a POR. **Note:** Clear by write SYS_RSTSTS[8] =1'b
[9]	**TIMWK**	**Wakeup from DPD From TIMER** The device was woken from Deep Power Down by count of 10 KHz timer. 0= No wakeup from TIMER. 1= The device was issued a wakeup from DPD by a TIMER event. **Note:** Clear by write SYS_RSTSTS[8] =1'b
[8]	**PINWK**	**Wakeup from DPD From PIN** The device was woken from Deep Power Down by a low transition on the RESETn pin. 0= No wakeup from RESETn pin. 1= The device was issued a wakeup from DPD by a RESETn pin trasition. **Note:** Write 1 to this register to clear all wakeup flags.
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[6]	**PMURSTF**	**Reset Source From PMU** The PMURSTF flag is set by the reset signal from the PMU module to indicate the previous reset source. 0= No reset from PMU. 1= The PMU has issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**BODRF**	**BOD Reset Flag** The BOD reset flag is set by the “Reset Signal” from the Brown Out Reset Controller to indicate the previous reset source. 0 = No reset from BOD. 1 = BOD controller had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[3]	**LVRF**	**LVR Reset Flag** The LVR reset flag is set by the “Reset Signal” from the Low Voltage Reset Controller to indicate the previous reset source. 0 = No reset from LVR. 1 = LVR controller had issued the reset signal to reset the system. **Note1:** Write 1 to clear this bit to 0. **Note2:** If power rising reach 1.6V under 20us when fast power on, the LVRF will not happen.
[2]	**WDTRF**	**Reset Source From WDG** The WDTRF flag is set if pervious reset source originates from the Watch-Dog module. 0= No reset from Watch-Dog. 1= The Watch-Dog module issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[1]	**PINRF**	**nRESET Pin Reset Flag** The nRESET pin reset flag is set by the “Reset Signal” from the nRESET Pin to indicate the previous reset source. 0 = No reset from nRESET pin. 1 = Pin nRESET had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[0]	**PORF**	**POR Reset Flag** The POR reset flag is set by the “Reset Signal” from the Power-on Reset (POR) Controller to indicate the previous reset source. 0 = No reset from POR. 1 = Power-on Reset (POR) Controller had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.

IP Reset Control Register0(SYS_IPRST0)

Register	Offset	R/W	Description	Reset Value
SYS_IPRST0	SYS_BA+0x08	R/W	IP Reset Control Resister0	0x0000_0000

To program these bits needs an open lock sequence, write “59h”, “16h”, “88h” to register SYS_REGLCTL to unlock these bits. Refer to the register SYS_REGLCTL at address SYS_BA+0x100.

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

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**CPURST**	**CPU Kernel One Shot Reset** Setting this bit will reset the CPU kernel and Flash Memory Controller (FMC), this bit will automatically return to “0” after the 2 clock cycles 0 = Normal. 1 = Reset CPU.
[0]	**CHIPRST**	**CHIP One Shot Reset** Set this bit will reset the whole chip, this bit will automatically return to “0” after 2 clock cycles. CHIPRST is same as POR reset, all the chip modules are reset and the chip configuration settings from flash are reloaded. 0 = Normal. 1 = Reset CHIP.

IP Reset Control Register1 (SYS_IPRST1)

Setting these bits “1” will generate an asynchronous reset signal to the corresponding peripheral block. The user needs to set bit to “0” to release block from the reset state.

Register	Offset	R/W	Description	Reset Value
SYS_IPRST1	SYS_BA+0x0C	R/W	IP Reset Control Resister1	0x0000_0000

31	30	29	28	27	26	25	24
ANARST	SDADCRST	DACRST	SARADCRST	Reserved			USBRST
23	22	21	20	19	18	17	16
Reserved		PWM1RST	PWM0RST	Reserved	BIQRST	UART1RST	UART0RST
15	14	13	12	11	10	9	8
Reserved		I2S0RST	SPI0RST	SPI1RST	Reserved	I2C1RST	I2C0RST
7	6	5	4	3	2	1	0
PDMARST	CPDRST	Reserved	TMR2RST	TMR1RST	TMR0RST	GPIORST	Reserved

Bits	Description	
[31]	**ANARST**	**Analog Block Controller Reset** 0 = Normal Operation. 1 = Reset.
[30]	**SDADCRST**	**SDADC Controller Reset** 0 = Normal Operation. 1 = Reset.
[29]	**DACRST**	**DAC Controller Reset** 0 = Normal Operation. 1 = Reset.
[28]	**SARADCRST**	**SARADC Controller Reset** 0 = Normal Operation. 1 = Reset.
[27:25]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**USBRST**	**USB Controller Reset** 0 = Normal Operation. 1 = Reset.
[23:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[21]	**PWM1RST**	**PWM1 Controller Reset** 0 = Normal Operation. 1 = Reset.
[20]	**PWM0RST**	**PWM0 Controller Reset** 0 = Normal Operation. 1 = Reset.
[19]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[18]	**BIQRST**	**BIQ Controller Reset** 0 = Normal Operation. 1 = Reset.
[17]	**UART1RST**	**UART1 Controller Reset** 0 = Normal Operation. 1 = Reset.
[16]	**UART0RST**	**UART0 Controller Reset** 0 = Normal Operation. 1 = Reset.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**I2S0RST**	**I2S0 Controller Reset** 0 = Normal Operation. 1 = Reset.
[12]	**SPI0RST**	**SPI0 Controller Reset** 0 = Normal Operation. 1 = Reset.
[11]	**SPI1RST**	**SPI1 Controller Reset** 0 = Normal Operation. 1 = Reset.
[10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**I2C1RST**	**I2C1 Controller Reset** 0 = Normal operation. 1 = Reset.
[8]	**I2C0RST**	**I2C0 Controller Reset** 0 = Normal operation. 1 = Reset.

[7]	**PDMARST**	**PDMA Controller Reset** 0 = Normal operation. 1 = Reset.
[6]	**CPDRST**	**Companding Controller Reset** 0 = Normal operation. 1 = Reset.
[5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**TMR2RST**	**Timer2 Controller Reset** 0 = Normal operation. 1 = Reset.
[3]	**TMR1RST**	**Timer1 Controller Reset** 0 = Normal Operation. 1 = Reset.
[2]	**TMR0RST**	**Timer0 Controller Reset** 0 = Normal Operation. 1 = Reset.
[1]	**GPIORST**	**GPIO Controller Reset** 0 = Normal operation. 1 = Reset.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Brown-Out Detector Control Register (SYS_BODCTL)

Register	Offset	R/W	Description	Reset Value
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-Out Detector Control Register	0x000X_00XX

Par of the SYS_BODCTL control register values are initiated by flash configuration. After the power on initialization, these bits are protected by the lock circuit. To program these bits an open lock sequence must be performed by writing “59h”, “16h”, “88h” to address 0x5000_0100 to un-lock these bits. Refer to the register SYS_REGLCTL at address SYS_BA+0x100.

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**LVRFILTER**		**LVREN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**BODINT**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BODOUT	**Reserved**	**BODLVL**				**BODRSTEN**	**BODEN**

Bits	Description	
[31:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[18:17]	**LVRFILTER**	00 = LVR output will be filtered by 1 HCLK. 01 = LVR output will be filtered by 2 HCLK 10 = LVR output will be filtered by 8 HCLK 11 = LVR output will be filtered by 15 HCLK Default value is 00.
[16]	**LVREN**	**Low Voltage Reset (LVR) Enable (Initialized & Protected Bit)** The LVR function resets the chip when the input power voltage is lower than LVR trip point. Default value is set by flash controller as inverse of CLVR config 0[27]. 0 = Disable LVR function. 1 = Enable LVR function.
[15:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**BODINT**	**Brown-Out Dectector Interrupt** 1 = indicates BOD_INT is active. Write 1 to clear.

<table>
<tr><td>[7]</td><td>BODOUT</td><td>Brown-Out Detector Output State
0 = Brown-out Detector status output is 0, the detected voltage is higher than BODLVL setting.
1 = Brown-out Detector status output is 1, the detected voltage is lower than BODLVL setting.</td></tr>
<tr><td>[6]</td><td>Reserved</td><td>Reserved. Any values read should be ignored. When writing to this field always write with reset value.</td></tr>
<tr><td>[5:2]</td><td>BODLVL</td><td>Brown-Out Detector Threshold Voltage Selection (Initialized & Protected Bit)
The default value is set by flash controller user configuration CBOV bit (config0 [25:22]).
<table>
<tr><th>BODLVL[3:0]</th><th>Brown-out voltage</th><th>BODLVL[3:0]</th><th>Brown-out voltage</th></tr>
<tr><td>0111</td><td>2.8V</td><td>1111</td><td>3.4V</td></tr>
<tr><td>0110</td><td>2.6V</td><td>1110</td><td>3.4V</td></tr>
<tr><td>0101</td><td>2.4V</td><td>1101</td><td>3.4V</td></tr>
<tr><td>0100</td><td>2.2V</td><td>1100</td><td>3.4V</td></tr>
<tr><td>0011</td><td>2.1V</td><td>1011</td><td>3.4V</td></tr>
<tr><td>0010</td><td>2.0V</td><td>1010</td><td>3.4V</td></tr>
<tr><td>0001</td><td>1.9V</td><td>1001</td><td>3.1V</td></tr>
<tr><td>0000</td><td>1.8V</td><td>1000</td><td>3.0V</td></tr>
</table>
Note: When set BODLVL[3:0] > 1010’b , BODLVL[3:0] value is fixed to 1010’b .</td></tr>
<tr><td>[1]</td><td>BODRSTEN</td><td>Brown-Out Detector Reset or Interrupt Bit (Initialized & Protected Bit)
The default value is set by flash controller as inverse of user configuration CBORST bit (config0 [21]).
0 = Brown-Out Detector generate an interrupt
1 = Brown-Out Detector will reset chip
When the BOD is enabled and the interrupt is asserted, the interrupt will be kept till the BOD is disabled. The interrupt for CPU can be blocked either by disabling the interrupt in the NVIC or by disabling the interrupt source by disabling the BOD. BOD can then be re-enabled as required.</td></tr>
<tr><td>[0]</td><td>BODEN</td><td>Brown-Out Detector Threshold Voltage Selection Extension (Initialized & Protected Bit)
The default value is set by flash controller as inverse of user configuration CBODEN bit (config0 [20]).
0 = Brown-Out Detector function is disabled
1 = Brown-Out Detector function enabled</td></tr>
</table>

GPIO PA Multiple Alternate Function and Input Type Control Register (SYS_GPA_MFP)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP	SYS_BA+0x20	R/W	GPIO PA Multiple Alternate Functions and Input Type Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PA15MFP		PA14MFP		PA13MFP		PA12MFP	
23	22	21	20	19	18	17	16
PA11MFP		PA10MFP		PA9MFP		PA8MFP	
15	14	13	12	11	10	9	8
PA7MFP		PA6MFP		PA5MFP		PA4MFP	
7	6	5	4	3	2	1	0
PA3MFP		PA2MFP		PA1MFP		PA0MFP	

Bits	Description
[31:30]	**PA15MFP** **PA.15 Multi-function Pin Selection** Pin Name: PA15MFP PA.15: 00 = PA.15; 01 = SPI0_SS0; 10 = UART1_RX; 11 =
[29:28]	**PA14MFP** **PA.14 Multi-function Pin Selection** Pin Name: PA14MFP PA.14: 00 = PA.14; 01 = SPI0_MISO0; 10 = UART1_TX; 11 =

<table>
<tr><td>[27:26]</td><td>PA13MFP</td><td>PA.13 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA13MFP</th></tr>
<tr><td rowspan="2">PA.13</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.13</td><td>SPI0_CLK</td><td></td><td>UART0_nRTS</td></tr>
</table></td></tr>
<tr><td>[25:24]</td><td>PA12MFP</td><td>PA.12 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA12MFP</th></tr>
<tr><td rowspan="2">PA.12</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.12</td><td>SPI0_MOSI0</td><td>I2C1_SDA</td><td>UART0_nCTS</td></tr>
</table></td></tr>
<tr><td>[23:22]</td><td>PA11MFP</td><td>PA.11 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA11MFP</th></tr>
<tr><td rowspan="2">PA.11</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.11</td><td>SPI0_MOSI1</td><td>I2C1_SCL</td><td>UART0_RX</td></tr>
</table></td></tr>
<tr><td>[21:20]</td><td>PA10MFP</td><td>PA.10 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA10MFP</th></tr>
<tr><td rowspan="2">PA.10</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.10</td><td>SPI0_MISO1</td><td>MCLKI</td><td>UART0_TX</td></tr>
</table></td></tr>
<tr><td>[19:18]</td><td>PA9MFP</td><td>PA.9 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA9MFP</th></tr>
<tr><td rowspan="2">PA.9</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.9</td><td>SPI0_SS1</td><td>UART0_RX</td><td>PWM13</td></tr>
</table></td></tr>
</table>

<table>
<tr>
<td>[17:16]</td>
<td>PA8MFP</td>
<td>PA.8 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA8MFP</th></tr>
<tr><td rowspan="2">PA.8</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.8</td><td>TM2</td><td>UART0_TX</td><td>PWM12</td></tr>
</table></td>
</tr>
<tr>
<td>[15:14]</td>
<td>PA7MFP</td>
<td>PA.7 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA7MFP</th></tr>
<tr><td rowspan="2">PA.7</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.7</td><td>TM1</td><td>SPI1_SS1</td><td>PWM11</td></tr>
</table></td>
</tr>
<tr>
<td>[13:12]</td>
<td>PA6MFP</td>
<td>PA.6 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA6MFP</th></tr>
<tr><td rowspan="2">PA.6</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.6</td><td>TM0</td><td>CAP1</td><td>PWM10</td></tr>
</table></td>
</tr>
<tr>
<td>[11:10]</td>
<td>PA5MFP</td>
<td>PA.5 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA5MFP</th></tr>
<tr><td rowspan="2">PA.5</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.5</td><td>MCLKI</td><td></td><td>SPI1_SS0</td></tr>
</table></td>
</tr>
<tr>
<td>[9:8]</td>
<td>PA4MFP</td>
<td>PA.4 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA4MFP</th></tr>
<tr><td rowspan="2">PA.4</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.4</td><td>I2S0_DI</td><td>UART1_RX</td><td>SPI1_MISO0</td></tr>
</table></td>
</tr>
</table>

<table>
<tr><td>[7:6]</td><td>PA3MFP</td><td>PA.3 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA3MFP</th></tr>
<tr><td rowspan="2">PA.3</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.3</td><td>I2S0_DO</td><td>UART1_TX</td><td>SPI1_CLK</td></tr>
</table></td></tr>
<tr><td>[5:4]</td><td>PA2MFP</td><td>PA.2 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA2MFP</th></tr>
<tr><td rowspan="2">PA.2</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.2</td><td>I2S0_BCLK</td><td>I2C0_SDA</td><td>SPI1_MOSI0</td></tr>
</table></td></tr>
<tr><td>[3:2]</td><td>PA1MFP</td><td>PA.1 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA1MFP</th></tr>
<tr><td rowspan="2">PA.1</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.1</td><td>I2S0_LRCK</td><td>I2C0_SCL</td><td>SPI1_MOSI1</td></tr>
</table></td></tr>
<tr><td>[1:0]</td><td>PA0MFP</td><td>PA.0 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PA0MFP</th></tr>
<tr><td rowspan="2">PA.0</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.0</td><td>I2S0_MCLK</td><td>IROUT</td><td>SPI1_MISO1</td></tr>
</table></td></tr>
</table>

GPIO PB Multiple Alternate Function and Input Type Control Register (SYS_GPB_MFP)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFP	SYS_BA+0x24	R/W	GPIO PB Multiple Alternate Functions and Input Type Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PB1MFP**		**PB0MFP**	

Bits	Description	
[31:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3:2]	**PB1MFP**	**PB.1 Multi-function Pin Selection**

Pin Name	PB1MFP			
PB.1	00	01	10	11
	PB.1	I2C1_SDA		

Bits	Description	
[1:0]	**PB0MFP**	**PB.0 Multi-function Pin Selection**

Pin Name	PB0MFP			
PB.0	00	01	10	11
	PB.0	I2C1_SCL		

GPIO PC Multiple Alternate Function and Input Type Control Register (SYS_GPC_MFP)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFP	SYS_BA+0x28	R/W	GPIO PC Multiple Alternate Functions and Input Type Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PC15MFP		PC14MFP		PC13MFP		PC12MFP	
23	22	21	20	19	18	17	16
PC11MFP		PC10MFP		PC9MFP		PC8MFP	
15	14	13	12	11	10	9	8
PC7MFP		PC6MFP		PC5MFP		PC4MFP	
7	6	5	4	3	2	1	0
PC3MFP		PC2MFP		PC1MFP		PC0MFP	

Bits	Description
[31:30]	**PC15MFP**

PC.15 Multi-function Pin Selection

Pin Name	PC15MFP			
PC.15	00	01	10	11
	PC.15		MCLKI	

Bits	Description
[29:28]	**PC14MFP**

PC.14 Multi-function Pin Selection

Pin Name	PC14MFP			
PC.14	00	01	10	11
	PC.14	SPI0_SS0	I2S0_DI	UART1_RX

<table>
<tr><td>[27:26]</td><td>PC13MFP</td><td>PC.13 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC13MFP</th></tr>
<tr><td rowspan="2">PC.13</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.13</td><td>SPI0_MISO0</td><td>I2S0_DO</td><td>UART1_TX</td></tr>
</table></td></tr>
<tr><td>[25:24]</td><td>PC12MFP</td><td>PC.12 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC12MFP</th></tr>
<tr><td rowspan="2">PC.12</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.12</td><td>SPI0_CLK</td><td>I2S0_BCLK</td><td>UART1_nRTS</td></tr>
</table></td></tr>
<tr><td>[23:22]</td><td>PC11MFP</td><td>PC.11 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC11MFP</th></tr>
<tr><td rowspan="2">PC.11</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.11</td><td>SPI0_MOSI0</td><td>I2S0_LRCK</td><td>UART1_nCTS</td></tr>
</table></td></tr>
<tr><td>[21:20]</td><td>PC10MFP</td><td>PC.10 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC10MFP</th></tr>
<tr><td rowspan="2">PC.10</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.10</td><td>SPI0_MOSI1</td><td>I2S0_MCLK</td><td>MCLKI</td></tr>
</table></td></tr>
<tr><td>[19:18]</td><td>PC9MFP</td><td>PC.9 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC9MFP</th></tr>
<tr><td rowspan="2">PC.9</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.9</td><td>SPI0_MISO1</td><td></td><td>PWM13</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>[17:16]</td><td>PC8MFP</td><td>PC.8 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC8MFP</th></tr>
<tr><td rowspan="2">PC.8</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.8</td><td>SPI0_SS1</td><td>I2S0_MCLK</td><td>PWM12</td></tr>
</table></td></tr>
<tr><td>[15:14]</td><td>PC7MFP</td><td>PC.7 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC7MFP</th></tr>
<tr><td rowspan="2">PC.7</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.7</td><td>I2C0_SDA</td><td>SPI1_SS1</td><td>PWM11</td></tr>
</table></td></tr>
<tr><td>[13:12]</td><td>PC6MFP</td><td>PC.6 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC6MFP</th></tr>
<tr><td rowspan="2">PC.6</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.6</td><td>I2C0_SCL</td><td>SPI1_SS0</td><td>PWM10</td></tr>
</table></td></tr>
<tr><td>[11:10]</td><td>PC5MFP</td><td>PC.5 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC5MFP</th></tr>
<tr><td rowspan="2">PC.5</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.5</td><td>UART0_RX</td><td>I2S0_DI</td><td>PWM03</td></tr>
</table></td></tr>
<tr><td>[9:8]</td><td>PC4MFP</td><td>PC.4 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC4MFP</th></tr>
<tr><td rowspan="2">PC.4</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.4</td><td>UART0_TX</td><td>I2S0_DO</td><td>PWM02</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>[7:6]</td><td>PC3MFP</td><td>PC.3 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC3MFP</th></tr>
<tr><td rowspan="2">PC.3</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.3</td><td>UART0_nRTS</td><td>I2S0_BCLK</td><td>PWM01</td></tr>
</table></td></tr>
<tr><td>[5:4]</td><td>PC2MFP</td><td>PC.2 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC2MFP</th></tr>
<tr><td rowspan="2">PC.2</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.2</td><td>UART0_nCTS</td><td>I2S0_LRCK</td><td>PWM00</td></tr>
</table></td></tr>
<tr><td>[3:2]</td><td>PC1MFP</td><td>PC.1 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC1MFP</th></tr>
<tr><td rowspan="2">PC.1</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.1</td><td>XT1_IN</td><td></td><td>I2C0_SDA</td></tr>
</table></td></tr>
<tr><td>[1:0]</td><td>PC0MFP</td><td>PC.0 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PC0MFP</th></tr>
<tr><td rowspan="2">PC.0</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.0</td><td>XT1_OUT</td><td></td><td>I2C0_SCL</td></tr>
</table></td></tr>
</table>

GPIO PD Multiple Alternate Function and Input Type Control Register (SYS_GPD_MFP)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFP	SYS_BA+0x2C	R/W	GPIO PD Multiple Alternate Functions and Input Type Control Register	0x0F00_0000

31	30	29	28	27	26	25	24
PD15MFP		**PD14MFP**		**PD13MFP**		**PD12MFP**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PD11MFP		**PD10MFP**		**PD9MFP**		**PD8MFP**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PD7MFP		**PD6MFP**		**PD5MFP**		**PD4MFP**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD3MFP		**PD2MFP**		**PD1MFP**		**PD0MFP**	

Bits	Description	
[31:30]	**PD15MFP**	**PD.15 Multi-function Pin Selection** **Pin Name** / **PC15MFP** **PD.15**: 00 = PD.15; 01 = I2C0_SDA; 10 = ; 11 = SPI1_MOSI1
[29:28]	**PD14MFP**	**PD.14 Multi-function Pin Selection** **Pin Name** / **PD14MFP** **PD.14**: 00 = PD.14; 01 = I2C0_SCL; 10 = SPI0_SS1; 11 = SPI1_MISO1

<table>
<tr>
<td>[27:26]</td>
<td>PD13MFP</td>
<td>PD.13 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD13MFP</th></tr>
<tr><td rowspan="2">PD.13</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.13</td><td>I2C1_SDA</td><td>SPI0_MISO1</td><td>ICE_DAT</td></tr>
</table></td>
</tr>
<tr>
<td>[25:24]</td>
<td>PD12MFP</td>
<td>PD.12 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD12MFP</th></tr>
<tr><td rowspan="2">PD.12</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.12</td><td>I2C1_SCL</td><td>SPI0_MOSI1</td><td>ICE_CLK</td></tr>
</table></td>
</tr>
<tr>
<td>[23:22]</td>
<td>PD11MFP</td>
<td>PD.11 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD11MFP</th></tr>
<tr><td rowspan="2">PD.11</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.11</td><td>PWM13</td><td>SPI0_MOSI0</td><td>I2C1_SDA</td></tr>
</table></td>
</tr>
<tr>
<td>[21:20]</td>
<td>PD10MFP</td>
<td>PD.10 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD10MFP</th></tr>
<tr><td rowspan="2">PD.10</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.10</td><td>PWM12</td><td>SPI0_CLK</td><td>I2C1_SCL</td></tr>
</table></td>
</tr>
<tr>
<td>[19:18]</td>
<td>PD9MFP</td>
<td>PD.9 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD9MFP</th></tr>
<tr><td rowspan="2">PD.9</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.9</td><td>PWM11</td><td>SPI0_MISO0</td><td>UART0_RX</td></tr>
</table></td>
</tr>
</table>

<table>
<tr>
<td>[17:16]</td>
<td>PD8MFP</td>
<td>PD.8 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD8MFP</th></tr>
<tr><td rowspan="2">PD.8</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.8</td><td>PWM10</td><td>SPI0_SS0</td><td>UART0_TX</td></tr>
</table></td>
</tr>
<tr>
<td>[15:14]</td>
<td>PD7MFP</td>
<td>PD.7 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD7MFP</th></tr>
<tr><td rowspan="2">PD.7</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.7</td><td>PWM03</td><td></td><td>SPI1_MISO0</td></tr>
</table></td>
</tr>
<tr>
<td>[13:12]</td>
<td>PD6MFP</td>
<td>PD.6 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD6MFP</th></tr>
<tr><td rowspan="2">PD.6</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.6</td><td>PWM02</td><td></td><td>SPI1_CLK</td></tr>
</table></td>
</tr>
<tr>
<td>[11:10]</td>
<td>PD5MFP</td>
<td>PD.5 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD5MFP</th></tr>
<tr><td rowspan="2">PD.5</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.5</td><td>PWM01</td><td></td><td>SPI1_MOSI0</td></tr>
</table></td>
</tr>
<tr>
<td>[9:8]</td>
<td>PD4MFP</td>
<td>PC.4 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD4MFP</th></tr>
<tr><td rowspan="2">PD.4</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.4</td><td>PWM00</td><td>CAP0</td><td>I2S0_DI</td></tr>
</table></td>
</tr>
</table>

<table>
<tr><td>[7:6]</td><td>PD3MFP</td><td>PC.3 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD3MFP</th></tr>
<tr><td rowspan="2">PD.3</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.3</td><td>UART1_RX</td><td>PWM03</td><td>I2S0_DO</td></tr>
</table></td></tr>
<tr><td>[5:4]</td><td>PD2MFP</td><td>PD.2 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD2MFP</th></tr>
<tr><td rowspan="2">PD.2</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PC.2</td><td>UART1_TX</td><td>PWM02</td><td>I2S0_BCLK</td></tr>
</table></td></tr>
<tr><td>[3:2]</td><td>PD1MFP</td><td>PD.1 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD1MFP</th></tr>
<tr><td rowspan="2">PD.1</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.1</td><td>UART1_nRTS</td><td>PWM01</td><td>I2S0_LRCK</td></tr>
</table></td></tr>
<tr><td>[1:0]</td><td>PD0MFP</td><td>PD.0 Multi-function Pin Selection
<table>
<tr><th>Pin Name</th><th colspan="4">PD0MFP</th></tr>
<tr><td rowspan="2">PD.0</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PD.0</td><td>UART1_nCTS</td><td>PWM00</td><td>I2S0_MCLK</td></tr>
</table></td></tr>
</table>

GPIO Input type control (SYS_GPIO_INTP)

Register	Offset	R/W	Description	Reset Value
SYS_GPIO_INTP	SYS_BA+0x40	R/W	GPIO input type and slew rate Control	0xFFFF_03FF

31	30	29	28	27	26	25	24
GPD[15:12]HS	**GPD[15:12]SS**	**GPD[11:8]HS**	**GPD[11:8]SS**	**GPD[7:4]HS**	**GPD[7:4]SS**	**GPD[3:0]HS**	**GPD[3:0]SS**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
GPC[15:12]HS	**GPC[15:12]SS**	**GPC[11:8]HS**	**GPC[11:8]SS**	**GPC[7:4]HS**	**GPC[7:4]SS**	**GPC[3:0]HS**	**GPC[3:0]SS**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**GPB[3:0]HS**	**GPB[3:0]SS**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPA[15:12]HS	**GPA[15:12]SS**	**GPA[11:8]HS**	**GPA[11:8]SS**	**GPA[7:4]HS**	**GPA[7:4]SS**	**GPA[3:0]HS**	**GPA[3:0]SS**

Bits	Description	
[31:0]	**GPx[m:n]SS** **GPx[m:n]HS**	This register controls whether the GPIO input buffer Schmitt trigger is enabled and whether high or low slew rate is selected for output driver. Each bit controls a group of four GPIO pins **GPx[m:n]SS = 1 :** input Schmitt Trigger enabled. **GPx[m:n]SS = 0 :** input CMOS enabled. **GPx[m:n]HS = 1:** Output high slew rate. **GPx[m:n]HS = 0 :** Output low slew rate.

PA.15~PA.0 Pull Resistance Control Register (SYS_GPA_PULL)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_PULL	SYS_BA+0x44	R/W	PA.15 ~ PA.0 Pull Resistance Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**PUEN[n]**	**PA.n Pull Control Register. n = 15~0** 1 = Pull-Up function Enable 0 = Pull-Up function Disable. This function only for the GPIO Px[n] pin as an INPUT mode.

PA.15~PA.0 Pull Resistance Select Control Register (SYS_GPA_HR)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_HR	SYS_BA+0x48	R/W	PA.15 ~ PA.0 Pull Resistance Select Control Register	0x0000_FFFF

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**PUHR[n]**	**PA.n Pull Resistance Select Control Register. n = 15~0** 1 = Pull-Up 1M resistance 0 = Pull-Up 100K resistance This function only for the GPIO Px[n] pin as an INPUT mode.

PA.15~PA.0 Digital Buffer Control Register (SYS_GPA_IEN)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_IEN	SYS_BA+0x4C	R/W	PA.15 ~ PA.0 Digital and Analog Input Buffer Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n = 0,1,…15	**IEN[n]**	**PA.n Digital Input Buffer Control Register. n = 15~0** 0 = Input buffer Enabled. 1 = Input buffer disabled, and input signal always equals to 0.

PB.1~PB.0 Pull Resistance Control Register (SYS_GPB_PULL)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_PULL	SYS_BA+0x54	R/W	PB.1 ~ PB.0 Pull Resistance Control Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0, 1	**PUEN[n]**	**PB.n Pull Control Register. n = 1~0** 1 = Pull-Up function Enable 0 = Pull-Up function Disable. This function only for the GPIO Px[n] pin as an INPUT mode.

PB.1~PB.0 Pull Resistance Select Control Register (SYS_GPB_HR)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_HR	SYS_BA+0x58	R/W	PB.1 ~ PB.0 Pull Resistance Select Control Register	0x0000_0003

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

Bits	Description	
[31:2	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1	**PUHR[n]**	**PB.n Pull Resistance Select Control Register. n = 1~0** 1 = Pull-Up 1M resistance 0 = Pull-Up 100K resistance This function only for the GPIO Px[n] pin as an INPUT mode.

PB.1~PB.0 Digital Input Buffer Control Register (SYS_GPB_IEN)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_IEN	SYS_BA+0x5C	R/W	PB.1 ~ PB.0 Digital Input Buffer Control Register	0x0000_0000

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

Bits	Description	
[31:2]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1	IEN[n]	**PB.n Digital Input Buffer Control Register. n = 1~0** 0 = Input buffer Enabled. 1 = Input buffer disabled, and input signal always equals to 0.

PC.15~PC.0 Pull Resistance Control Register (SYS_GPC_PULL)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_PULL	SYS_BA+0x64	R/W	PC.15 ~ PC.0 Pull Resistance Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**PUEN[n]**	**PC.n Pull Control Register. n = 15~0** 1 = Pull-Up function Enable 0 = Pull-Up function Disable. This function only for the GPIO Px[n] pin as an INPUT mode.

PC.15~PC.0 Pull Resistance Select Control Register (SYS_GPC_HR)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_HR	SYS_BA+0x68	R/W	PC.15 ~ PC.0 Pull Resistance Select Control Register	0x0000_FFFF

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**PUHR[n]**	**PC.n Pull Resistance Select Control Register. n = 15~0** 1 = Pull-Up 1M resistance 0 = Pull-Up 100K resistance This function only for the GPIO Px[n] pin as an INPUT mode.

PC.15~PC.0 Digital Buffer Control Register (SYS_GPC_IEN)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_IEN	SYS_BA+0x6C	R/W	PC.15 ~ PC.0 Digital Input Buffer Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n = 0,1,…15	**IEN[n]**	**PC.n Digital Input Buffer Control Register. n = 15~0** 0 = Input buffer Enabled. 1 = Input buffer disabled, and input signal always equals to 0.

PD.15~PD.0 Pull Resistance Control Register (SYS_GPD_PULL)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_PULL	SYS_BA+0x74	R/W	PD.15 ~ PD.0 Pull Resistance Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**PUEN[n]**	**PD.n Pull Control Register. n = 15~0** 1 = Pull-Up function Enable 0 = Pull-Up function Disable. This function only for the GPIO Px[n] pin as an INPUT mode.

PD.15~PD.0 Pull Resistance Select Control Register (SYS_GPD_HR)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_HR	SYS_BA+0x78	R/W	PD.15 ~ PD.0 Pull Resistance Select Control Register	0x0000_FFFF

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**PUHR[n]**	**PD.n Pull Resistance Select Control Register. n = 15~0** 1 = Pull-Up 1M resistance 0 = Pull-Up 100K resistance This function only for the GPIO Px[n] pin as an INPUT mode.

PD.15~PD.0 Digital Buffer Control Register (SYS_GPD_IEN)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_IEN	SYS_BA+0x7C	R/W	PD.15 ~ PD.0 Digital Input Buffer Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n = 0,1,…15	**IEN[n]**	**PD.n Digital Input Buffer Control Register. n = 15~0** 0 = Input buffer Enabled. 1 = Input buffer disabled, and input signal always equals to 0.

Register Lock Control Register (SYS_REGLCTL)

Certain critical system control registers are protected against inadvertent write operations which may disturb chip operation. These system control registers are locked after power on reset until the user specifically issues an unlock sequence to disable register protection. The unlock sequence is to write to SYS_REGLCTL the data 0x59, 0x16, 0x88 sequentially. Any different data value, different sequence or any other write to other address during these three data writing will abort the whole sequence.

User can check the lock status by reading SYS_REGLCTL bit0: "1" is unlocked, "0" is locked. Once unlocked, user can update the target protected register value. To lock registers again, write any data to the register SYS_REGLCTL to enable register protection.

The protected registers are:

Registers	Address	Note
SYS_IPRST0	0x5000_0008	
SYS_BODCTL	0x5000_0018	
CLK_PWRCTL	0x5000_0200	
CLK_APBCLK bit[0]	0x5000_0208	Bit[0] is watchdog clock enable.
CLK_CLKSEL0	0x5000_0210	HCLK and CPU STCLK clock source select.
CLK_CLKSEL1 bit[0]	0x5000_0214	Watchdog clock source select.
ISPCON	0x5000_C000	Flash ISP Control.
ISPTRG	0x5000_C010	ISP Trigger Control.
WDT_CTL	0x4000_4000	Watchdog Timer Control.

This register is" write" accessible to disable/enable register protection and "read" accessible to know the lock/unlock status.

Register	Offset	R/W	Description	Reset Value
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control Register	0x0000_0000

7	6	5	4	3	2	1	0
SYS_REGLCTL[7:1]							**SYS_REGLCTL[0]/ REGLCTL**

Bits	Description

[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**SYS_REGLCTL[7:0] /REGLCTL[0]**	**Register Lock Control Code (Write Only)** Some registers have write-protection function. Writing these registers have to disable the protected function by writing the sequence value “59h”, “16h”, “88h” to this field. After this sequence is completed, the REGLCTL bit will be set to 1 and write-protection registers can be normal write. **Protected Register Lock/Unlock Index (Read Only)** 0 = Protected registers are locked. Any write to the target register is ignored. 1 = Protected registers are unlocked.

Oscillator Trim Control Register (SYS_OSCTRIM)

The master oscillator of the ISD91500 has an adjustable frequency and is controlled by the OSCTRIM and OSC_TRIM[n] registers. There are three factory trimmed settings available for the oscillator, the active one being selected by the OSCFSel bits of CLKSEL0. The OSCTRIM register accesses the current active oscillator trim. The OSC_TRIM[2:0] registers access the individual trims of OSCFSel=0,1&2.

This register is a protected register, to write to register first issue the unlock sequence (see Register Lock Control Register (SYS_REGLCTL))

Register	Offset	R/W	Description	Reset Value
SYS_OSCTRIM	SYS_BA+0x110	R/W	Internal oscillator trim register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EN2MHZ	**Reserved**					**TRIM[9:8]**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRIM[7:0]							

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**EN2MHZ**	1: High frequency mode (20-50 MHz) 0: Low Frequency mode of oscillator active (2 MHz).
[14:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9:0]	**TRIM**	10 bit trim for oscillator,

10kHz Oscillator Trim Control Register (SYS_OSC10K)

Register	Offset	R/W	Description	Reset Value
SYS_OSC10K	SYS_BA+0x114	R/W	10KHz Oscillator and Bias trim register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
TRM_CLK	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**OSC10K_TRIM[22:16]**						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
OSC10K_TRIM[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OSC10K_TRIM[7:0]							

Bits	Description	
[31]	**TRM_CLK**	Must be toggled to load a new OSC10K_TRIM
[30:23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[22:0]	**OSC10K_TRIM**	23bit trim for 10 KHz oscillator. [7:0] = OSC10K TC Trim value [12:8] = OSC10K SLDO Trim value [22:13] = OSC10K RC Frequency Trim value

Oscillator Trim Control Register (OSC_TRIM[n])

Register	Offset	R/W	Description	Reset Value
SYS_OSC_TRIMn **n=0,1,2**	SYS_BA+0x118+0x04*n	R/W	Oscillator Frequency Adjustment control register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
EN2MHZ	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TRIM[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRIM[7:0]							

Bits	Description	
[31]	**EN2MHZ**	1: High frequency mode (20-50 MHz) 0: Low Frequency mode of oscillator active (2 MHz).
[30:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**TRIM**	16bit sign extended representation of 10bit trim. SYS_OSC_TRIM[n] load from factory trim value after reset. One of SYS_OSC_TRIM[n] will map to SYS_OSCTRIM base on OSCFSEL

HIRC Trim Control Register (SYS_IRCTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTCTL	SYS_BA+0x130	R/W	HIRC Trim Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					REFCKSEL	Reserved	CESTOPEN
7	6	5	4	3	2	1	0
RETRYCNT		LOOPSEL		Reserved		FREQSEL	

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**REFCKSEL**	**Reference Clock Selection** 0 = HIRC trim reference clock is from HXT (4~24.576 MHz) . 1 = HIRC trim reference clock USB SOF (Start-Of-Frame) packet. **Note1:** Even HIRC support USB SOF auto-trim, HIRC might still hard to meet USB frequency +/-0.25% spec.
[9]	**Reserved**	**Reserved**
[8]	**CESTOPEN**	**Clock Error Stop Enable Bit** 0 = The trim operation is keep going if clock is inaccuracy. 1 = The trim operation is stopped if clock is inaccuracy.
[7:6]	**RETRYCNT**	**Trim Value Update Limitation Count** This field defines that how many times the auto trim circuit will try to update the HIRC trim value before the frequency of HIRC locked. Once the HIRC locked, the internal trim value update counter will be reset. If the trim value update counter reached this limitation value and frequency of HIRC still doesn't lock, the auto trim operation will be disabled and FREQSEL will be cleared to 00. 00 = Trim retry count limitation is 64 loops. 01 = Trim retry count limitation is 128 loops. 10 = Trim retry count limitation is 256 loops. 11 = Trim retry count limitation is 512 loops.

[5:4]	**LOOPSEL**	**Trim Calculation Loop Selection** This field defines that trim value calculation is based on how many internal reference clocks. 00 = Trim value calculation is based on average difference in 4 clocks of reference clock. 01 = Trim value calculation is based on average difference in 8 clocks of reference clock. 10 = Trim value calculation is based on average difference in 16 clocks of reference clock. 11 = Trim value calculation is based on average difference in 32 clocks of reference clock. **Note1:** For example, if LOOPSEL is set as 00, auto trim circuit will calculate trim value based on the average frequency difference in 4 clocks of reference clock. **Note2:** If source clock from HXT , the internal reference clock is 32 KHz If source clock from SOF , the internal reference clock is 1 KHz.
[3:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**FREQSEL**	**Trim Frequency Selection** This field indicates the target frequency of internal high speed RC oscillator (HIRC) auto trim. During auto trim operation, if clock error detected with CESTOPEN is set to 1 or trim retry limitation count reached, this field will be cleared to 00 automatically. 00 = Disable HIRC auto trim function. 01 = Enable HIRC auto trim function and trim HIRC to 48 MHz. 10 = Disable HIRC auto trim function. 11 = Enable HIRC auto trim function and trim HIRC to 49.152 MHz.

HIRC Trim Interrupt Enable Register (SYS_IRCTIEN)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTIEN	SYS_BA+0x134	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**CLKEIEN**	**Clock Error Interrupt Enable Bit** This bit controls if CPU would get an interrupt while clock is inaccuracy during auto trim operation. If this bit is set to1, and CLKERRIF (SYS_IRCTSTS [2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = Disable CLKERRIF (SYS_IRCTSTS [2]) status to trigger an interrupt to CPU. 1 = Enable CLKERRIF (SYS_IRCTSTS [2]) status to trigger an interrupt to CPU.
[1]	**TFAILIEN**	**Trim Failure Interrupt Enable Bit** This bit controls if an interrupt will be triggered while HIRC trim value update limitation count reached and HIRC frequency still not locked on target frequency set by FREQSEL (SYS_IRCTCTL [1:0]). If this bit is high and TFAILIF (SYS_IRCTSTS [1]) is set during auto trim operation, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. 0 = Disable TFAILIF (SYS_IRCTSTS [1]) status to trigger an interrupt to CPU. 1 = Enable TFAILIF (SYS_IRCTSTS [1]) status to trigger an interrupt to CPU.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

HIRC Trim Interrupt Status Register (SYS_IRCTISTS)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTISTS	SYS_BA+0x138	R/W	HIRC Trim Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**CLKERRIF**	**Clock Error Interrupt Status** When the frequency of external high speed crystal oscillator (HXT) or internal high speed RC oscillator (HIRC) is shift larger to unreasonable value, this bit will be set and to be an indicate that clock frequency is inaccuracy Once this bit is set to 1, the auto trim operation stopped and FREQSEL (SYS_IRCTCL [1:0]) will be cleared to 00 by hardware automatically if CESTOPEN (SYS_IRCTCTL [8]) is set to 1. If this bit is set and CLKEIEN (SYS_IRCTIEN [2]) is high, an interrupt will be triggered to notify the clock frequency is inaccuracy. Write 1 to clear this to 0. 0 = Clock frequency is accuracy. 1 = Clock frequency is inaccuracy.
[1]	**TFAILIF**	**Trim Failure Interrupt Status** This bit indicates that HIRC trim value update limitation count reached and the HIRC clock frequency still doesn't be locked. Once this bit is set, the auto trim operation stopped and FREQSEL (SYS_IRCTCTL [1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TFAILIEN (SYS_IRCTIEN [1]) is high, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. Write 1 to clear this to 0. 0 = Trim value update limitation count does not reach. 1 = Trim value update limitation count reached and HIRC frequency still not locked.

[0]	**FREQLOCK**	**HIRC Frequency Lock Status** This bit indicates the HIRC frequency is locked. This is a status bit and doesn't trigger any interrupt Write 1 to clear this to 0. This bit will be set automatically, if the frequecy is lock and the RC_TRIM is enabled. 0 = The internal high-speed oscillator frequency doesn't lock at target frequency yet. 1 = The internal high-speed oscillator frequency locked at target frequency.

HIRC Trim Clock Reference Frequency Register (SYS_IRCTCKRF)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTCKRF	SYS_BA+0x13C	R/W	HIRC Trim Clock Reference Frequency Register	0x0000_0020

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

Bits	Description	
[31:15]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14:0]	HXTFREQ	**HIRC Trim reference clock frequency value when reference clock from HXT** User can insert the HXT frequency on PCB to this register for internal trim. The insert frequency value is unit KHz. For example: If HXT = 4 MHz(4000KHz), register value = 0xFA0. If HXT = 12 MHz(12000KHz), register value = 0x2EE0. If HXT = 24.576 MHz(24576KHz), register value = 0x6000. **Note1: The HXT frequency register should set correct value before HIRC auto trim enable.** **Note2: It recommends the HXT should be multiple of 4MHz or 4.096MHz.**

Bandgap Trim Control Register (SYS_BGAPTRIM)

Register	Offset	R/W	Description	Reset Value
SYS_BGAPTRIM	SYS_BA+0x140	R/W	Bandgap Trim Control Register	0x0000_000X

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

Bits	Description	
[31:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3:0]	**TRIM**	**4 bit trim for Bandgap.**

Uniq Customer ID Register (SYS_UCID[n])

Register	Offset	R/W	Description	Reset Value
SYS_UCID[n] n=0,1,2,3	SYS_BA+0x150+0x04*n	R	Specified ID register for library and customized feature checking	0xXXXX_XXXX

This register provides specific read-only information for software to check the UCID.

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

Bits	Description	
[31:0]	**UCID**	**UCID Value** This register provides specific read-only information for the UCID

5.2.4 System Timer (SYST)

The Cortex-M0 includes an integrated system timer, SYST. SYST provides a simple, 24-bit,Clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used in several different ways, for example:

- An RTOS tick timer which fires at a programmable rate (for example 100Hz) and invokes a SYST routine.
- A high speed alarm timer using Core clock.
- A variable rate alarm or signal timer – the duration range dependent on the reference clock used and the dynamic range of the counter.
- A simple counter. Software can use this to measure time to completion and time used.
- An internal clock source control based on missing/meeting durations. The COUNTFLAG bit-field in the control and status register can be used to determine if an action completed within a set duration, as part of a dynamic clock management control loop.

When enabled, the timer will count down from the value in the SYST Current Value Register (SYST_CVR) to zero, reload (wrap) to the value in the SYST Reload Value Register (SYST_RVR) on the next clock edge, then decrement on subsequent clocks. When the counter transitions to zero, the COUNTFLAG status bit is set. The COUNTFLAG bit clears on reads.

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

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

In DEEPSLEEP and power down modes, the SYST timer is disabled so cannot be used to wake up the device.

For more detailed information, please refer to the documents “ARM®Cortex™-M0 Technical Reference Manual” and “ARM® v6-M Architecture Reference Manual”.

5.2.4.1 Register Map

R: read only, **W**: write only, **R/W**: both read and write, **W&C**: Write 1 clear

Register	Offset	R/W	Description	Reset Value
SYSTICK Base Address: **SYSTICK_BA = 0xE000_E000**				
SYST_CSR	SYSTICK_BA+0x10	R/W	SYST Control and Status Register	0x0000_0000
SYST_RVR	SYSTICK_BA+0x14	R/W	SYST Reload Value Register	0xXXXX_XXXX
SYST_CVR	SYSTICK_BA+0x18	R/W	SYST Current Value Register	0xXXXX_XXXX

5.2.4.2 **Register Description**

SYST Control and Status（SYST_CSR）

Register	Offset	R/W	Description	Reset Value
SYST_CSR	SYSTICK_BA+0x10	R/W	SYST Control and Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**COUNTFLAG**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**CLKSRC**	**TICKINT**	**ENABLE**

Bits	Description	
[31:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**COUNTFLAG**	**Count Flag** Returns 1 if timer counted to 0 since last time this register was read. 0= Cleared on read or by a write to the Current Value register. 1= Set by a count transition from 1 to 0.
[15:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**CLKSRC**	**Clock Source** 0 = Clock selected from CLK_CLKSEL0.STCLKSEL is used as clock source. 1 = Core clock used for SYST.
[1]	**TICKINT**	**Enables SYST Exception Request** 0 = Counting down to 0 does not cause the SYST exception to be pended. Software can use COUNTFLAG to determine if a count to zero has occurred. 1 = Counting down to 0 will cause SYST exception to be pended. Clearing the SYST Current Value register by a register write in software will not cause SYST to be pended.
[0]	**ENABLE**	**ENABLE** 0 = The counter is disabled. 1 = The counter will operate in a multi-shot manner.

SYST Reload Value Register（SYST_RVR）

Register	Offset	R/W	Description	Reset Value
SYST_RVR	SYSTICK_BA+0x14	R/W	SYST Reload Value Register	0xXXXX_XXXX

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:0]	**RELOAD**	**SYST Reload** Value to load into the Current Value register when the counter reaches 0. To generate a multi-shot timer with a period of N processor clock cycles, use a RELOAD value of N-1. For example, if the SYST interrupt is required every 200 clock pulses, set RELOAD to 199.

SYST Current Value Register（SYST_CVR）

Register	Offset	R/W	Description	Reset Value
SYST_CVR	SYSTICK_BA+0x18	R/W	SYST Current Value Register	0xXXXX_XXXX

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:0]	**CURRENT**	**Current Counter Value** This is the value of the counter at the time it is sampled. The counter does not provide read-modify-write protection. The register is write-clear. A software write of any value will clear the register to 0 and also clear the COUNTFLAG bit.

5.2.5 Nested Vectored Interrupt Controller (NVIC)

Cortex-M0 provides an interrupt controller as an integral part of the exception mode, named as “Nested Vectored Interrupt Controller (NVIC)”. It is closely coupled to the processor kernel and provides following features:

- Nested and Vectored interrupt support
- Automatic processor state saving and restoration
- Dynamic priority changing
- Reduced and deterministic interrupt latency

The NVIC prioritizes and handles all supported exceptions. All exceptions are handled in “Handler Mode”. This NVIC architecture supports 16 (IRQ [15:0]) discrete interrupts with 4 levels of priority. All of the interrupts and most of the system exceptions can be configured to different priority levels. When an interrupt occurs, the NVIC will compare the priority of the new interrupt to the current running one’s priority. If the priority of the new interrupt is higher than the current one, the new interrupt handler will override the current handler.
When any interrupt is accepted, the starting address of the interrupt service routine (ISR) is fetched from a vector table in memory. There is no need to determine which interrupt is accepted and branch to the starting address of the correlated ISR by software. While the starting address is fetched, NVIC will also automatically save processor state including the registers “PC, PSR, LR, R0~R3, R12” to the stack. At the end of the ISR, the NVIC will restore the mentioned registers from stack and resume the normal execution. Thus it will take less and deterministic time to process the interrupt request.
The NVIC supports “Tail Chaining” which handles back-to-back interrupts efficiently without the overhead of states saving and restoration and therefore reduces delay time in switching to pending ISR at the end of current ISR. The NVIC also supports “Late Arrival” which improves the efficiency of concurrent ISRs. When a higher priority interrupt request occurs before the current ISR starts to execute (at the stage of state saving and starting address fetching), the NVIC will give priority to the higher one without delay penalty. Thus it advances the real-time capability.

For more detailed information, please refer to the documents “ARM® Cortex™-M0 Technical Reference Manual” and “ARM® v6-M Architecture Reference Manual”.

5.2.5.1 Exception Model and System Interrupt Map

The following table lists the exception model supported by ISD91500064. Software can set four levels of priority on certain exceptions as well as on all interrupts. The highest user-configurable priority is denoted as “0” and the lowest priority is denoted as “3”. The default priority of all the user-configurable interrupts is “0”. Note that priority “0” is treated as the fourth priority on the system, after three system exceptions “Reset”, “NMI” and “Hard Fault”.

Table 5.2-2 Exception Model

Exception Name	Vector Number	Priority
Reset	1	-3
NMI	2	-2
Hard Fault	3	-1
Reserved	4 ~ 10	N/A
SVCall	11	Configurable
Reserved	12 ~ 13	N/A

PendSV	14	Configurable
SysTick	15	Configurable
Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	Configurable

Table 5.2-3 System Interrupt Map

Vector Number	Interrupt Number (Bit In Interrupt Registers)	Interrupt Name	Source IP	Interrupt Description
0 ~ 15	-	**-**	-	System exceptions
16	0	**WDT_INT**	WDT	Watch Dog Timer interrupt
17	1	**DAC_INT**	DAC	DAC threshold interrupt
18	2	**SARADC_INT**	SARADC	Interrupt from SARADC
19	3	**SDADC_INT**	SDADC	Interrupt from SDADC
20	4	**I2S0_INT**	I2S0	I2S0 interrupt
21	5	**TMR0_INT**	Timer0	Timer0 interrupt
22	6	**TMR1_INT**	Timer1	Timer1 interrupt
23	7	**TMR2_INT**	Timer2	Timer2 interrupt
24	8	**GPA_INT**	GPIO A	Port interrupt from GPIO port A
25	9	**GPB_INT**	GPIO B	Port interrupt from GPIO port B
26	10	**GPC_INT**	GPIO C	Port interrupt from GPIO port C
27	11	**GPD_INT**	GPIO D	Port interrupt from GPIO port D
28	12	**SPI0_INT**	SPI0	Interrupt from SPI0
29	13	**PWM0_INT**	PWM0 Timer	PWM0 Timer interrupt
30	14	**PWM1_INT**	PWM1 Timer	PWM1 Timer interrupt
31	15	**PDMA**	PDMA	Interrupt from PDMA
32	16	**I2C0_INT**	I2C0	I2C0 interrupt
33	17	**I2C1_INT**	I2C1	I2C1 interrupt
34	18	**BOD_INT**	BOD	Brown-out detector interrupt.
35	19	**Reserved**	-	-
36	20	**UART0_INT**	UART0	UART0 Serial interface interrupt
37	21	**UART1_INT**	UART1	UART1 Serial interface interrupt.

38	22	**IRCTRIM_INT**	IRCTRIM	IRCTRIM intrrupt
39	23	**USB_INT**	USB	USB interrupt
40	24	**CPD_INT**	Companding	Companding interrupt
41	25	**XCLKF_INT**	XCLK Fail	XCLK Fail interrupt
42	26	**SPI1_INT**	SPI1	Interrupt from SPI1
>43	>27	**Reserved**	-	-

5.2.5.2 Vector Table

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

Table 5.2-4 Vector Table Format

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

5.2.5.3 Operation Description

NVIC interrupts can be enabled and disabled by writing to their corresponding Interrupt Set-Enable or Interrupt Clear-Enable register bit-field. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current enabled state of the corresponding interrupts. When an interrupt is disabled, interrupt assertion will cause the interrupt to become Pending, however, the interrupt will not activate. If an interrupt is Active when it is disabled, it remains in its Active state until cleared by reset or an exception return. Clearing the enable bit prevents new activations of the associated interrupt.
NVIC interrupts can be pended/un-pended using a complementary pair of registers to those used to enable/disable the interrupts, named the Set-Pending Register and Clear-Pending Register respectively. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current pended state of the corresponding interrupts. The Clear-Pending Register has no effect on the execution status of an Active interrupt.
NVIC interrupts are prioritized by updating an 8-bit field within a 32-bit register (each register supporting four interrupts).
The general registers associated with the NVIC are all accessible from a block of memory in the System Control Space and will be described in next section.

5.2.5.4 NVIC Control Registers

R: read only, **W**: write only, **R/W**: both read and write, **W&C**: Write 1 clear

Register	Offset	R/W	Description	Reset Value
SCS Base Address: **SCS_BA = 0xE000_E000**				
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ26 Set-Enable Control Register	0x0000_0000
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ26 Clear-Enable Control Register	0x0000_0000
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ26 Set-Pending Control Register	0x0000_0000
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ26 Clear-Pending Control Register	0x0000_0000
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 Priority Control Register	0x0000_0000
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 Priority Control Register	0x0000_0000
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11 Priority Control Register	0x0000_0000
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15 Priority Control Register	0x0000_0000
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19 Priority Control Register	0x0000_0000
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 Priority Control Register	0x0000_0000
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ26 Priority Control Register	0x0000_0000

IRQ0 ~ IRQ26 Set-Enable Control Register （NVIC_ISER）

Register	Offset	R/W	Description	Reset Value
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ26 Set-Enable Control Register	0x0000_0000

If a pending interrupt is enabled, the NVIC activates the interrupt based on its priority. If an interrupt is not enabled, asserting its interrupt signal changes the interrupt state to pending, but the NVIC never activates the interrupt, regardless of its priority.

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

Bits	Description	
[31:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25:0]	**SETENA**	**Interrupt Set-Enable Bit** The NVIC_ISER register enables interrupts, and shows what interrupts are enabled. Each bit represents an interrupt number from IRQ0 ~ IRQ26 (Vector number from 16 ~ 42). Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ26 Clear-Enable Control Register (NVIC_ICER）

Register	Offset	R/W	Description	Reset Value
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ26 Clear-Enable Control Register	0x0000_0000

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

Bits	Description	
[31:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25:0]	**CLRENA**	**Interrupt Clear-Enable Bit** The NVIC_ICER register disables interrupts, and shows what interrupts are enabled. Each bit represents an interrupt number from IRQ0 ~ IRQ26 (Vector number from 16 ~ 42). Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ26 Set-Pending Control Register （NVIC_ISPR）

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ26 Set-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25:0]	**SETPEND**	**Interrupt Set-Pending Bit** The NVIC_ISPR register forces interrupts into the pending state, and shows what interrupts are pending. Each bit represents an interrupt number from IRQ0 ~ IRQ26 (Vector number from 16 ~ 42). Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ26 Clear-Pending Control Register（NVIC_ICPR）

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ26 Clear-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25:0]	**CLRPEND**	**Interrupt Clear-Pending Bit** The NVIC_ICPR register removes the pending state of associated interrupts, and shows what interrupts are pending. Each bit represents an interrupt number from IRQ0 ~ IRQ26 (Vector number from 16 ~ 42). Write Operation: 0 = No effect. 1 = Removes pending state of an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ3 Interrupt Priority Register （NVIC_IPR0）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_3		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_2		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_1		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_0		**Reserved**					

Bits	Description	
[31:30]	**PRI_3**	**Priority Of IRQ3** “0” denotes the highest priority and “3” denotes lowest priority
[23:22]	**PRI_2**	**Priority Of IRQ2** “0” denotes the highest priority and “3” denotes lowest priority
[15:14]	**PRI_1**	**Priority Of IRQ1** “0” denotes the highest priority and “3” denotes lowest priority
[7:6]	**PRI_0**	**Priority Of IRQ0** “0” denotes the highest priority and “3” denotes lowest priority

IRQ4 ~ IRQ7 Interrupt Priority Register （NVIC_IPR1）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_7		Reserved					
23	22	21	20	19	18	17	16
PRI_6		Reserved					
15	14	13	12	11	10	9	8
PRI_5		Reserved					
7	6	5	4	3	2	1	0
PRI_4		Reserved					

Bits	Description	
[31:30]	**PRI_7**	**Priority Of IRQ7** “0” denotes the highest priority and “3” denotes lowest priority
[23:22]	**PRI_6**	**Priority Of IRQ6** “0” denotes the highest priority and “3” denotes lowest priority
[15:14]	**PRI_5**	**Priority Of IRQ5** “0” denotes the highest priority and “3” denotes lowest priority
[7:6]	**PRI_4**	**Priority Of IRQ4** “0” denotes the highest priority and “3” denotes lowest priority

IRQ8 ~ IRQ11 Interrupt Priority Register （NVIC_IPR2）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11		Reserved					
23	22	21	20	19	18	17	16
PRI_10		Reserved					
15	14	13	12	11	10	9	8
PRI_9		Reserved					
7	6	5	4	3	2	1	0
PRI_8		Reserved					

Bits	Description	
[31:30]	**PRI_11**	**Priority Of IRQ11** “0” denotes the highest priority and “3” denotes lowest priority
[23:22]	**PRI_10**	**Priority Of IRQ10** “0” denotes the highest priority and “3” denotes lowest priority
[15:14]	**PRI_9**	**Priority Of IRQ9** “0” denotes the highest priority and “3” denotes lowest priority
[7:6]	**PRI_8**	**Priority Of IRQ8** “0” denotes the highest priority and “3” denotes lowest priority

IRQ12 ~ IRQ15 Interrupt Priority Register （NVIC_IPR3）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_15		RESERVED					
23	22	21	20	19	18	17	16
PRI_14		RESERVED					
15	14	13	12	11	10	9	8
PRI_13		RESERVED					
7	6	5	4	3	2	1	0
PRI_12		RESERVED					

Bits	Description	
[31:30]	PRI_15	**Priority Of IRQ15** “0” denotes the highest priority and “3” denotes lowest priority
[23:22]	PRI_14	**Priority Of IRQ14** “0” denotes the highest priority and “3” denotes lowest priority
[15:14]	PRI_13	**Priority Of IRQ13** “0” denotes the highest priority and “3” denotes lowest priority
[7:6]	PRI_12	**Priority Of IRQ12** “0” denotes the highest priority and “3” denotes lowest priority

IRQ16 ~ IRQ19 Interrupt Priority Register （NVIC_IPR4）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_19		**RESERVED**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_18		**RESERVED**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_17		**RESERVED**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_16		**RESERVED**					

Bits	Description	
[31:30]	**PRI_19**	**Priority Of IRQ19** “0” denotes the highest priority and “3” denotes lowest priority
[23:22]	**PRI_18**	**Priority Of IRQ18** “0” denotes the highest priority and “3” denotes lowest priority
[15:14]	**PRI_17**	**Priority Of IRQ17** “0” denotes the highest priority and “3” denotes lowest priority
[7:6]	**PRI_16**	**Priority Of IRQ16** “0” denotes the highest priority and “3” denotes lowest priority

IRQ20 ~ IRQ23 Interrupt Priority Register （NVIC_IPR5）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_23		RESERVED					
23	22	21	20	19	18	17	16
PRI_22		RESERVED					
15	14	13	12	11	10	9	8
PRI_21		RESERVED					
7	6	5	4	3	2	1	0
PRI_20		RESERVED					

Bits	Description	
[31:30]	PRI_23	**Priority Of IRQ23** “0” denotes the highest priority and “3” denotes lowest priority
[23:22]	PRI_22	**Priority Of IRQ22** “0” denotes the highest priority and “3” denotes lowest priority
[15:14]	PRI_21	**Priority Of IRQ21** “0” denotes the highest priority and “3” denotes lowest priority
[7:6]	PRI_20	**Priority Of IRQ20** “0” denotes the highest priority and “3” denotes lowest priority

IRQ24 ~ IRQ26 Interrupt Priority Register （NVIC_IPR6）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ26 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
RESERVED							
23	22	21	20	19	18	17	16
PRI_26		RESERVED					
15	14	13	12	11	10	9	8
PRI_25		RESERVED					
7	6	5	4	3	2	1	0
PRI_24		RESERVED					

Bits	Description	
[23:22]	PRI_26	**Priority Of IRQ26** “0” denotes the highest priority and “3” denotes lowest priority
[15:14]	PRI_25	**Priority Of IRQ25** “0” denotes the highest priority and “3” denotes lowest priority
[7:6]	PRI_24	**Priority Of IRQ24** “0” denotes the highest priority and “3” denotes lowest priority

5.2.5.5 Interrupt Source Control Registers

Along with the interrupt control registers associated with the NVIC, ISD91500 also implements some specific control registers to facilitate the interrupt functions, including “interrupt source identify” and”NMI source selection”. They are described as below.

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
INT Base Address: **INT_BA = 0x5000_0300**				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (WDT) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (DAC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (SARADC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (SDADC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (I2S0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (Timer0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (Timer1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (Timer2) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (GPA) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (GPB) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (GPC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (GPD) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (SPI0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (PWM0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (PWM1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (PDMA) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (I2C0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (I2C1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (BOD) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	R	Reserved	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (UART0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (UART1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (IRCTRIM) Interrupt Source Identity Register	0xXXXX_XXXX

IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USB) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	IRQ24 (CPD) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (XCLKF) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (SPI1) Interrupt Source Identity Register	0xXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_001F

IRQ0 (WDT) Interrupt Source Identify Register (IRQ0_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (WDT) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: WDT_INT

IRQ1 (DAC) Interrupt Source Identify Register (IRQ1_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (DAC) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: DAC_INT

IRQ2 (SARADC) Interrupt Source Identify Register (IRQ2_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (SARADC) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: SARADC_INT

IRQ3 (SDADC) Interrupt Source Identify Register (IRQ3_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (SDADC) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: SDADC_INT

IRQ4 (I2S0) Interrupt Source Identify Register (IRQ4_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (I2S0) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: I2S0_INT

IRQ5 (Timer0) Interrupt Source Identify Register (IRQ5_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (Timer0) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: Timer0_INT

IRQ6 (Timer1) Interrupt Source Identify Register (IRQ6_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (Timer1) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: Timer1_INT

IRQ7 (Timer2) Interrupt Source Identify Register (IRQ7_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (Timer2) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: Timer2_INT

IRQ8 (GPA) Interrupt Source Identify Register (IRQ8_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (GPA) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: GPA_INT

IRQ9 (GPB) Interrupt Source Identify Register (IRQ9_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (GPB) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: GPB_INT

IRQ10 (GPC) Interrupt Source Identify Register (IRQ10_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (GPC) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: GPC_INT

IRQ11 (GPD) Interrupt Source Identify Register (IRQ11_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (GPD) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: GPD_INT

IRQ12 (SPI0) Interrupt Source Identify Register (IRQ12_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (SPI0) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: SPI0_INT

IRQ13 (PWM0) Interrupt Source Identify Register (IRQ13_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (PWM0) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: PWM0_INT

IRQ14 (PWM1) Interrupt Source Identify Register (IRQ14_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (PWM1) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: PWM1_INT

IRQ15 (PDMA) Interrupt Source Identify Register (IRQ15_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (PDMA) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: PDMA_INT

IRQ16 (I2C0) Interrupt Source Identify Register (IRQ16_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (I2C0) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: I2C0_INT

IRQ17 (I2C1) Interrupt Source Identify Register (IRQ17_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (I2C1) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: I2C1_INT

IRQ18 (BOD) Interrupt Source Identify Register (IRQ18_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (BOD) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: BOD_INT

IRQ20 (UART0) Interrupt Source Identify Register (IRQ20_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (UART0) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: UART0_INT

IRQ21 (UART1) Interrupt Source Identify Register (IRQ21_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (UART1) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: UART1_INT

IRQ22 (IRCTRIM) Interrupt Source Identify Register (IRQ22_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (IRCTRIM) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: IRCTRIM_INT

IRQ23 (USB) Interrupt Source Identify Register (IRQ23_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USB) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: USB_INT

IRQ24 (CPD) Interrupt Source Identify Register (IRQ24_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ24_SRC	INT_BA+0x60	R	IRQ24 (CPD) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: CPD_INT

IRQ25 (XCLKF) Interrupt Source Identify Register (IRQ25_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (XCLKF) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: XCLKF_INT

IRQ26 (SPI1) Interrupt Source Identify Register (IRQ26_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (SPI1) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: SPI1_INT

NMI Interrupt Source Select Control Register (NMI_SEL)

Register	Offset	R/W	Description	Reset Value
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_001F

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

Bits	Description	
[31:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4:0]	**NMI_SEL**	**NMI Source Interrupt Select** The NMI interrupt to Cortex-M0 can be selected from one of the interrupt [0:25]. The NMI_SEL bit is used to select the NMI interrupt source. **Note:** IRQ19 are reserved in ISD91500.

5.2.6 System Control Registers

Key control and status features of Coterx-M0 are managed centrally in a System Control Block within the System Control Registers.

For more detailed information, please refer to the documents "ARM® Cortex™-M0 Technical Reference Manual" and ""ARM® v6-M Architecture Reference Manual".

R: read only, **W**: write only, **R/W**: both read and write, **W&C**: Write 1 clear

Register	Offset	R/W	Description	Reset Value
SYSINFO Base Address: **SYSINFO_BA = 0xE000_ED00**				
SYSINFO_CPUID	SYSINFO_BA+0x000	R	CPUID Base Register	0x410C_C200
SYSINFO_ICSR	SYSINFO_BA+0x004	R/W	Interrupt Control State Register	0x0000_0000
SYSINFO_AIRCTL	SYSINFO_BA+0x00C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000
SYSINFO_SCR	SYSINFO_BA+0x010	R/W	System Control Register	0x0000_0000
SYSINFO_SHPR2	SYSINFO_BA+0x01C	R/W	System Handler Priority Register 2	0x0000_0000
SYSINFO_SHPR3	SYSINFO_BA+0x020	R/W	System Handler Priority Register 3	0x0000_0000

CPUID Base Register (SYSINFO_CPUID)

Register	Offset	R/W	Description	Reset Value
SYSINFO_CPUID	SYSINFO_BA+0x000	R	CPUID Base Register	0x410C_C200

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

Bits	Description	
[31:24]	**IMPCODE**	**Implementer Code Assigned by ARM** ARM = 0x41.
[23:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[19:16]	**PART**	**ARMv6-m Parts** Reads as 0xC for ARMv6-M parts
[15:4]	**PARTNO**	**Part Number** Reads as 0xC20.
[3:0]	**REVISION**	**Revision** Reads as 0x0

Interrupt Control State Register (SYSINFO_ICSR)

Register	Offset	R/W	Description	Reset Value
SYSINFO_ICSR	SYSINFO_BA+0x004	R/W	Interrupt Control State Register	0x0000_0000

31	30	29	28	27	26	25	24
NMIPNSET	Reserved		PPSVISET	PPSVICLR	PSTKISET	PSTKICLR	Reserved
23	22	21	20	19	18	17	16
ISRPREEM	ISRPEND	Reserved	VTPNDING[8:4]				
15	14	13	12	11	10	9	8
VTPEND[3:0]				Reserved			VTACT[8]
7	6	5	4	3	2	1	0
VTACT[7:0]							

Bits	Description	
[31]	**NMIPNSET**	**NMI Pending Set Control** Setting this bit will activate an NMI. Since NMI is the highest priority exception, it will activate as soon as it is registered. Reads back with current state (1 if Pending, 0 if not).
[30:29]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[28]	**PPSVISET**	**Set a Pending PendSV Interrupt** This is normally used to request a context switch. Reads back with current state (1 if Pending, 0 if not).
[27]	**PPSVICLR**	**Clear a Pending PendSV Interrupt** Write 1 to clear a pending PendSV interrupt.
[26]	**PSTKISET**	**Set a Pending SYST** Reads back with current state (1 if Pending, 0 if not).
[25]	**PSTKICLR**	**Clear a Pending SYST** Write 1 to clear a pending SYST.
[23]	**ISRPREEM**	**ISR Preemptive** If set, a pending exception will be serviced on exit from the debug halt state.
[22]	**ISRPEND**	**ISR Pending** Indicates if an external configurable (NVIC generated) interrupt is pending.
[21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[20:12]	**VTPEND**	**Vector Pending** Indicates the exception number for the highest priority pending exception. The pending state includes the effect of memory-mapped enable and mask registers. It does not include the PRIMASK special-purpose register qualifier. A value of zero indicates no pending exceptions.
[11:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:0]	**VTACT**	**Vector Active** 0: Thread mode Value > 1: the exception number for the current executing exception.

Application Interrupt and Reset Control Register (SYSINFO_AIRCTL)

Register	Offset	R/W	Description	Reset Value
SYSINFO_AIRCTL	SYSINFO_BA+0x00C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000

31	30	29	28	27	26	25	24
VTKEY							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
VTKEY							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ENDIANES	**Reserved**						
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**SRSTREQ**	**CLRACTVT**	**Reserved**

Bits	Description	
[31:16]	**VTKEY**	**Vector Key** The value 0x05FA must be written to this register, otherwise a write to register is UNPREDICTABLE.
[15]	**ENDIANES**	**Endianness** Read Only. Reads 0 indicating little endian machine.
[14:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**SRSTREQ**	**System Reset Request** **0 =**do not request a reset. **1 =**request reset. Writing 1 to this bit asserts a signal to request a reset by the external system.
[1]	**CLRACTVT**	**Clear All Active Vector** Clears all active state information for fixed and configurable exceptions. **0=** do not clear state information. **1=** clear state information. The effect of writing a 1 to this bit if the processor is not halted in Debug, is UNPREDICTABLE.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

System Control Register (SYSINFO_SCR)

Register	Offset	R/W	Description	Reset Value
SYSINFO_SCR	SYSINFO_BA+0x010	R/W	System Control Register	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**SEVNONPN**	**Send Event on Pending Bit** 0 = only enabled interrupts or events can wake-up the processor, disabled interrupts are excluded. 1 = enabled events and all interrupts, including disabled interrupts, can wake-up the processor. When enabled, interrupt transitions from Inactive to Pending are included in the list of wakeup events for the WFE instruction. When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects the next WFE. The processor also wakes up on execution of an SEV instruction.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**SLPDEEP**	**Controls Whether the Processor Uses Sleep or Deep Sleep As Its Low Power Mode** 0 = sleep. 1 = deep sleep. The SLPDEEP flag is also used in conjunction with CLK_PWRCTL register to enter deeper power-down states than purely core sleep states.

[1]	**SLPONEXC**	**Sleep on Exception** When set to 1, the core can enter a sleep state on an exception return to Thread mode. This is the mode and exception level entered at reset, the base level of execution. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

System Handler Priority Register2 (SYSINFO_SHPR2)

Register	Offset	R/W	Description	Reset Value
SYSINFO_SHPR2	SYSINFO_BA+0x01C	R/W	System Handler Priority Register 2	0x0000_0000

31	30	29	28	27	26	25	24
PRI11		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:30]	**PRI11**	**Priority of System Handler 11 – SVCall** “0” denotes the highest priority and “3” denotes lowest priority
[29:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

System Handler Priority Register3 (SYSINFO_SHPR3)

Register	Offset	R/W	Description	Reset Value
SYSINFO_SHPR3	SYSINFO_BA+0x020	R/W	System Handler Priority Register 3	0x0000_0000

31	30	29	28	27	26	25	24
PRI15		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI14		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:30]	**PRI15**	**Priority of System Handler 15 – SYST** “0” denotes the highest priority and “3” denotes lowest priority
[29:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:22]	**PRI14**	**Priority of System Handler 14 – PendSV** “0” denotes the highest priority and “3” denotes lowest priority
[21:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

5.3 Clock Controller

5.3.1 Overview

The clock controller generates the clock sources for the whole chip. It includes all AMBA interface modules and all peripheral clocks, ADC, and so on. The controller also implements the power control function, include the individually clock on or off control register, clock source select and the divided number from clock source. These functions minimize the extra power consumption and the chip run on the just clock condition. The chip will enter power-down mode after setting both the PD_WAIT_CPU and PWR_DOWN bits, and later the CPU Cortex-M0 executes the WFI or the WFE instruction. On the power down mode, the controller turns off the internal oscillator and system clock related circuit (except LIRC) to reduce the power consumption to minimum.

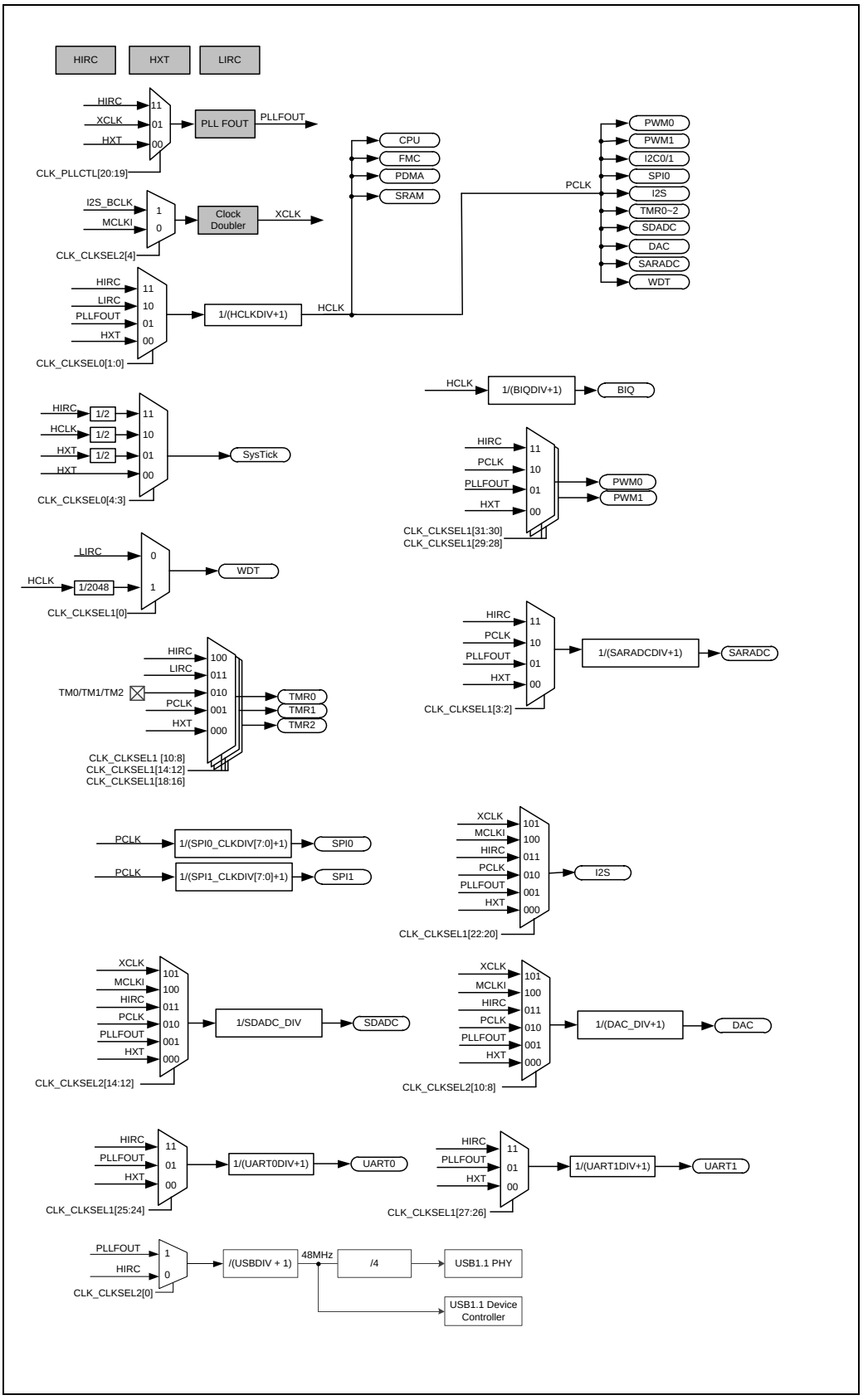


Figure 5.3-1 Clock Tree

5.3.2 Clock Generator

The clock generator consists of 3 sources which are listed below:

- One external 4~24MHz crystal (High speed external crystal, HXT).
- One internal 10 KHz RC oscillator (Low speed internal Oscillator, LIRC).
- One internal 49.152 MHz RC oscillator (High speed internal Oscillator, HIRC).
- Programmable PLL output clock frequency (PLLFOUT), PLL source can be selected from external 4~24.576 MHz external high speed crystal (HXT), internal high speed oscillator (HIRC) and Clock doubler(XCLK).
- Clock Doubler minimum input frequency 512KHz

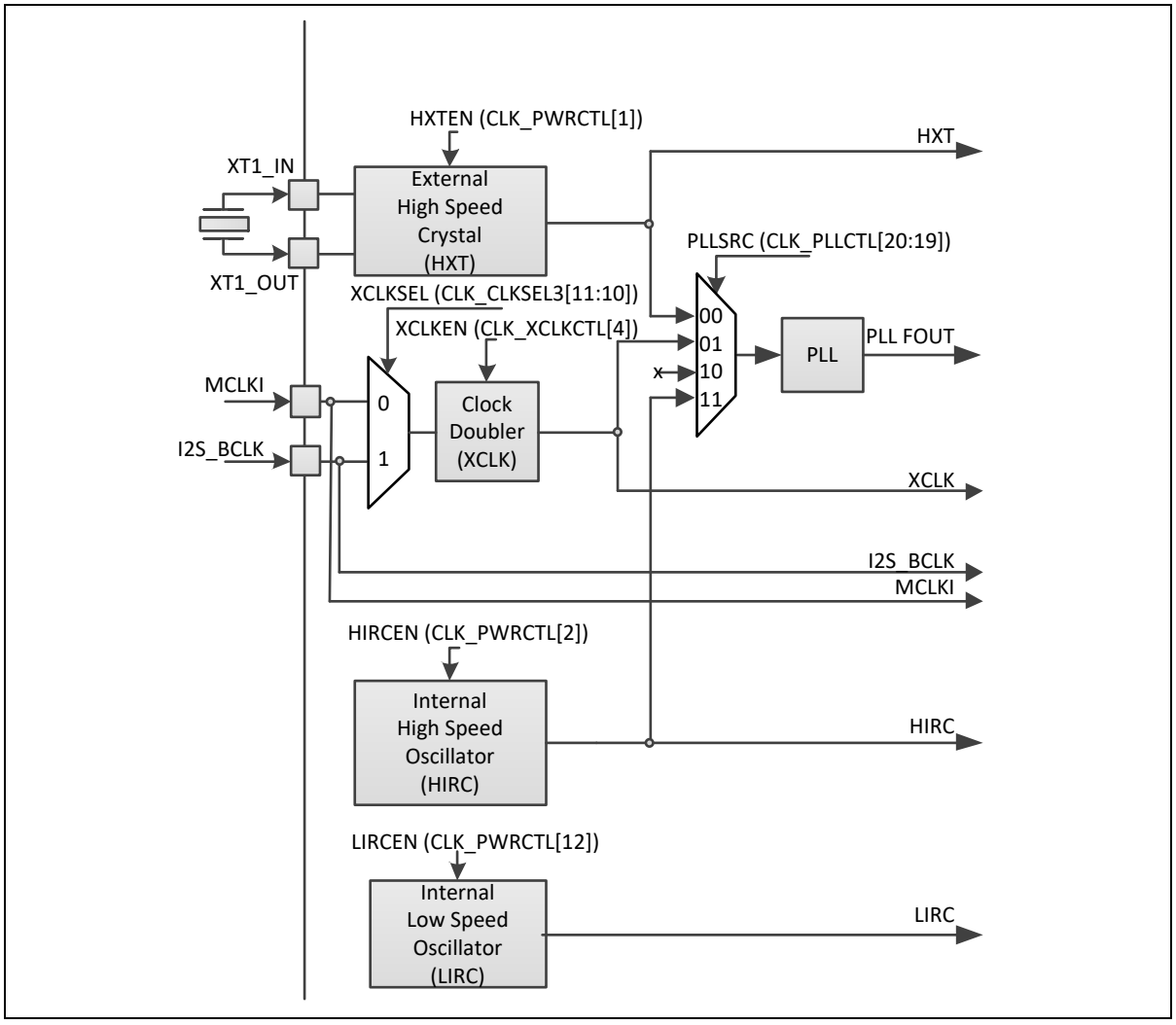


Figure 5.3-2 Clock generator block diagram

5.3.3 System Clock

The system clock has 3 clock sources which selected by the clock generator block. The clock source switch depends on the register HCLKSEL (CLK_CLKSEL0 [2:0]). The block diagram is listed as Figure 5.3-3.

Need to notice that the system clock is provided to CPU and whole system , the clock source should be stable and never disconnected. When PLL source from XCLK, the system clock is not suggested to select PLLFOUT as its clock source.

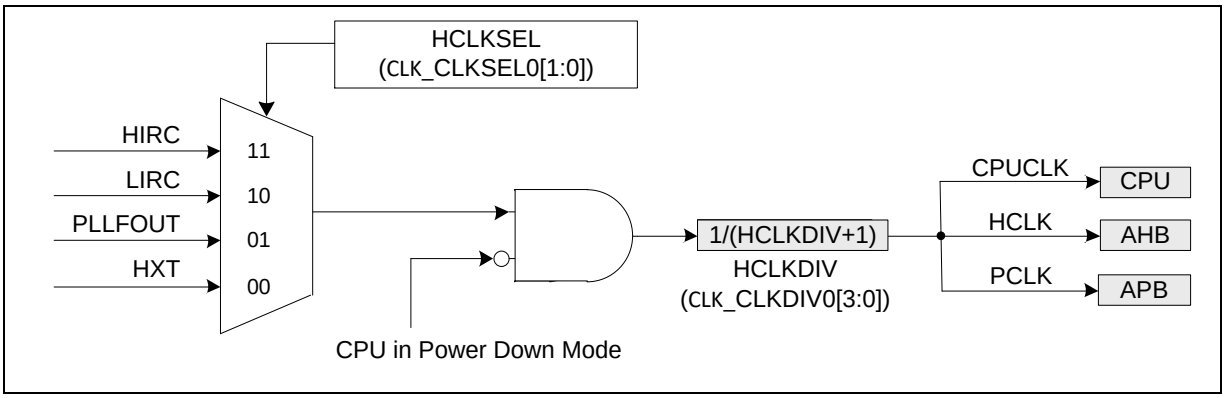


Figure 5.3-3 System Clock Block Diagram

5.3.4 Peripheral Clock

The peripheral clock has different clock source switch setting which depends on the different peripheral. Please refer to the descriptions in CLK_CLKSEL1, CLK_CLKSEL2 and CLK_APBCLK registers.

5.3.5 Power Management

The ISD91500 is equipped with a Power Management Unit (PMU) that implements a variety of power saving modes. There are four levels of power control with increasing functionality (and power consumption):

- Level0 : Deep Power Down (DPD)
- Level1 : Standby Power Down (STOP)
- Level2 : Deep Sleep
- Level3 : Sleep
- Level4 : Normal Operation

Within each of these levels there are further options to optimize power consumption.

5.3.5.1 Level0: Deep Power Down (DPD)

Deep Power Down (DPD) is the lowest power state the device can obtain. In this state there is no power provided to the logic domain and power consumption is only from the higher voltage chip supply domain. All logic state in the Cortex-M0 is lost as is contents of all RAM. All IO pins of the device are in a high impedance state. On a release from DPD the Cortex-M0 boots as if from a power-on reset. There are certain registers that can be interrogated to allow software to determine that previous state was a DPD state.

In DPD there are three ways to wake up the device:

1. A high to low transition on the Reset pin or Wakeup(GPA15) pin.
2. A timed wakeup where the LIRC oscillator is configured active and reaches a certain count.
3. A power cycle of main chip supply triggering a POR event.

To assist software in determining previous state of device before a DPD, a one-byte register is available

PD_STATE [7:0] that can be loaded with a value to be preserved before issuing a DPD request.

To configure the device for DPD the user sets the following options:

- SYSCLK->PWRCTL.SELWKTMR: Select OSC10K cycles in this register,it will trigger a wakeup event after a certain number of OSC10K clock cycles when **WK10KEN** = 0'b .
- SYSCLK->PWRCTL.WK10KEN: If set to '0' then the 10 KHz oscillator will power on in DPD, and start to count OSC10K clock cycles for wakeup.

When a WAKEUP event occurs the PMU will start the Cortex-M0 processor and execute the reset vector. The condition that generated the WAKEUP event can be interrogated by reading the registers SYSCLK->PWRCTL. TMRWKF.

To enter the DPD state the user must set the register bit SYSCLK->PWRCTL.DPDEN & SCB->SCR |= (1UL << 2) then execute a WFI or WFE instruction. Note that when debug interface is active, device will not enter DPD. Also once device enters DPD the debug interface will be inactive. It is possible that user could write code that makes it impossible to activate the debug interface and reprogram device, for instance if device re-enters DPD mode with insufficient time to allow an ICE tool to activate the SWD debug port. Especially during development it is recommended that some checks are placed in the boot sequence to prevent device going to power down.

5.3.5.2 Level1: Standby Power Down (STOP) mode.

Standby Power Down mode is the power state that some logic operation can be performed. In this mode a low power standby reference is enabled that supplies power to the logic. STOP mode is equivalent to DEEP_SLEEP mode but with the main regulator and analog shut down.

When a wake up event occurs the PMU will start the Cortex-M0 processor and continue execution from whichever interrupt vector triggered wakeup. Software can determine whether the device woke up from STOP by interrogating the register bit SYSCLK->PFLAG.STOPF.

To enter the STOP state the user must set the register bit SYSCLK->PWRCTL.STOPEN & SCB->SCR |= (1UL << 2) then execute a WFI or WFE instruction. Note that when debug interface is active, device will not enter STOP. Also once device enters STOP the debug interface will be inactive.

5.3.5.3 Level2: Deep Sleep mode.

The Deep Sleep mode is the power state where the Cortex-M0 and all logic state are preserved. In Deep Sleep mode the HIRC oscillator and HXT external crystal are shut down and LIRC is enabled and selected. All clocks to the Cortex-M0 core are gated eliminating dynamic power in the core. HCLK is operating at a low frequency and HIRC and HXT is not available. Deep Sleep mode is entered by setting System Control register bit 2: SCB->SCR |= (1UL << 2) and executing a WFI/WFE instruction. Software can determine whether the device woke up from Deep Sleep by interrogating the register bit SYSCLK->PFLAG.DSF.

5.3.5.4 Level3: Sleep mode.

The Sleep mode gates all clocks to the Cortex-M0 eliminating dynamic power in the core. The mode is entered by executing a WFI/WFE instruction and is released when an event occurs. Peripheral functions, including PDMA can be continued while in Sleep mode. Using this mode power consumption can be minimized while waiting for events such as a PDMA operation collecting data from the peripheral (ex: SARADC), once PDMA has finished the core can be woken up to process the data.

5.3.6 Clock Control Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
CLK Base Address: **CLK_BA = 0x5000_0200**				
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power Control Register	0x0803_X125
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Device Clock Enable Control Register	0x0000_0005
CLK_APBCLK	CLK_BA+0x08	R/W	APB Device Clock Enable Control Register	0x0000_0001
CLK_DPDFLR	CLK_BA+0x0C	R/W	DPD State Register and Flash Regulator Control	0x0000_XXXX
CLK_CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0003_001B
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xAF37_77A8
CLK_CLKDIV0	CLK_BA+0x18	R/W	Clock Divider Number Register 0	0x0018_0000
CLK_CLKDIV1	CLK_BA+0x1C	R/W	Clock Divider Number Register 1	0x0000_0000
CLK_STATUS	CLK_BA+0x20	R	Clock Status Monitor Register	0x0000_0000
CLK_PFLAG	CLK_BA+0x24	R/W	Power down Flag Register	0x0000_000X
CLK_CLKSEL2	CLK_BA+0x28	R/W	Clock Source Select Control Register 2	0x0000_3300
CLK_XCLKCTL	CLK_BA+0x2C	R/W	Clock doubler Output Control Register	0x0000_0000
CLK_PLLCTL	CLK_BA+0x30	R/W	PLL Control Register	0x0005_8430
CLK_ILDOCTL	CLK_BA+0xF4	R/W	Internal LDO Control Register	0x0000_0000

5.3.7 Clock Control Register Description

System Power Control Register (CLK_PWRCTL)

All the bits are protected. To program these bits needs an open lock sequence, write “59h”, “16h”, “88h” to register SYS_REGLCTL to un-lock these bits. Refer to the register SYS_REGLCTL at address SYS_BA + 0x100.

Register	Offset	R/W	Description	Reset Value
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power Control Register	0x0803_X125

31	30	29	28	27	26	25	24
WKTMRSTS				Reserved		TMRWKF	WKPINWKF
23	22	21	20	19	18	17	16
Reserved	SELWKTMR			FLASHEN		WK10KEN	WKPINEN
15	14	13	12	11	10	9	8
VSET			LIRCEN	Reserved	DPDEN	STOPEN	Reserved
7	6	5	4	3	2	1	0
IOFWK	HXTTBEN	HXTGAIN			HIRCEN	HXTEN	Reserved

Bits	Description	
[31:28]	**WKTMRSTS**	Read-Only. Read back of the current WAKEUP timer setting. This value is updated with SELWKTMR upon entering DPD mode.
[27:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25]	**TMRWKF**	Read Only. This flag indicates that wakeup of device was requested with TIMER count of the 10Khz oscillator. Flag is cleared when DPD mode is entered or any of the DPD bits of RSTSRC register (RSTSRC [10:8]) are cleared.
[24]	**WKPINWKF**	Read Only. This flag indicates that wakeup of device was requested with a high to low transition of the WAKEUP pin. Flag is cleared when DPD mode is entered or any of the DPD bits of RSTSRC register (RSTSRC[10:8]) are cleared.
[23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[22:20]	**SELWKTMR**	**Select WAKEUP Timer**: 000 = Time-out interval is 128 LIRC clocks (About 12.8 ms). 001 = Time-out interval is 256 LIRC clocks (About 25.6 ms). 010 = Time-out interval is 512 LIRC clocks (About 51.2 ms). 011 = Time-out interval is 1024 LIRC clocks (About 102.4ms). 100 = Time-out interval is 4096 LIRC clocks (About 409.6ms). 101 = Time-out interval is 8192 LIRC clocks (About 819.2ms). 110 = Time-out interval is 16384 LIRC clocks (About 1638.4ms). 111 = Time-out interval is 65536 LIRC clocks (About 6553.6ms).
[19:18]	**FLASHEN**	Determine whether FLASH memory enters deep power down. FLASHEN [0]= 1: flash enters deep power down upon DEEP_SLEEP FLASHEN [1]= 1: flash enters deep power down upon STOP mode. If FLASHEN is selected for a power state mode, current consumption is reduced, but a 10us wakeup time must be added to the wakeup sequence. Trade-off is wakeup time for standby power.
[17]	**WK10KEN**	Determines whether OSC10K is enabled in DPD mode. 0= Enabled in DPD 1= Disabled in DPD. **Note:** If **WK10KEN** is disabled, device cannot wake from DPD with SELWKTMR delay.
[16]	**WKPINEN**	Determines whether WAKEUP pin(PA15) is enabled in DPD mode. 0=Enabled 1=Disabled.
[15:13]	**VSET**	Adjusts the digital supply voltage. Should be left as default.
[12]	**LIRCEN**	**Internal 10kHz Oscillator Control** After reset, this bit is “0”. 0 = Internal 10 KHz oscillator Disabled. 1 = Internal 10 KHz oscillator Enabled.
[11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**DPDEN**	Deep Power Down (DPD) bit. Set to ‘1’ and issue WFI/WFE instruction to enter DPD mode.
[9]	**STOPEN**	STOP mode bit. Set to ‘1’ and issue WFI/WFE instruction to enter STOP mode.
[8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[7]	**IOFWK**	**All IO pin is enabled fast wakeup in STOP/DeepSleep mode**. When this bit set 0'b, trigger IO will delay 3 LIRC and then trigger wakeup from STOP/DeepSleep mode. When this bit set 1'b, trigger IO will wakeup from STOP/DeepSleep mode immediately. 0= Slow wakeup 1= Fast wakeup
[6]	**HXTTBEN**	**HXT Crystal TURBO Mode (Write Protect)** This is a protected register. Please refer to open lock sequence to program it. 0 = HXT Crystal TURBO mode disabled. 1 = HXT Crystal TURBO mode enabled.
[5:3]	**HXTGAIN**	**HXT Gain Control Bit** This is a protected register. Please refer to open lock sequence to program it. Gain control is used to enlarge the gain of crystal to make sure crystal work normally. If gain control is enabled, crystal will consume more power than gain control off. 001 = HXT frequency is from 4 MHz to 8 MHz. 010= HXT frequency is from 8 MHz to 12 MHz. 011 = HXT frequency is from 12 MHz to 16 MHz. 100= HXT frequency is higher than 16 MHz. Others = Reserved
[2]	**HIRCEN**	**Internal high speed RC Oscillator Control** After reset, this bit is "1". 0 = Internal high speed oscillation Disabled. 1 = Internal high speed oscillation Enabled.
[1]	**HXTEN**	**External high speed Crystal Oscillator Control** After reset, this bit is "0". 0 = External high speed crystal oscillation Disabled. 1 = External high speed crystal oscillation Enabled.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

AHB Device Clock Enable Control Register (CLK_AHBCLK)

These register bits are used to enable/disable the clock source for AMBA, AHB (Advanced High-Performance Bus) blocks and peripherals.

Register	Offset	R/W	Description	Reset Value
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Device Clock Enable Control Register	0x0000_0005

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**ISPCKEN**	**Flash ISP Controller Clock Enable Control.** The Flash ISP engine clock always is from 49 MHz RC oscillator. 0 = Flash ISP engine clock Disabled. 1 = Flash ISP engine clock Enabled.
[1]	**CPDCKEN**	**Companding Clock Enable Control** 0 = Companding engine clock Disabled. 1 = Companding engine clock Enabled.
[0]	**PDMACKEN**	**PDMA Clock Enable Control** 0 = PDMA engine clock Disabled. 1 = PDMA engine clock Enabled.

APB Device Clock Enable Control Register (CLK_APBCLK)

These register bits are used to enable/disable clocks for APB (Advanced Peripheral Bus) peripherals. To enable the clocks write '1' to the appropriate bit. To reduce power consumption and disable the peripheral, write '0' to the appropriate bit.

Register	Offset	R/W	Description	Reset Value
CLK_APBCLK	CLK_BA+0x08	R/W	APB Device Clock Enable Control Register	0x0000_0001

31	30	29	28	27	26	25	24
ANAEN	SDADCEN	DACEN	SARADCEN	Reserved			USBEN
23	22	21	20	19	18	17	16
Reserved		PWM1EN	PWM0EN	Reserved	BIQEN	UART1EN	UART0EN
15	14	13	12	11	10	9	8
Reserved		I2S0EN	SPI0EN	SPI1EN	Reserved	I2C1EN	I2C0EN
7	6	5	4	3	2	1	0
Reserved			TMR2EN	TMR1EN	TMR0EN	Reserved	WDTEN

Bits	Description	
[31]	**ANAEN**	**Analog Block Clock Enable Control** 0 = Analog block clock Disabled 1 = Analog block clock Enabled
[30]	**SDADCEN**	**SDADC Clock Enable Control** 0 = SDADC clock Disabled 1 = SDADC clock Enabled
[29]	**DACEN**	**DAC Clock Enable Control** 0 = DAC clock Disabled. 1 = DAC clock Enabled.
[28]	**SARADCEN**	**Analog-Digital-Converter (SARADC) Clock Enable Control** 0 = SARADC clock Disabled. 1 = SARADC clock Enabled.
[27:25]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**USBEN**	**USB Clock Enable Control** 0 = USB clock Disabled. 1 = USB clock Enabled.

[23:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**PWM1EN**	**PWM1 Block Clock Enable Control** 0 = PWM1 clock Disabled. 1 = PWM1 clock Enabled.
[20]	**PWM0EN**	**PWM0 Block Clock Enable Control** 0 = PWM0 clock Disabled. 1 = PWM0 clock Enabled.
[18]	**BIQEN**	**Biquad Filter(BIQ) Block Clock Enable Control** 0 = BIQ clock Disabled. 1 = BIQ clock Enabled.
[17]	**UART1EN**	**UART1 Block Clock Enable Control** 0 = UART1 clock Disabled. 1 = UART1 clock Enabled.
[16]	**UART0EN**	**UART0 Block Clock Enable Control** 0 = UART0 clock Disabled. 1 = UART0 clock Enabled.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**I2S0EN**	**I2S0 Clock Enable Control** 0 = I2S0 clock Disabled. 1 = I2S0 clock Enabled.
[12]	**SPI0EN**	**SPI0 Clock Enable Control** 0 = SPI0 clock Disabled. 1 = SPI0 clock Enabled.
[11]	**SPI1EN**	**SPI1 Clock Enable Control** 0 = SPI1 clock Disabled. 1 = SPI1 clock Enabled.
[10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**I2C1EN**	**I2C1 Clock Enable Control** 0 = I2C1 clock Disabled. 1 = I2C1 clock Enabled.
[8]	**I2C0EN**	**I2C0 Clock Enable Control** 0 = I2C0 clock Disabled. 1 = I2C0 clock Enabled.
[7:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[4]	**TMR2EN**	**Timer2 Clock Enable Control** 0 = Timer2 clock Disabled. 1 = Timer2 clock Enabled.
[3]	**TMR1EN**	**Timer1 Clock Enable Control** 0 = Timer1 clock Disabled. 1 = Timer1 clock Enabled.
[2]	**TMR0EN**	**Timer0 Clock Enable Control** 0 = Timer0 clock Disabled. 1 = Timer0 clock Enabled.
[1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**WDTEN**	**Watchdog Clock Enable Control** This bit is the protected bit. To program this bit needs an open lock sequence, write “59h”, “16h”, “88h” to register SYS_REGLCTL to un-lock this bit. Refer to the register SYS_REGLCTL at address SYS_BA+0x100. 0 = WDT clock Disabled. 1 = WDT clock Enabled.

DPD State Register and Flash Regulator Control (CLK_DPDFLR)

Register	Offset	R/W	Description	Reset Value
CLK_DPDFLR	CLK_BA+0x0C	R/W	DPD State Register and Flash Regulator Control	0x0000_XXXX

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:8]	**PD_STATE_RB**	Current values of PD_STATE register.
[7:0]	**PD_STATE**	An 8bit register that is preserved when DPD (Deep Power Down) state is entered and after wakeup is available by reading PD_STATE_RB.

Clock Source Select Control Register 0 (CLK_CLKSEL0)

These bits are protected bits. To program these bits needs an open lock sequence, write “59h”, “16h”, “88h” to register SYS_REGLCTL to un-lock these bits. Refer to the register SYS_REGLCTL at address SYS_BA+0x100.

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0003_001B

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**FCLK_MUX_STATE**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OSCFSEL		**Reserved**	**STICKSEL**		**Reserved**	**HCLKSEL**	

Bits	Description	
[31:19]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[18:16]	**FCLK_MUX_STATE**	These register state shows the current HCLK is from which source clock 000 = clock source from HXT 001 = clock source from PLLFOUT 010 = clock source from LIRC 011 = clock source from HIRC Others reserved.
[15:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:6]	**OSCFSEL**	**HIRC Frequency Selection register** These bits are protected, to write to bits first perform the unlock sequence (see Register Lock Control Register (SYS_REGLCTL)) 00 = Trim for 49.152MHz@VCC=3.3V selected. 01 = Trim for 48MHz@VCC=3.3V selected. 10= Trim for 49.152MHz@VCC=1.8V selected. 11= **Reserved don’t use.**
[5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[4:3]	**STICKSEL**	**SYS_TICK Clock Source Select** 00 = clock source from HXT 01 = clock source from HXT/2 10 = clock source from HCLK/2 11 = clock source from HIRC/2 **Note:** 1. When power on, HIRC is selected as HCLK clock source. 2. Before clock switch, the related clock sources (pre-select and new-select) must be turned on. 3. SysTick clock source must less than or equal to HCLK/2.
[2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**HCLKSEL**	**HCLK Clock Source Select** 00 = clock source from HXT 01 = clock source from PLLFOUT 10 = clock source from LIRC 11 = clock source from HIRC **Note:** 1. When power on, HIRC is selected as HCLK clock source. 2. Before clock switch, the related clock sources (pre-select and new-select) must be turned on.

Clock Source Select Control Register 1 (CLK_CLKSEL1)

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xAF37_77A8

31	30	29	28	27	26	25	24
PWM1SEL		PWM0SEL		UART1SEL		UART0SEL	
23	22	21	20	19	18	17	16
Reserved	I2S0SEL			Reserved	TMR2SEL		
15	14	13	12	11	10	9	8
Reserved	TMR1SEL			Reserved	TMR0SEL		
7	6	5	4	3	2	1	0
Reserved				SARADCSEL		Reserved	WDTSEL

Bits	Description	
[31:30]	**PWM1SEL**	**PWM Timer Clock Source Select** 00 = Clock source from HXT. 01 = Clock source from PLLFOUT. 10 = Clock source from PCLK. 11 = Clock source from HIRC.
[29:28]	**PWM0SEL**	**PWM Timer Clock Source Select** 00 = Clock source from HXT. 01 = Clock source from PLLFOUT. 10 = Clock source from PCLK. 11 = Clock source from HIRC.
[27:26]	**UART1SEL**	**UART1 Clock Source Select** 00 = Clock source from HXT. 01 = Clock source from PLLFOUT. 10 = Clock source from HIRC. 11 = Clock source from HIRC.
[25:24]	**UART0SEL**	**UART0 Clock Source Select** 00 = Clock source from HXT. 01 = Clock source from PLLFOUT. 10 = Clock source from HIRC. 11 = Clock source from HIRC.

[23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[22:20]	**I2S0SEL**	**I2S0 Clock Source Select** 000 = Clock source from HXT. 001 = Clock source from PLLFOUT. 010 = Clock source from PCLK. 011 = Clock source from HIRC. 100 = Clock source from MCLKI. 101 = Clock source from XCLK. Others = Equivalent with “011”.
[19]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[18:16]	**TMR2SEL**	**Timer2 Clock Source Select** 000 = Clock source from HXT. 001 = Clock source from PCLK. 010 = Clock source from External Trigger. 011 = Clock source from LIRC. 100 = Clock source from HIRC. Others = Equivalent with “100”.
[15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14:12]	**TMR1SEL**	**Timer1 Clock Source Select** 000 = Clock source from HXT. 001 = Clock source from PCLK. 010 = Clock source from External Trigger. 011 = Clock source from LIRC. 100 = Clock source from HIRC. Others = Equivalent with “100”.
[11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:8]	**TMR0SEL**	**Timer0 Clock Source Select** 000 = Clock source from HXT. 001 = Clock source from PCLK. 010 = Clock source from External Trigger. 011 = Clock source from LIRC. 100 = Clock source from HIRC. Others = Equivalent with “100”.
[7:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[3:2]	**SARADCSEL**	**SARADC Clock Source Select** 00 = Clock source from HXT. 01 = Clock source from PLLFOUT. 10 = Clock source from PCLK. 11 = Clock source from HIRC.
[1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**WDTSEL**	**Watchdog Timer Clock Source Selection (Write Protect)** These bits are protected bits. To program these bits needs an open lock sequence, write “59h”, “16h”, “88h” to SYS_REGLCTL to un-lock these bits. Refer to the register SYS_REGLCTL at address SYS_BA+0x100.. 0= Clock source from LIRC. 1= Clock source from HCLK/2048.

Note: When user need to switch the IP clock source , user needs to make sure the current source clock and target clock both enable and stable.

Clock Divider Register0 (CLK_CLKDIV0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV0	CLK_BA+0x18	R/W	Clock Divider Number Register 0	0x0018_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	SARADCDIV						
15	14	13	12	11	10	9	8
USBDIV				UART1DIV			
7	6	5	4	3	2	1	0
UART0DIV				HCLKDIV			

Bits	Description	
[31:23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[22:16]	**SARADCDIV**	**SARADC Clock Divide Number From ADC Clock Source** The SARADC clock frequency SARADCLK = (SARADC clock source frequency) / (SARADCDIV + 1).
[15:12]	**USBDIV**	**USB Clock Divide Number From PLL Clock** USB clock frequency = (PLL frequency) / (USBDIV + 1).
[11:8]	**UART1DIV**	**UART1 Clock Divide Number From UART1 Clock Source** The **UART1** clock frequency = (**UART1** clock source frequency) / (**UART1**DIV + 1). **Note:** UART1 engine clock must smaller or equal to PCLK.
[7:4]	**UART0DIV**	**UART0 Clock Divide Number From UART0 Clock Source** The **UART0** clock frequency = (**UART0** clock source frequency) / (**UART0**DIV + 1). **Note:** UART0 engine clock must smaller or equal to PCLK.
[3:0]	**HCLKDIV**	**HCLK Clock Divide Number From HCLK Clock Source** The HCLK clock frequency = (HCLK clock source frequency) / (HCLKDIV + 1).

Clock Divider Register1 (CLK_CLKDIV1)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV1	CLK_BA+0x1C	R/W	Clock Divider Number Register 1	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:16]	**SDADCDIV**	**SDADC Clock Divide Number From SDADC Clock Source** The SDADC clock frequency = (SDADC clock source frequency) / (SDADCDIV + 1).
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**BIQDIV**	**BIQ Clock Divide Number From HCLK** The BIQ clock frequency = HCLK / (BIQDIV + 1).
[7:0]	**DACDIV**	**DAC Clock Divide Number From DAC Clock Source** The DAC clock frequency = (DAC clock source frequency) / (DACDIV + 1).

Clock Status Monitor Register (CLK_STATUS)

Register	Offset	R/W	Description	Reset Value
CLK_STATUS	CLK_BA+0x20	R	Clock Status Monitor Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			XCLKSTB	PLLSTB	HIRCSTB	LIRCSTB	HXTSTB

Bits	Description	
[31:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**XCLKSTB**	**XCLK Clock Source Stable Flag (Read Only)** 0 = Clock doubler (XCLK) clock is not stable or disabled. 1 = Clock doubler (XCLK) clock is stable and enabled.
[3]	**PLLSTB**	**Internal PLL Clock Source Stable Flag (Read Only)** 0 = Internal PLL clock is not stable or disabled. 1 = Internal PLL clock is stable and enabled.
[2]	**HIRCSTB**	**HIRC clock source stable flag(Read only)** 0 = Internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = Internal high speed RC oscillator (HIRC) clock is stable and enabled.
[1]	**LIRCSTB**	**LIRC Clock Source Stable Flag (Read Only)** 0 = Internal low speed RC oscillator (LIRC) clock is not stable or disabled. 1 = Internal low speed RC oscillator (LIRC) clock is stable and enabled.
[0]	**HXTSTB**	**HXT Clock Source Stable Flag (Read Only)** 0 = External high speed crystal oscillator (HXT) clock is not stable or disabled. 1 = External high speed crystal oscillator (HXT) clock is stabled and enabled.

Power Down Flag Register (CLK_PFLAG)

Register	Offset	R/W	Description	Reset Value
CLK_PFLAG	CLK_BA+0x24	R/W	Power down Flag Register	0x0000_000X

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

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**STOPF**	**Stop Flag** This flag is set if core logic was stopped but not powered down. Write ‘1’ to clear flag.
[0]	**DSF**	**Deep Sleep Flag** This flag is set if core logic was placed in Deep Sleep mode. Write ‘1’ to clear flag.

Clock Source Select Control Register 2 (CLK_CLKSEL2)

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL2	CLK_BA+0x28	R/W	Clock Source Select Control Register 2	0x0000_3300

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

Bits	Description	
[31:15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14:12]	**SDADCSEL**	**SDADC Clock Source Select** 000 = Clock source from HXT. 001 = Clock source from PLLFOUT. 010 = Clock source from PCLK. 011 = Clock source from HIRC. 100 = Clock source from MCLKI. 101 = Clock source from XCLK. Others = Equivalent with “000”.
[11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:8]	**DACSEL**	**DAC Clock Source Select** 000 = Clock source from HXT. 001 = Clock source from PLLFOUT. 010 = Clock source from PCLK. 011 = Clock source from HIRC. 100 = Clock source from MCLKI. 101 = Clock source from XCLK. Others = Equivalent with “000”.
[7:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[4]	**XCLKSEL**	**Clock Doubler Source Selection** 0 = Clock source from MCLK input (MCLKI). 1 = Clock source from BCLK of I2S0 (I2S0_BCLK)
[3:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**USBSEL**	**USB Clock Source Select** 0 = Clock source from HIRC. 1 = Clock source from PLLFOUT.

Note: When user need to switch the IP clock source , user needs to make sure the current source clock and target clock both enable and stable.

Clock Doubler Control Register (CLK_XCLKCTL)

Register	Offset	R/W	Description	Reset Value
CLK_XCLKCTL	CLK_BA+0x2C	R/W	Clock doubler Output Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	XCLKFIF	XCLKFIEN	XCLKFDEN	Reserved			
7	6	5	4	3	2	1	0
Reserved		RELOCK	XCLKEN	Reserved		XCLKMUL	

Bits	Description	
[31:15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14]	**XCLKFIF**	**XCLK Clock Fail Interrupt Flag** 0 = Clock doubler clock (XCLK) clock is normal. 1 = Clock doubler clock (XCLK) stops. **Note:** Write 1 to clear the bit to 0.
[13]	**XCLKFIEN**	**XCLK Clock Fail Interrupt Enable Bit** 0 = Clock doubler clock (XCLK) fail interrupt Disabled. 1 = Clock doubler clock (XCLK) fail interrupt Enabled.
[12]	**XCLKFDEN**	**XCLK Clock Fail Detector Enable Bit** 0 = Clock doubler clock (XCLK) fail detector Disabled. 1 = Clock doubler clock (XCLK) fail detector Enabled.
[11:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**RELOCK**	**XCLK relock enable setting** When write this bit to 1'b, the XCLK will execute relock action. And this bit will auto clear to 0'b after relock finish & XCLK output clock stable.
[4]	**XCLKEN**	**XCLK Enable Bit** 0 = Clock doubler (XCLK) Disabled. 1 = Clock doubler (XCLK) Enabled.
[3:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field

		always write with reset value.
[1:0]	**XCLKMUL**	**Clock doubler Output Frequency Multiplication** 00 = Output frequency multiply by 1 (Bypass) 01 = Output frequency multiply by 2 10 = Output frequency multiply by 4 11 = Output frequency multiply by 8

PLL Control Register (CLK_PLLCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PLLCTL	CLK_BA+0x30	R/W	PLL Control Register	0x0005_8430

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
STBSEL	FTREN	Reserved	PLLSRC		OE	BP	PD
15	14	13	12	11	10	9	8
OUTDIV		Reserved	INDIV				Reserved
7	6	5	4	3	2	1	0
Reserved		FBDIV					

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23]	**STBSEL**	**PLL Stable Counter Selection (Write Protected)** 0 = PLL stable time is 1293 PLL source clock (suitable for source clock is equal to or less than 12 MHz). 1 = PLL stable time is 5044 PLL source clock (suitable for source clock is larger than 12 MHz). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[22]	**FTREN**	**Fliter Enable Control** 0 = Disable Filter 1 = Enable Filter
[21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[20:19]	**PLLSRC**	**PLL Source Clock Selection (Write Protected)** 00 = PLL source clock from external high-speed crystal oscillator (HXT). 01 = PLL source clock from clock doubler output (XCLK). 10 = Reserved. Do not use. 11 = PLL source clock from internal high-speed oscillator (HIRC). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

[18]	**OE**	**PLL OE (FOUT Enable) Pin Control (Write Protected)** 0 = PLL FOUT Enabled. 1 = PLL FOUT is fixed low. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[17]	**BP**	**PLL Bypass Control (Write Protected)** 0 = PLL is in normal mode (default). 1 = PLL clock output is same as PLL input clock FIN. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[16]	**PD**	**Power-down Mode (Write Protected)** If set the PDEN bit to 1 in CLK_PWRCTL register, the PLL will enter Power-down mode, too. 0 = PLL is in normal mode. 1 = PLL is in Power-down mode (default). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[15:14]	**OUTDIV**	**PLL Output Divider Control (Write Protected)** Refer to the formulas below the table. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12:9]	**INDIV**	**PLL Input Divider Control (Write Protected)** Refer to the formulas below the table. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[8:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:0]	**FBDIV**	**PLL Feedback Divider Control (Write Protected)** Refer to the formulas below the table. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Output Clock Frequency Formula:

$$FREF = FIN \times \frac{1}{NR}$$

$$FVCO = FIN \times \frac{NF}{NR}$$

$$FOUT = FIN \times \frac{NF}{NR} \times \frac{1}{NO}$$

where FREF is the comparison frequency for the PFD (phase frequency detector).

For proper operation in normal mode, the following constraints must be satisfied:

4 MHz ≤ FREF ≤ 8 MHz

64 MHz ≤ FVCO ≤ 100 MHz

16 MHz ≤ FOUT ≤ 50 MHz

Note : Before PLL enable , please make sure internal HIRC is also enable and stable ready.

Table 5.3-1 The symbol definition of PLL Output Frequency formula

Symbol	Description
FOUT	Output Clock Frequency
FIN	Input (Reference) Clock Frequency
NR	Input Divider INDIV=0 , NR =16 INDIV=1~16 , NR =INDIV
NF	Feedback Divider FBDIV=0 , NF =64 FBDIV=1~64, NF =FBDIV
NO	OUTDIV = “00” : NO = 1 OUTDIV = “01” : NO = 2 OUTDIV = “10” : NO = 4 OUTDIV = “11” : NO = 4

Internal LDO Control Register (CLK_ILDOCTL)

Register	Offset	R/W	Description	Reset Value
CLK_ILDOCTL	CLK_BA+0xF4	R/W	Ineternal LDO Control Register	0x0000_0000

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

Bits	Description	
[31:1]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	PD	**Internal LDO Power down control under deepsleep mode** 0 = Power on under deepsleep mode 1 = Power down under deepsleep mode

5.4 SRAM

5.4.1 Overview

ISD91500 equips with 20KB SRAM for program code and data storage. It's an AHB slave interface. It supports byte, half, and word read, and write access.

5.4.2 Block Diagram

The block diagram of SRAM Controller with rotation engine is depicted as following:

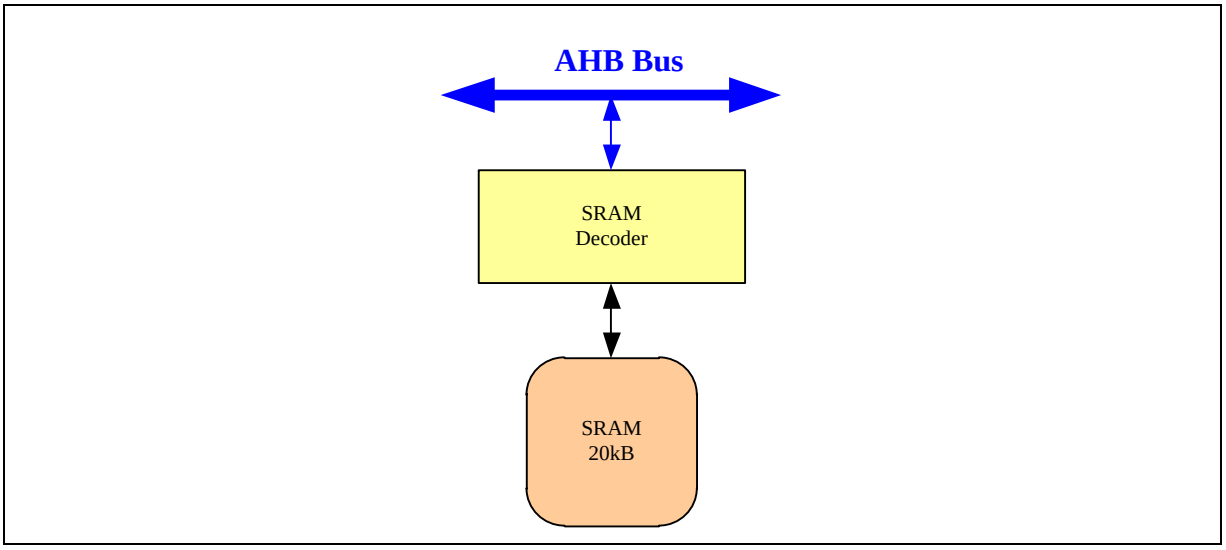


Figure 5.4-1 SRAM Controller Block Diagram

5.5 General Purpose I/O

5.5.1 Overview

There are 50 pins of General Purpose I/O shared with special feature functions. These pins are arranged in 4 groups, PA, PB, PC and PD. Each pin of the 50 pins is independent and has the corresponding register bits to control the pin mode function and data.

The I/O type of each of I/O pins can be independently software configured as input, output or open-drain mode.

In input mode, the I/O pin is in tristate (high impedance) without output drive capability. Also input mode has internal pull up option, and pull up resistance 100K/1Mohm @ 3.3V is optional.

In output mode, the I/O pin supports digital output function with source/sink current capability.

In open-drain mode, the I/O pin supports digital output function but only with sink current capability, an additional pull-up resistor is needed for driving high state.

5.5.2 Features

- Three I/O modes:
  - Push-Pull Output mode
  - Open-Drain Output mode
  - Input only with high impendence mode
- TTL/Schmitt trigger input capability
- I/O pin can be configured as interrupt source with edge/level trigger option
- Supports High Drive and High Slew Rate I/O mode
- Support I/O pin with internal pull-up resistance 100K/1Mohm@ 3.3V optional
- Enabling the pin interrupt function will also enable the wake-up function
- Support 5V tolerance

5.5.3 Block Diagram

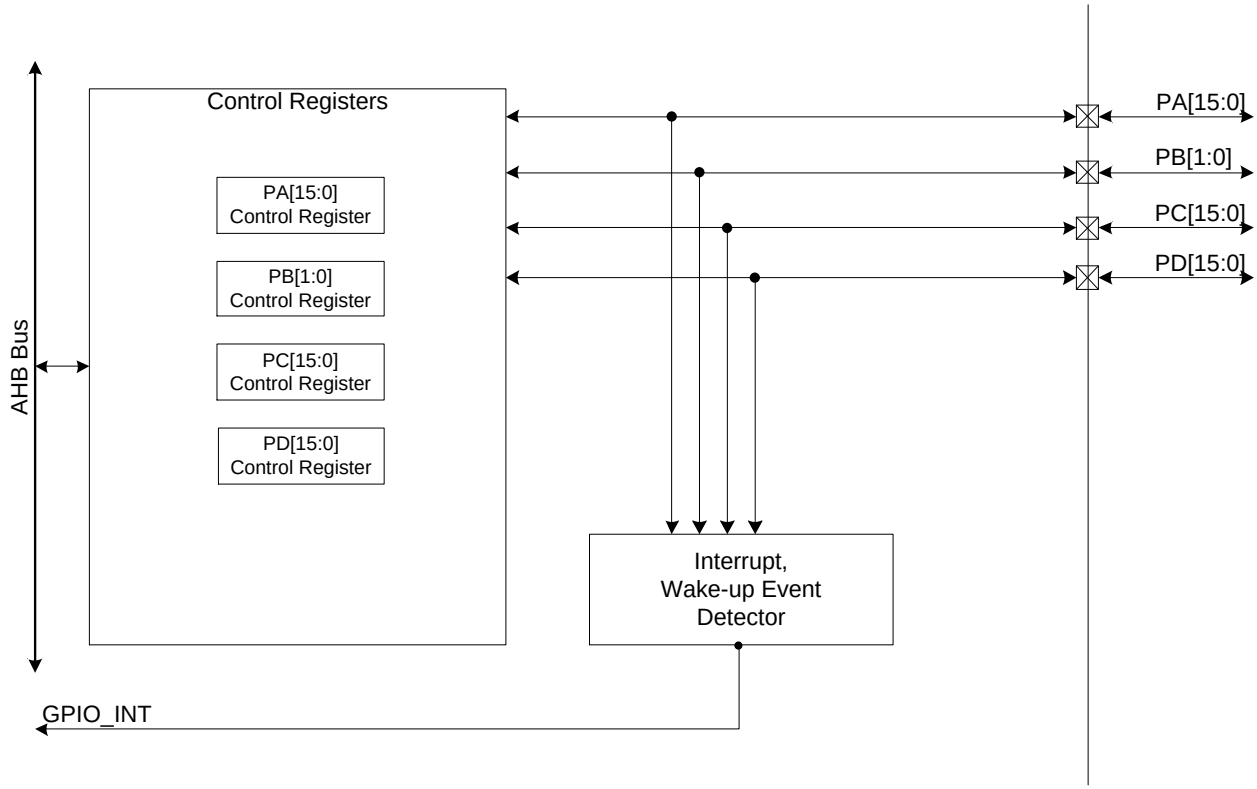


Figure 5.5-1 GPIO Controller Block Diagram

5.5.4 Functional Description

5.5.4.1 Input Mode

For Px_MODEn = 00'b the GPIO Px[n] pin is in Input mode, and the pin will be in tri-state (high impedance). The input pin's status is reflected in PX PIN[n] bit. For example if PA.0 is an input pin, the input level can be read by reading PA_PIN register, and PA_PIN[0] has the input value for PA.0 pin.

5.5.4.2 Push-pull Output Mode

For Px_MODEn = 01'b the GPIO Px[n] pin is in Push-pull Output mode, and the pin supports digital output function with source/sink current capability. As shown in Figure 5.5-2, the bit value in the corresponding DOUT (Px_DOUT[n]) is driven on the pin.

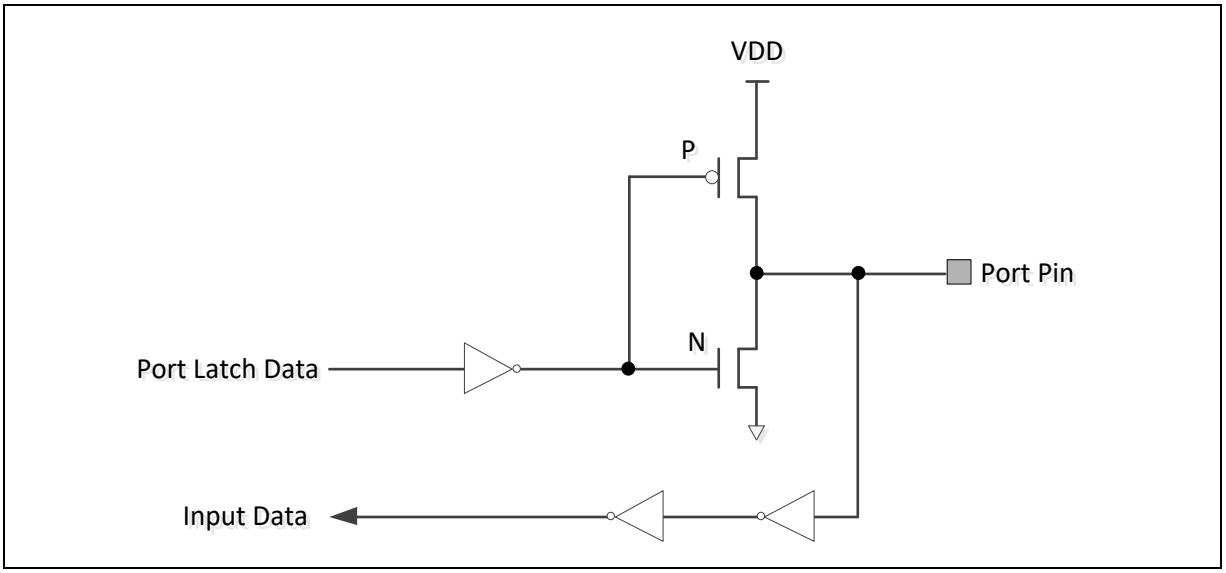


Figure 5.5-2 Push-Pull Output

5.5.4.3 Open-Drain mode explanation

For Px_MODEn = 10'b the GPIO Px[n] pin is in Open-Drain mode.

If the bit value in the corresponding bit Px_DOUT[n] is "0", the pin drive a "low" output on the pin.

If the bit value in the corresponding bit Px_DOUT[n] is "1", the pin output drive is disabled and the pin status is controlled by external pull-high resistor.

.

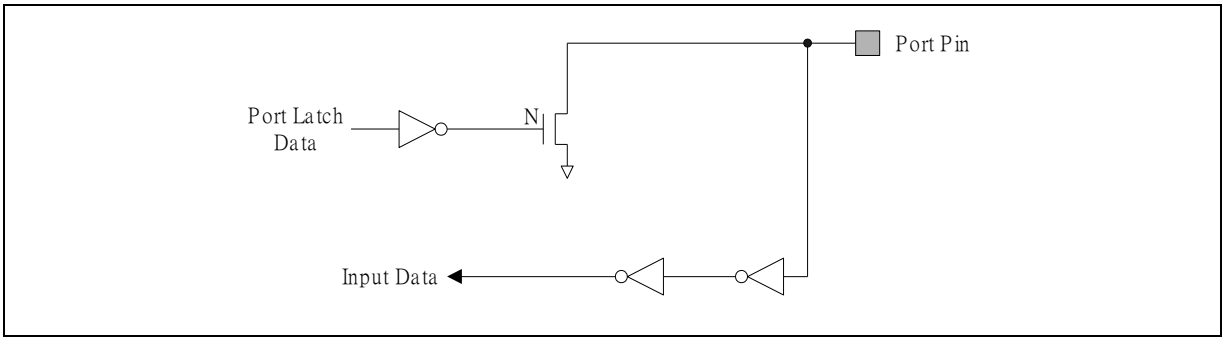


Figure 5.5-3 Open-Drain Output

5.5.5 GPIO Control Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
GPIO Base Address: **GPIO_BA = 0x5000_4000**				
PA_MODE	GPIO_BA+0x000	R/W	GPIO PA Pin I/O Mode Control	0x0000_0000
PA_DOUT	GPIO_BA+0x008	R/W	GPIO PA Data Output Value	0x0000_FFFF
PA_PIN	GPIO_BA+0x010	R	GPIO PA Pin Value	0x0000_XXXX
PA_INTTYPE	GPIO_BA+0x018	R/W	GPIO PA Interrupt Trigger Type	0x0000_0000
PA_INTEN	GPIO_BA+0x01C	R/W	GPIO PA Interrupt Enable	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	GPIO PA Interrupt Source Flag	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	GPIO PB Pin I/O Mode Control	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	GPIO PB Data Output Value	0x0000_0003
PB_PIN	GPIO_BA+0x050	R	GPIO PB Pin Value	0x0000_00XX
PB_INTTYPE	GPIO_BA+0x058	R/W	GPIO PB Interrupt Trigger Type	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	GPIO PB Interrupt Enable	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	GPIO PB Interrupt Source Flag	0x0000_0000
PC_MODE	GPIO_BA+0x080	R/W	GPIO PC Pin I/O Mode Control	0x0000_0000
PC_DOUT	GPIO_BA+0x088	R/W	GPIO PC Data Output Value	0x0000_FFFF
PC_PIN	GPIO_BA+0x090	R	GPIO PC Pin Value	0x0000_XXXX
PC_INTTYPE	GPIO_BA+0x098	R/W	GPIO PC Interrupt Trigger Type	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	GPIO PC Interrupt Enable	0x0000_0000
PC_INTSRC	GPIO_BA+0x0A0	R/W	GPIO PC Interrupt Source Flag	0x0000_0000
PD_MODE	GPIO_BA+0x0C0	R/W	GPIO PD Pin I/O Mode Control	0x0000_0000
PD_DOUT	GPIO_BA+0x0C8	R/W	GPIO PD Data Output Value	0x0000_FFFF
PD_PIN	GPIO_BA+0x0D0	R	GPIO PD Pin Value	0x0000_XXXX
PD_INTTYPE	GPIO_BA+0x0D8	R/W	GPIO PD Interrupt Trigger Type	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	GPIO PD Interrupt Enable	0x0000_0000
PD_INTSRC	GPIO_BA+0x0E0	R/W	GPIO PD Interrupt Source Flag	0x0000_0000
PAn_PDIO **n=0,1..15**	GPIO_BA+0x800+(0x04*n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X

PBn_PDIO n=0,1..11	GPIO_BA+0x840+(0x04*n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO n=0,1..15	GPIO_BA+0x880+(0x04*n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO n=0,1..15	GPIO_BA+0x8C0+(0x04*n)	R/W	GPIO PD.n Pin Data Input/Output Register	0x0000_000X

5.5.6 GPIO Control Register Description

GPIO Port [A/B/C/D] I/O Mode Control (Px_MODE)

Register	Offset	R/W	Description	Reset Value
PA_MODE	GPIO_BA+0x000	R/W	GPIO PA Pin I/O Mode Control	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	GPIO PB Pin I/O Mode Control	0x0000_0000
PC_MODE	GPIO_BA+0x080	R/W	GPIO PC Pin I/O Mode Control	0x0000_0000
PD_MODE	GPIO_BA+0x0C0	R/W	GPIO PD Pin I/O Mode Control	0x0000_0000

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

Bits	Description	
[2n+1 :2n] n=0,1..15	**MODEn**	**Port [A/B/C/D] Pin[N] I/O Mode Control** Each GPIO Px pin has four modes: 00 = GPIO Px[n] pin is in INPUT mode. 01 = GPIO Px[n] pin is in OUTPUT mode. 10 = GPIO Px[n] pin is in Open-Drain mode. 11 = GPIO Px[n] pin is in INPUT with internal PULLUP resister mode. **Note:** PB_MODE [31:4] are reserved to 0.

- **GPIO Port [A/B/C/D] Data Output Value (Px_DOUT)**

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	GPIO PA Data Output Value	0x0000_FFFF
PB_DOUT	GPIO_BA+0x048	R/W	GPIO PB Data Output Value	0x0000_0003
PC_DOUT	GPIO_BA+0x088	R/W	GPIO PC Data Output Value	0x0000_FFFF
PD_DOUT	GPIO_BA+0x0C8	R/W	GPIO PD Data Output Value	0x0000_FFFF

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**DOUT**	**Port [A/B/C/D] Pin[N] Output Value** Each of these bits controls the status of a GPIO pin when the GPIO pin is configured as output or open-drain mode. 0 = GPIO port [A/B/C/D] Pin[n] will drive Low if the corresponding output mode bit is set. 1 = GPIO port [A/B/C/D] Pin[n] will drive High if the corresponding output mode bit is set. **Note:** PB_DOUT [15:2] are reserved to 0.

GPIO Port [A/B/C/D] Pin Value (Px_PIN)

Register	Offset	R/W	Description	Reset Value
PA_PIN	GPIO_BA+0x010	R	GPIO PA Pin Value	0x0000_XXXX
PB_PIN	GPIO_BA+0x050	R	GPIO PB Pin Value	0x0000_00XX
PC_PIN	GPIO_BA+0x090	R	GPIO PC Pin Value	0x0000_XXXX
PD_PIN	GPIO_BA+0x0D0	R	GPIO PD Pin Value	0x0000_XXXX

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**PIN[n]**	**Port [A/B/C/D] Pin[N] Pin Values** Each bit of the register reflects the actual status of the respective Px.n pin. If the bit is 1, it indicates the corresponding pin status is high; else the pin status is low. **Note:** PB_PIN [15:2] are reserved to 0.

GPIO Port [A/B/C/D] Interrupt Mode Control (Px_INTTYPE)

Register	Offset	R/W	Description	Reset Value
PA_INTTYPE	GPIO_BA+0x018	R/W	GPIO PA Interrupt Trigger Type	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	GPIO PB Interrupt Trigger Type	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	GPIO PC Interrupt Trigger Type	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	GPIO PD Interrupt Trigger Type	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**TYPE[n]**	**Port [A/B/C/D] Pin[N] Edge Or Level Detection Interrupt Trigger Type Control** TYPE[n] is used to control whether the interrupt mode is level triggered or edge triggered. If the interrupt mode is level triggered, the input source is sampled by one HCLK clock to generate the interrupt 0 = Edge triggered interrupt. 1 = Level triggered interrupt. **Note 1:** If level triggered interrupt is selected, then only one level can be selected in the Px_INTEN register. If both levels are set, the setting is ignored and no interrupt will occur **Note 2:** PB_INTTYPE [15:2] are reserved to 0.

GPIO Port [A/B/C/D] Interrupt Enable Control (Px_INTEN)

Register	Offset	R/W	Description	Reset Value
PA_INTEN	GPIO_BA+0x01C	R/W	GPIO PA Interrupt Enable	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	GPIO PB Interrupt Enable	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	GPIO PC Interrupt Enable	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	GPIO PD Interrupt Enable	0x0000_0000

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

Bits	Description	
[n+16] n=0,1..15	**RHIEN[n]**	**Port [A/B/C/D] Interrupt Enable By Input Rising Edge Or Input Level High** RHIEN[n] is used to enable the rising/high-level interrupt for each of the corresponding input Px.n pin. To set “1” also enables the pin wake-up function. When setting the RHIEN[n] (Px_INTEN[n+16]) bit to 1 : If the interrupt is configured as level trigger mode (TYPE[n] is set to 1), one interrupt will occur while the input Px.n state is at high level. If the interrupt is configured as edge trigger mode (TYPE[n] is set to 0), one interrupt will occur while he input Px.n state changes from low to high. 0 = Disable Px.n for low-to-high or level-high interrupt. 1 = Enable Px.n for low-to-high or level-high interrupt. **Note:** PB_RHIEN [15:2] are reserved to 0.

[n] n=0,1..15	**FLIEN[n]**	**Port [A/B] Interrupt Enable By Input Falling Edge Or Input Level Low** FLIEN[n] is used to enable the falling/low-level interrupt for each of the corresponding input Px.n pin. To set “1” also enables the pin wake-up function. When setting the FLIEN[n] (Px_INTEN[n]) bit to 1 : If the interrupt is configured as level trigger mode (TYPE[n] is set to 1), one interrupt will occur while the input Px.n state is at low level. If the interrupt is configured as edge trigger mode (TYPE[n] is set to 0), one interrupt will occur while he input Px.n state changes from high to low. 0 = Disable Px.n for low-level or high-to-low interrupt. 1 = Enable Px.n for low-level or high-to-low interrupt. **Note:** PB_FLIEN [15:2] are reserved to 0.

GPIO Port [A/B/C/D] Interrupt Source Flag(Px_INTSRC)

Register	Offset	R/W	Description	Reset Value
PA_INTSRC	GPIO_BA+0x020	R/W	GPIO PA Interrupt Source Flag	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	GPIO PB Interrupt Source Flag	0x0000_0000
PC_INTSRC	GPIO_BA+0x0A0	R/W	GPIO PC Interrupt Source Flag	0x0000_0000
PD_INTSRC	GPIO_BA+0x0E0	R/W	GPIO PD Interrupt Source Flag	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**INTSRC[n]**	**Port [A/B/C/D] Interrupt Source Flag** Read operation: 0 = No interrupt from Px.n. 1 = Px.n generated an interrupt. Write operation: 0 = No action. 1 = Clear the corresponding pending interrupt. **Note:** PB_INTSRC [15:2] are reserved to 0.

GPIO Px.n Pin Data Input/Output Register (Pxn_PDIO)

Register	Offset	R/W	Description	Reset Value
PAn_PDIO n=0,1..15	GPIO_BA+0x800+(0x04*n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO n=0,1..4	GPIO_BA+0x840+(0x04*n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO n=0,1..15	GPIO_BA+0x880+(0x04*n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO n=0,1..15	GPIO_BA+0x8C0+(0x04*n)	R/W	GPIO PD.n Pin Data Input/Output Register	0x0000_000X

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**PDIO**	**GPIO Px.n Pin Data Input/Output** Writing this bit can control one GPIO pin output value. 0 = Corresponding GPIO pin set to low. 1 = Corresponding GPIO pin set to high. Read this register to get GPIO pin status. For example, writing PA0_PDIO will reflect the written value to bit DOUT (Px_DOUT[0]), reading PA0_PDIO will return the value of PIN (PA_PIN[0]). **Note1:** The writing operation will not be affected by register DATMSK (Px_DATMSK[n]). **Note2:** Max. n=15 for port A. C. D Max. n=1 for port B.

5.6 PWM Generator and Capture Timer

5.6.1 Overview

The ISD91500 has 2 sets of PWM timers. Each PWM timer consists of 1 prescaler, 1 clock divider, 1 clock selector with 5 input sources, one 16-bit counter, and four 16-bit comparators.

Clock divider provides PWM timer with 5 clock sources (1, 1/2, 1/4, 1/8, 1/16). PWM timer receives its own clock signal from clock divider which receives clock from 8-bit prescaler. The 16-bit counter receives clock signal from clock selector and can be used to handle one PWM period. The 16-bit comparator compares number in counter with threshold number in register loaded previously to generate PWM duty cycle.

The clock signal from clock divider is called PWM clock. Dead-Zone generator utilize PWM clock as clock source. Once Dead-Zone generator is enabled, output of two PWM comparators is blocked. Two output pin are used as Dead-Zone generator output signal to control off-chip power device. Dead-Zone generator 0 is used to control outputs of PWM channel 0 & PWM channel 1, and Dead-Zone generator 1 is used to control outputs of PWM channel 2 & PWM channel 3.

When 16-bit down counter reaches zero and interrupt function is enabled, the interrupt request is generated to inform CPU that time is up. When counter reaches zero, if counter is set as toggle mode, it is reloaded automatically and starts to generate next cycle. User can set counter as one-shot mode instead of toggle mode. If counter is set as one-shot mode, counter will stop and generate one interrupt request when it reaches zero.

The value of comparator is used for pulse width modulation. The comparator control logic changes the output level when down-counter value matches the value of compare register.

The PWM timer includes a capture channel. The Capture input and PWM output share the PWM timer. Therefore user must setup the PWM timer before turn on Capture feature. After enabling Capture feature, the Capture always latches PWM counter to CRLR register when capture input has a rising transition and latches PWM counter to CFLR register when capture input has a falling transition. Capture input interrupt is programmable by setting PWM_CAPCTL [1] (Rising latch Interrupt enable) and PWM_CAPCTL [2] (Falling latch Interrupt enable) to decide the condition of interrupt event. Whenever Capture event issues, the PWM counter will be reloaded at this moment.

There is only an interrupt from PWM to interrupt controller. PWM Timer and Capture input share the same interrupt. Therefore, PWM function and Capture function cannot be used at the same time.

5.6.2 Features

General PWM (PWM0/1) function:

- Up to 2 sets group , each set support 4 PWM channels or 2 PWM paired channels
- One 8-bit prescaler and clock divider with 5 clock input selector in each set
- Each set support one 16-bit down counters and four 16-bit comparators
- Support One-shot or Auto-reload mode
- Each set support two Dead-Zone generator

Capture timer function:

- Supports 2 Capture input channels
- Supports rising or falling capture condition
- Supports rising or falling capture interrupt

5.6.3 PWM Generator Architecture

The following figures illustrate the architecture of the PWM0/1 generator.

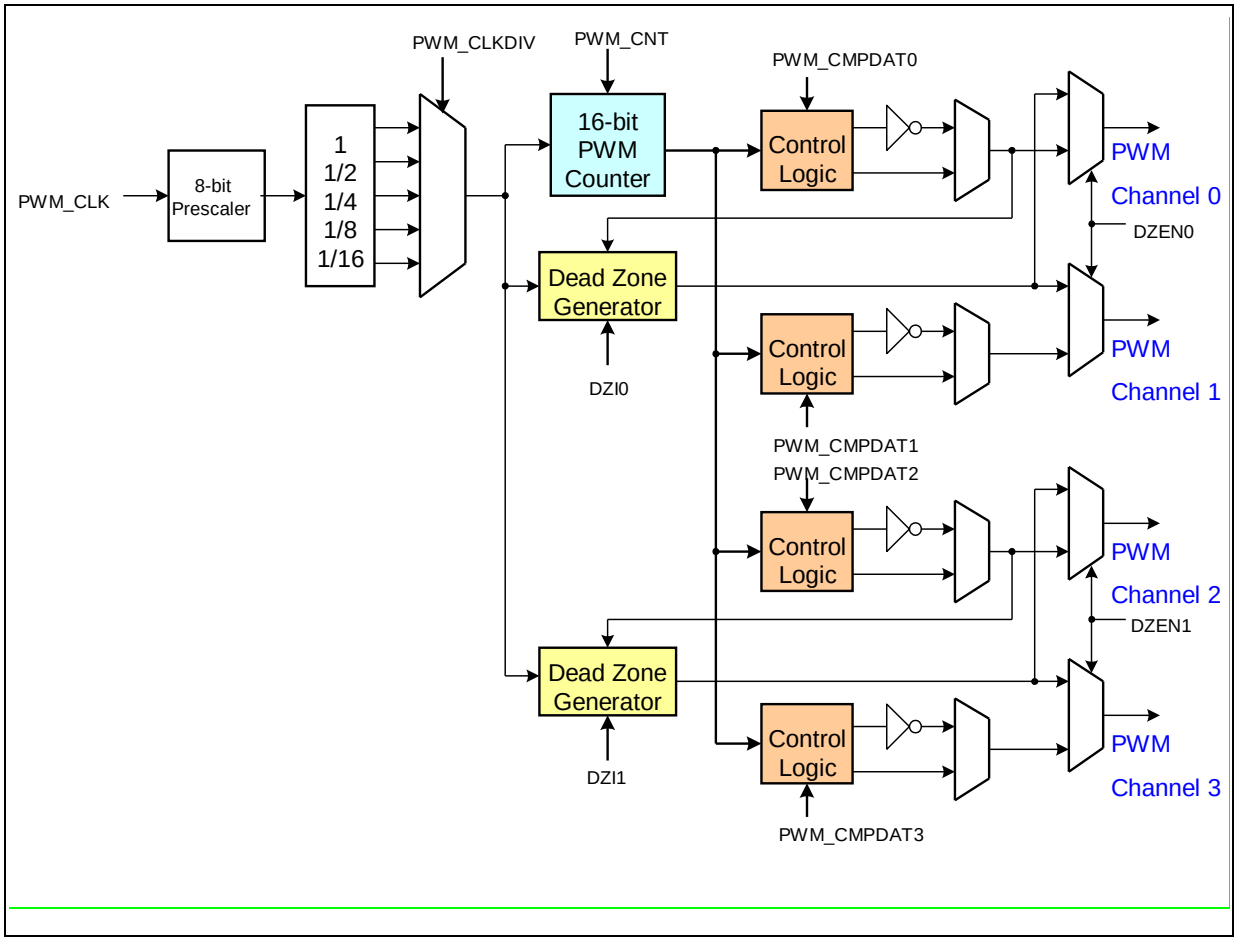


Figure 5.6-1 PWM0/1 Generator Architecture Diagram

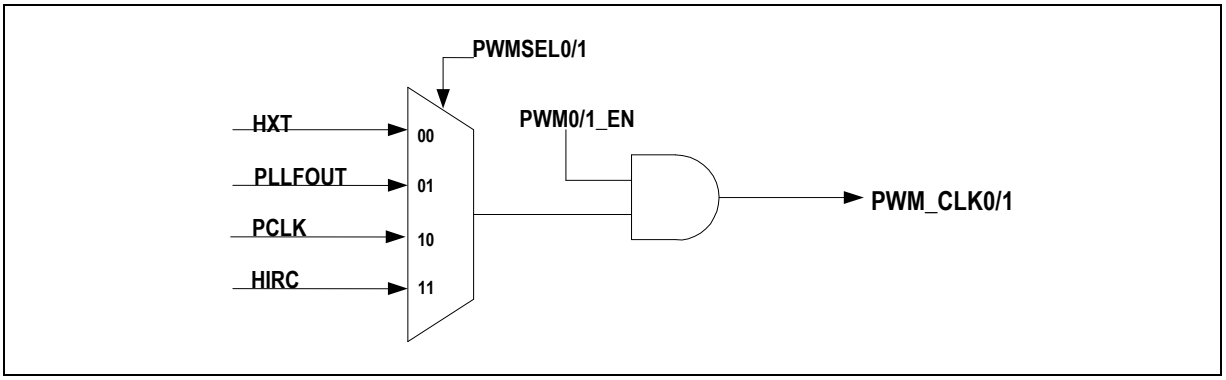


Figure 5.6-2 PWM0/1 Generator Clock Source Control

5.6.4 PWM-Timer Operation

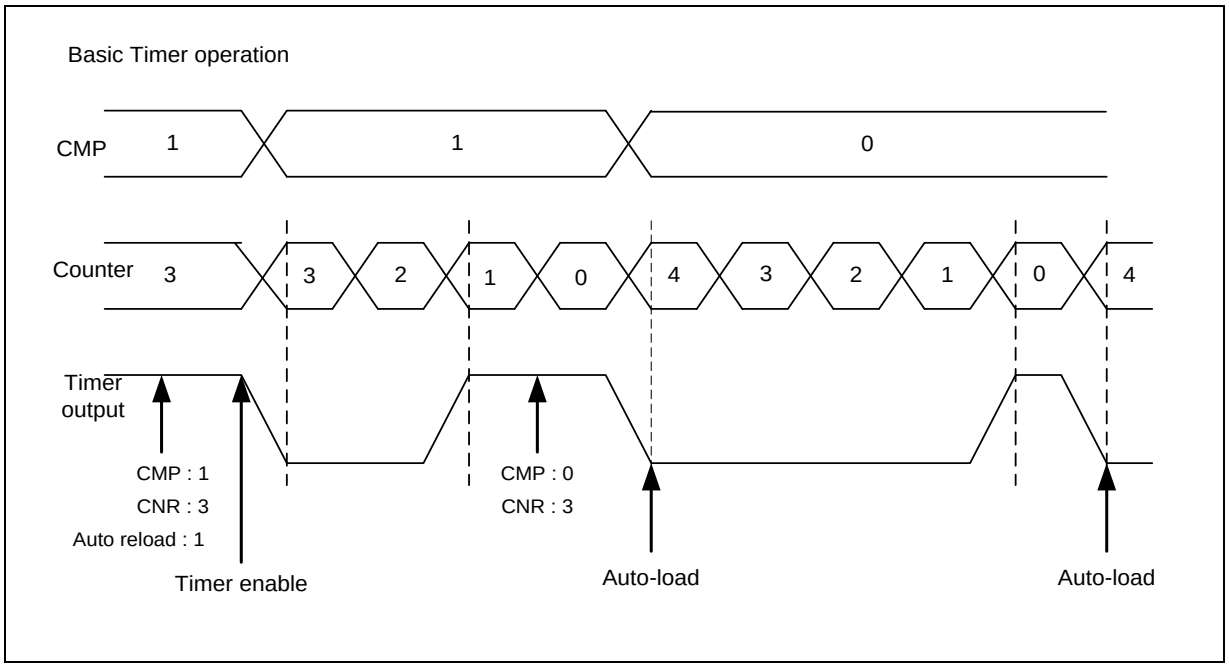


Figure 5.6-3 PWM Timer Operation Timing

5.6.5 PWM Auto-reload

The counter value can be written into PWM_PERIOD and current counter value can be read from PWM_CNT.

The auto-reload operation copies from PWM_PERIOD to down-counter when down-counter reaches zero. If PWM_PERIOD is set as zero, counter will be halt when counter counts to zero. If auto-reload bit is set as zero, counter will be stopped immediately.

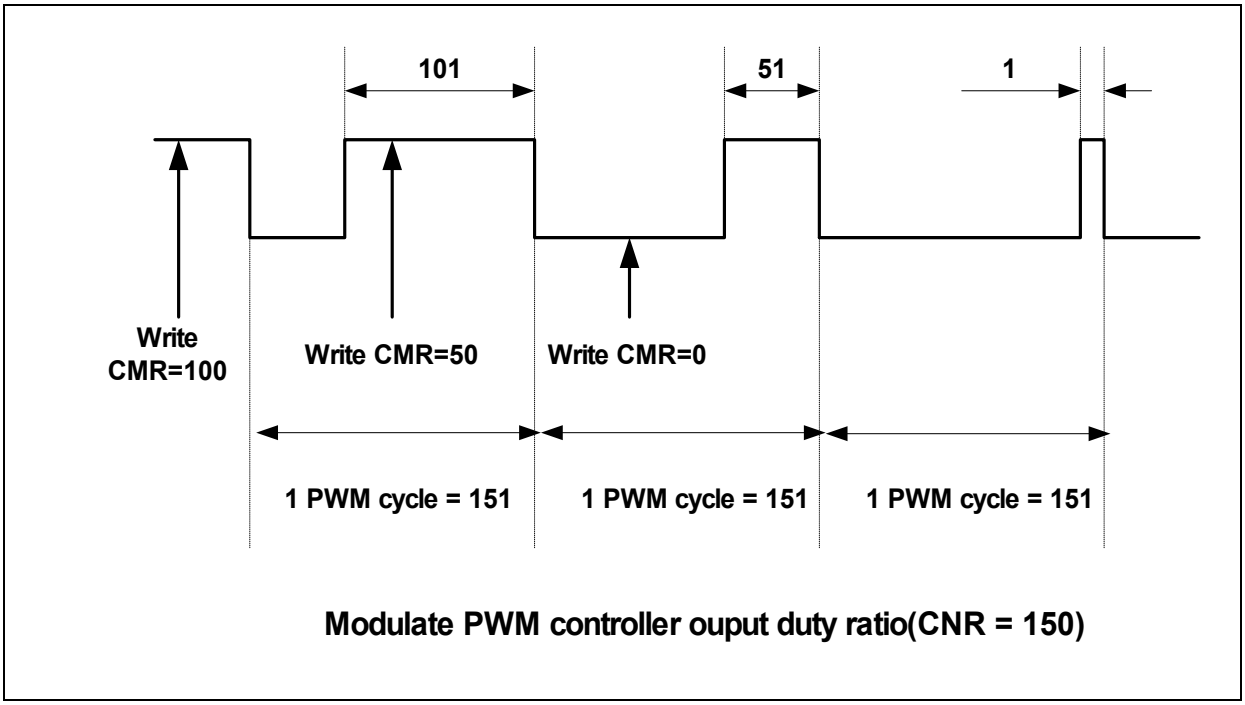


Figure 5.6-4 PWM Controller Output Duty Ratio.

5.6.6 Dead-Zone Generator

ISD91500 PWM is implemented with Dead Zone generator. They are built for power device protection. This function enables generation of a programmable time gap at the rising of PWM output waveform. User can program PWM_CLKPSC[31:24] and PWM_CLKPSC[23:16] to determine the two Dead Zone interval respectively.

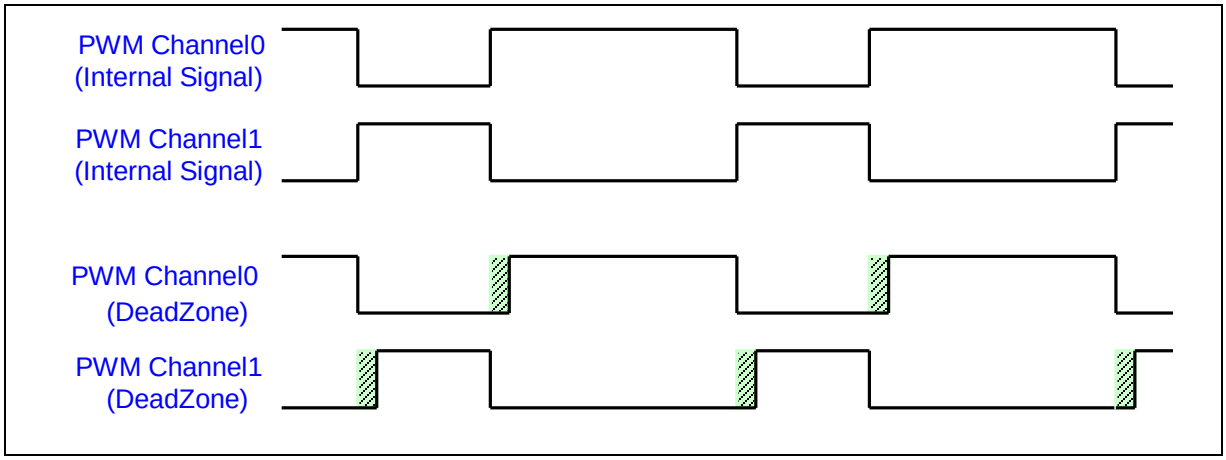


Figure 5.6-5 Dead Zone Generation Operation

5.6.7 PWM-Timer Start Procedure

The following procedure is recommended for starting a PWM generator.

1. Setup clock selector (PWM_CLKDIV)
2. Setup prescaler (PWM_CLKPSC)
3. Setup inverter on/off, dead zone generator on/off, auto-reload/one-shot mode and PWM timer off (PWM_CTL)
4. Setup comparator registers (PWM_CMPDATn) to set PWM duty cycle.
5. Setup PWM down-counter register (PWM_PERIOD) to set PWM period.
6. Setup interrupt enable register (PWM_INTEN)
7. Setup PWM output enable (PWM_PCEN)
8. Setup the corresponding GPIO pins to PWM function (SYS_GPA_MFP/ SYS_GPB_MFP)
9. Enable PWM timer start running (PWM_CTL)

5.6.8 PWM-Timer Stop Procedure

Method 1:

Set 16-bit down counter (PWM_PERIOD) as 0, and monitor PWM_CNT (current value of 16-bit down-counter). When PWM_CNT reaches to 0, disable PWM-Timer (PWM_CTL). ***(Recommended)***

Method 2:

Set 16-bit down counter (PWM_PERIOD) as 0. When interrupt request occurs, disable PWM-Timer (PWM_CTL). ***(Recommended)***

Method 3:

Disable PWM-Timer directly (PWM_CTL). ***(Not recommended)***

5.6.9 Capture Start Procedure

1. Setup clock selector (PWM_CLKDIV)
2. Setup prescaler and dead zone interval (PWM_CLKPSC)
3. Setup inverter on/off, dead zone generator on/off, auto-reload/one-shot mode and PWM timer off (PWM_CTL)
4. Setup PWM down-counter register (PWM_PERIOD)
5. Setup capture register (PWM_CAPCTL)
6. Enable PWM timer start running (PWM_CTL)

5.6.10 Capture Timer Operation

The Capture and PWM output function share the same PWM timer. The capture timer latches PWM-counter to PWM_RCAPDAT when input channel has a rising transition and latches PWM-counter to PWM_FCAPDAT when input channel has a falling transition. Capture interrupt is programmable by setting PWM_CAPCTL [1] (Rising latch Interrupt enable) and PWM_CAPCTL [2] (Falling latch Interrupt enable) to decide the condition of interrupt occurrence. Whenever the Capture module issues a capture event, the corresponding PWM counter will be reloaded with PERIOD at this moment. Note that the corresponding GPIO pin must be configured as their alternate function before Capture function is enabled.

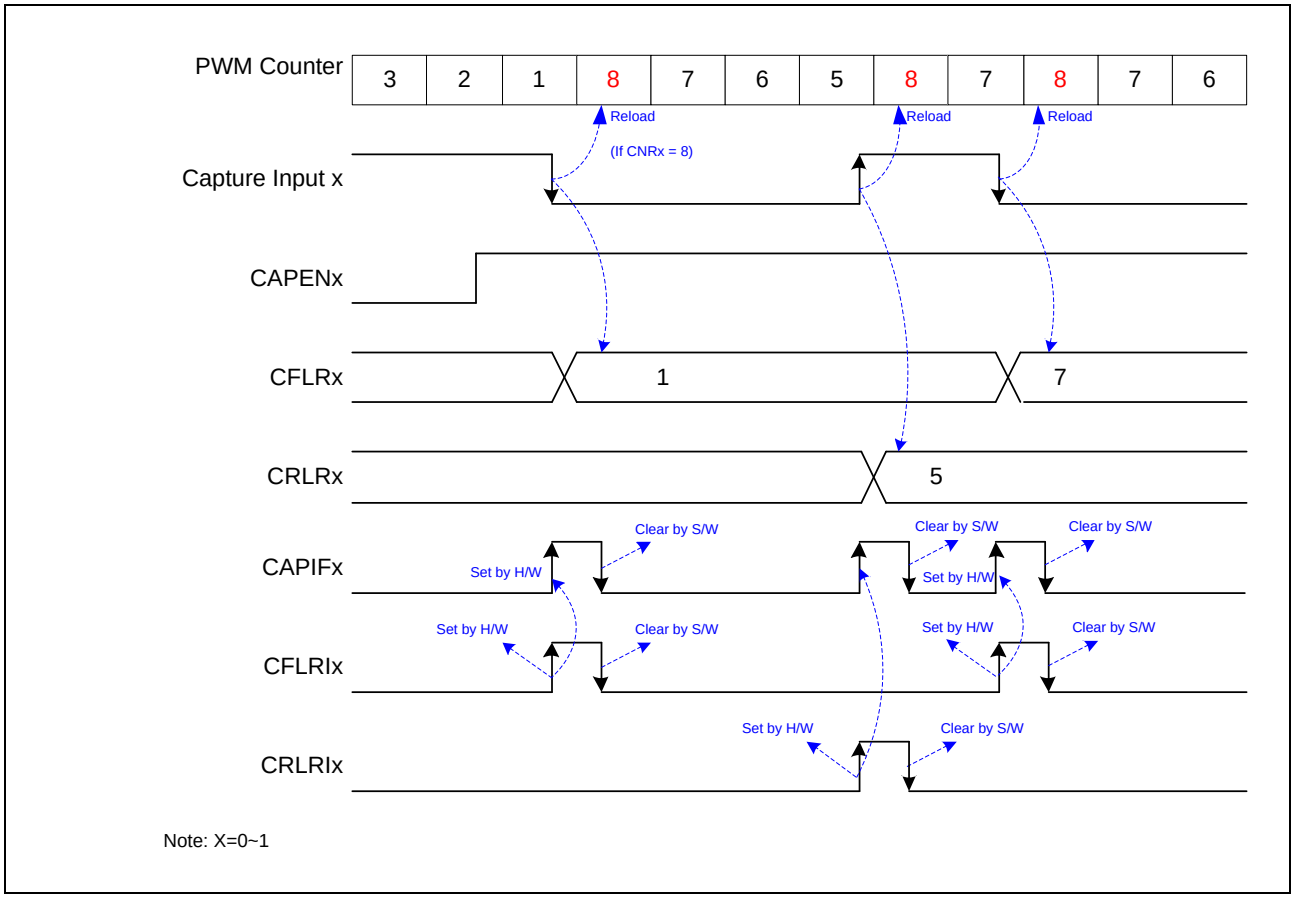


Figure 5.6-6 Capture Operation Timing

Figure 5.6-6 demonstrates the case where PERIOD = 8:

1. The PWM counter will be reloaded with PERIOD=8 when CAPIF is set from 0 to 1. .
2. The channel low pulse width is given by (PERIOD - RCAPDAT).
3. The channel high pulse width is given by (PERIOD - FCAPDAT).

5.6.11 PWM Control Register Map

R: read only, **W**: write only, **R/W**: both read and write, **C**: Only value 0 can be written

Register	Offset	R/W	Description	Reset Value
PWM Base Address: **PWMx_BA = 0x4004_0000+0x1_0000*x** **x= 0, 1**				
PWM_CLKPSC	PWMx_BA+0x000	R/W	PWM Prescaler Register	0x0000_0000
PWM_CLKDIV	PWMx_BA+0x004	R/W	PWM Clock Select Register	0x0000_0000
PWM_CTL	PWMx_BA+0x008	R/W	PWM Control Register	0x0000_0000
PWM_PERIOD	PWMx_BA+0x00C	R/W	PWM Period Register	0x0000_0000
PWM_CMPDAT0	PWMx_BA+0x010	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CNT	PWMx_BA+0x014	R	PWM Counter Register	0x0000_0000
PWM_CMPDAT1	PWMx_BA+0x01C	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2	PWMx_BA+0x028	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3	PWMx_BA+0x034	R/W	PWM Comparator Register 3	0x0000_0000
PWM_INTEN	PWMx_BA+0x040	R/W	PWM Interrupt Enable Register	0x0000_0000
PWM_INTSTS	PWMx_BA+0x044	R/W	PWM Interrupt Flag Register	0x0000_0000
PWM_CAPCTL	PWMx_BA+0x050	R/W	Capture Control Register	0x0000_0000
PWM_RCAPDAT	PWMx_BA+0x058	R	Capture Rising Latch Register	0x0000_0000
PWM_FCAPDAT	PWMx_BA+0x05C	R	Capture Falling Latch Register	0x0000_0000
PWM_PCEN	PWMx_BA+0x07C	R/W	PWM Output and Capture Input Enable Register	0x0000_0000

5.6.12 PWM Control Register Description

- **PWM Pre-Scale Register (PWM_CLKPSC)**

Register	Offset	R/W	Description	Reset Value
PWM_CLKPSC	PWMx_BA+0x000	R/W	PWM Prescaler Register	0x0000_0000

31	30	29	28	27	26	25	24
DZI1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DZI0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC							

Bits	Description	
[31:24]	**DZI1**	**Dead Zone Interval Register 1** These 8 bits determine dead zone length. The unit time of dead zone length is that from clock selector.
[23:16]	**DZI0**	**Dead Zone Interval Register 0** These 8 bits determine dead zone length. The unit time of dead zone length is that from clock selector.
[15:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**CLKPSC**	**Clock Prescaler For PWM Timer** Clock input is divided by (CLKPSC + 1) If CLKPSC = 0, then the prescaler output clock will be stopped. This implies PWM counter will also be stopped.

PWM Clock Select Register (PWM_CLKDIV)

Register	Offset	R/W	Description	Reset Value
PWM_CLKDIV	PWMx_BA+0x004	R/W	PWM Clock Select Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**CLKDIV**	**PWM Timer Clock Source Selection** Value : Input clock divided by 000 : 2 001 : 4 010 : 8 011 : 16 1xx : 1

PWM Control Register (PWM_CTL)

Register	Offset	R/W	Description	Reset Value
PWM_CTL	PWMx_BA+0x008	R/W	PWM Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**DTEN1**	**DTEN0**	**CNTMODE**	**PINV**	**Reserved**	**CNTEN**

Bits	Description	
[31:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**DTEN1**	**Dead-Zone 1 Generator Enable/Disable** 0 = Disable. 1 = Enable.
[4]	**DTEN0**	**Dead-Zone 0 Generator Enable/Disable** 0 = Disable. 1 = Enable.
[3]	**CNTMODE**	**PWM-Timer Auto-Reload/One-Shot Mode** 0 = One-Shot Mode. 1 = Auto-reload Mode.
[2]	**PINV**	**PWM-Timer Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.
[1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**CNTEN**	**PWM-Timer Enable** 0 = Stop PWM-Timer Running. 1 = Enable PWM-Timer.

PWM Period Register (PWM_PERIOD)

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD	PWMx_BA+0x00C	R/W	PWM Period Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**PERIOD**	**PWM Counter/Timer Reload Value** PERIOD determines the PWM period. **Note:** One PWM cycle width = (PERIOD + 1).

PWM Comparator Register 3-0 (PWM_CMPDAT3-0)

Register	Offset	R/W	Description	Reset Value
PWM_CMPDAT0	PWMx_BA+0x010	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1	PWMx_BA+0x01C	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2	PWMx_BA+0x028	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3	PWMx_BA+0x034	R/W	PWM Comparator Register 3	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**CMP**	**PWM Comparator Register** CMP determines the PWM duty ratio. Assumption: PWM output initial is high • CMP > = PERIOD: PWM output is always high. • CMP < PERIOD: PWM low width = (PERIOD - CMP) unit; PWM high width = (CMP+1) unit. • CMP = 0: PWM low width = (PERIOD) unit; PWM high width = 1 unit. **Note1:** Unit = one PWM clock cycle. **Note2:** Any write to CMP will take effect in next PWM cycle.

PWM Counter Register (PWM_CNT)

Register	Offset	R/W	Description	Reset Value
PWM_CNT	PWMx_BA+0x014	R	PWM Counter Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**CNT**	**PWM Counter Register** Reports the current value of the 16-bit down counter.

PWM Interrupt Enable Register (PWM_INTEN)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN	PWMx_BA+0x040	R/W	PWM Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**PIEN**	**PWM Timer Interrupt Enable** 0 = Disable. 1 = Enable.

PWM Interrupt Flag Register (PWM_INTSTS)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS	PWMx_BA+0x044	R/W	PWM Interrupt Flag Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**PIF**	**PWM Timer Interrupt Flag** Flag is set by hardware when PWM down counter reaches zero, software can clear this bit by writing ‘1’ to it.

Capture Control Register (PWM_CAPCTL)

Register	Offset	R/W	Description	Reset Value
PWM_CAPCTL	PWMx_BA+0x050	R/W	Capture Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CFLIF	CRLIF	Reserved	CAPIF	CAPEN	CFLIEN	CRLIEN	CAPINV

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**CFLIF**	**PWM_FCAPDAT Latched Indicator Bit** When input channel has a **falling** transition, PWM_FCAPDAT was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[6]	**CRLIF**	**PWM_RCAPDAT Latched Indicator Bit** When input channel has a **rising** transition, PWM_RCAPDAT was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**CAPIF**	**Capture Indication Flag** When capture input has a falling/rising transition with CFLIF = 1 or CRLIF = 1, CAPIF0 is set 1 by hardware. Software can clear this bit by writing a one to it. **Note:** If this bit is “1”(not clear by SW), PWM counter will not be reloaded when next capture event occurs.

[3]	**CAPEN**	**Capture Channel Input Transition Enable/Disable** 0 = Disable capture function. 1 = Enable capture function. When enabled, Capture function latches the PMW-counter to RCAPDAT (Rising latch) and FCAPDAT (Falling latch) registers on input edge transition. When disabled, Capture function is inactive as is interrupt.
[2]	**CFLIEN**	**Falling Latch Interrupt Enable ON/OFF** 0 = Disable falling latch interrupt. 1 = Enable falling latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.
[1]	**CRLIEN**	**Rising Latch Interrupt Enable ON/OFF** 0 = Disable rising latch interrupt. 1 = Enable rising latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.
[0]	**CAPINV**	**Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before fed to Capture timer

Capture Rising Latch Register (PWM_RCAPDAT)

Register	Offset	R/W	Description	Reset Value
PWM_RCAPDAT	PWMx_BA+0x058	R	Capture Rising Latch Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**RCAPDAT**	**Capture Rising Latch Register** In Capture mode, this register is latched with the value of the PWM counter on a rising edge of the input signal.

Capture Falling Latch Register (PWM_FCAPDAT)

Register	Offset	R/W	Description	Reset Value
PWM_FCAPDAT	PWMx_BA+0x05C	R	Capture Falling Latch Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**FCAPDAT**	**Capture Falling Latch Register** In Capture mode, this register is latched with the value of the PWM counter on a falling edge of the input signal.

PWM Output and Capture Input Enable Register (PWM_PCEN)

Register	Offset	R/W	Description	Reset Value
PWM_PCEN	PWMx_BA+0x07C	R/W	PWM Output and Capture Input Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**CAPINEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**POEN3**	**POEN2**	**POEN1**	**POEN0**

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**CAPINEN**	**Capture Input Enable Register** 0 = OFF (PA.7/PB.4 pin input disconnected from Capture block). 1 = ON (PA.7/PB.4 pin, if in PWM alternative function, will be configured as an input and fed to capture function).
[7:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3]	**POEN3**	**PWM Channel 3 Output Enable Register** 0 = Disable **PWM Channel 3** output to pin. 1 = Enable **PWM Channel 3** output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)
[2]	**POEN2**	**PWM Channel 2 Output Enable Register** 0 = Disable **PWM Channel 2**output to pin. 1 = Enable **PWM Channel 2** output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)

[1]	**POEN1**	**PWM Channel 1 Output Enable Register** 0 = Disable **PWM Channel 1** output to pin. 1 = Enable **PWM Channel 1** output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)
[0]	**POEN0**	**PWM Channel0 Output Enable Register** 0 = Disable **PWM Channel0** output to pin. 1 = Enable **PWM Channel0** output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)

5.7 Serial Peripheral Interface (SPI0/1) Controller

5.7.1 Overview

The Serial Peripheral Interface (SPI) is a synchronous serial data communication protocol which operates in full duplex mode. Devices communicate in master/slave mode with 4-wire bi-directional interface. The ISD91500 series contains two SPI controller performing a serial-to-parallel conversion of data received from an external device, and a parallel-to-serial conversion of data transmitted to an external device. The SPI controller can be set as a master with up to 2 slave select (SSB) address lines to access two slave devices; it also can be set as a slave controlled by an off-chip master device.

In addition the SPI interface supports Dual and Quad IO as is common on serial flash memories, where data is transferred 2 or 4 bits per clock period.

5.7.2 Features

- Supports master or slave mode operation.
- Supports one or two channels of serial data.
- 8 word FIFO on transmit and receive.
- MSB or LSB first transfer.
- 2 device/slave select lines in master mode, single device/slave select line in slave mode.
- Configurable bit length of a transaction word from 1 to 32-bit
- Supports MSB first or LSB first transfer sequence
- Supports Byte Reorder function
- Support 3-wire mode in slave mode
- Support Dual/Quad I/O mode
- Byte or word Sleep Suspend Mode.
- PDMA access support.

5.7.3 SPI Block Diagram

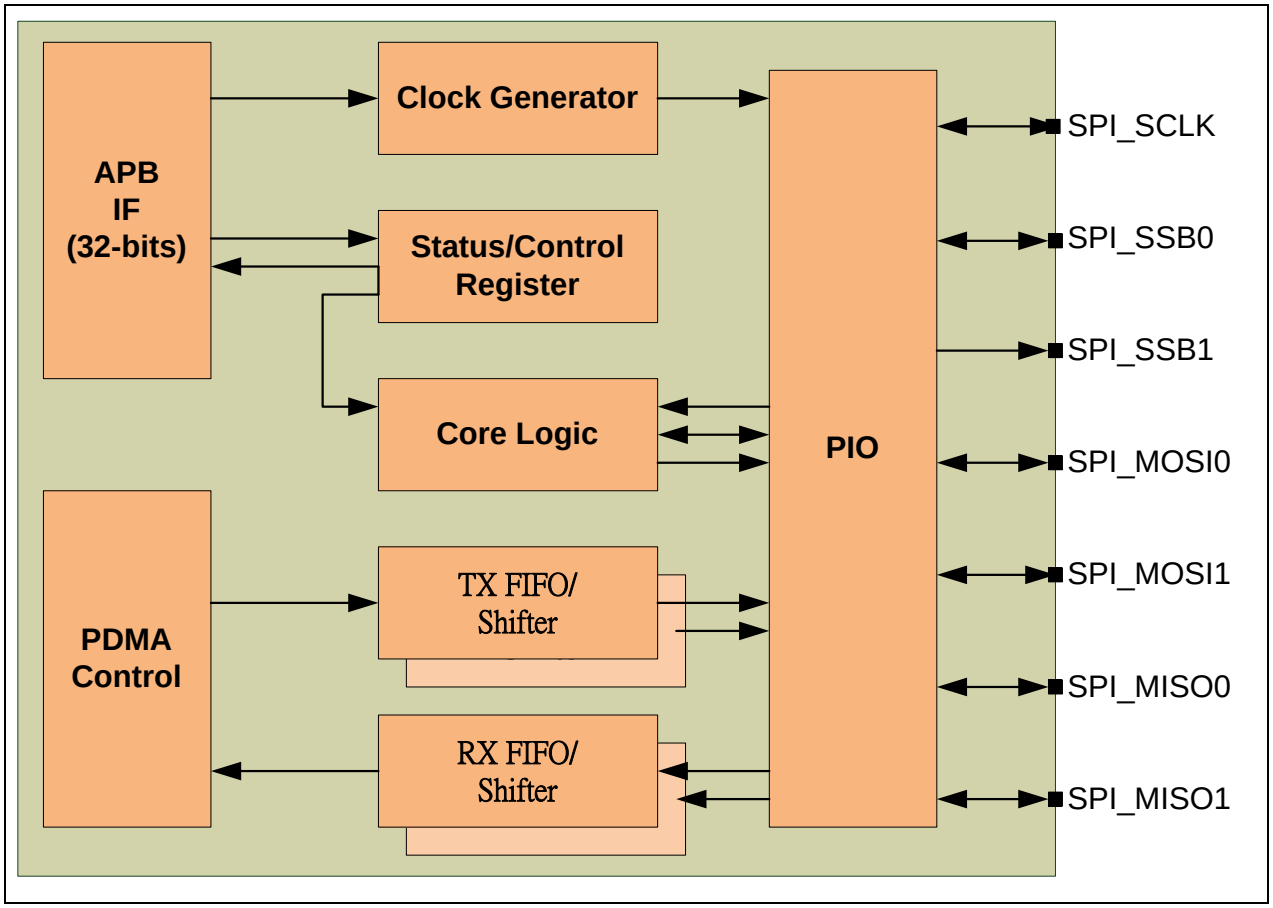


Figure 5.7-1 SPI Block Diagram

5.7.4 SPI Function Descriptions

5.7.4.1 SPI Engine Clock and SPI Serial Clock

The SPI controller derives its clock source from the system HCLK as determined by the CLK_CLKSEL1 register. The frequency of the SPI master clock is determined by the divisor ratio SPI_CLKDIV.

In Master mode, the output frequency of the SPI serial clock output pin is equal to the SPI engine clock rate. In general, the SPI serial clock is denoted as SPI clock. In Slave mode, the SPI serial clock is provided by an off-chip master device. The SPI engine clock rate of slave device must be faster than the SPI serial clock rate of the master device. The frequency of SPI engine clock cannot be faster than the APB clock rate regardless of Master or Slave mode.

5.7.4.2 Master/Slave Mode

This SPI controller can be configured as in master or slave mode by setting the SLAVE bit (SPI_CTL.SLAVE). In master mode the ISD91500 generates SCLK and SSB signals to access one or more slave devices. In slave mode the ISD91500 monitors SCLK and SSB signals to respond to data transactions from an off-chip master. The signal directions are summarized in the application block diagrams.

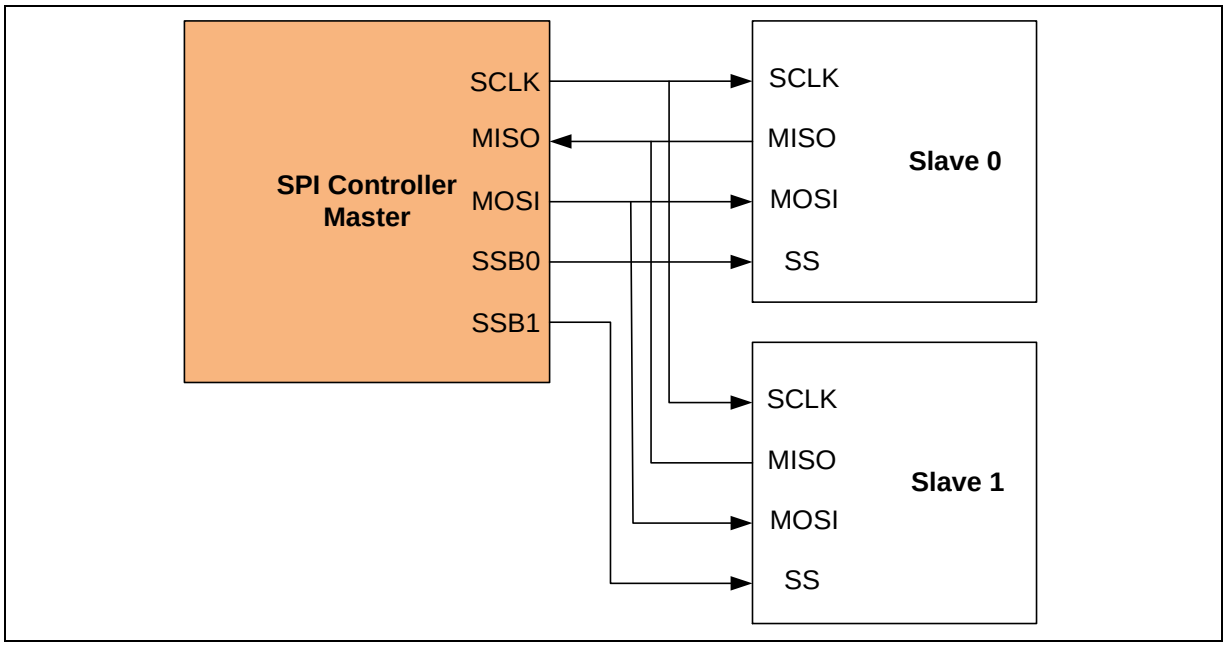


Figure 5.7-2 SPI Master Mode Application Block Diagram

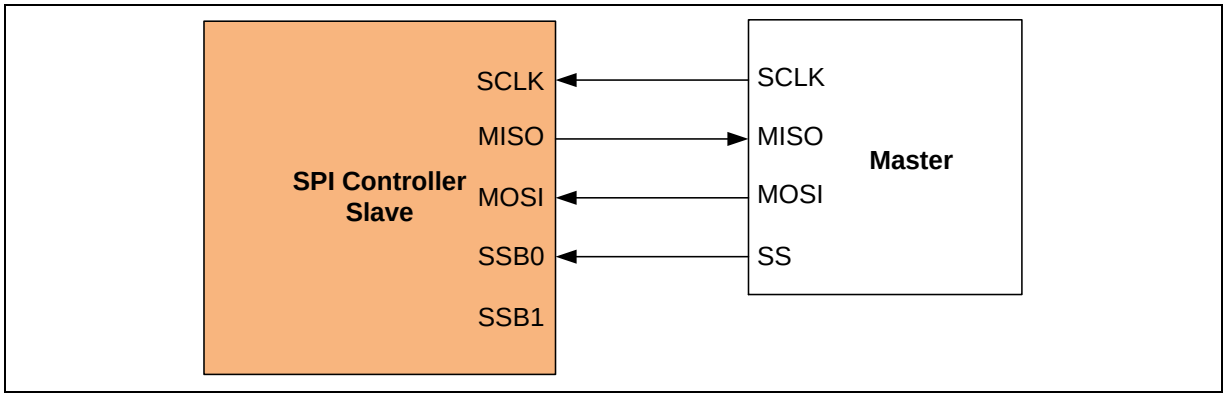


Figure 5.7-3 SPI Slave Mode Application Block Diagram

5.7.4.3 Slave Select

In master mode, the SPI controller can address up to two off-chip slave devices through the slave select output pins SPI_SSB0 and SPI_SSB1. Only one slave can be addressed at any one time. If more slave address lines are required, GPIO pins can be manually configured to provide additional SSB lines. In slave mode, the off-chip master device drives the slave select signal SPI_SSB0 to address the SPI controller. The slave select signal can be programmed to be active low or active high via the SPI_SSCTL.SSLVL bit. In addition the SPI_SSCTL.SSLVL bit defines whether the slave select signals are level triggered or edge triggered. The selection of trigger condition depends on what type of peripheral slave/master device is connected.

5.7.4.4 Automatic Slave Select

In master mode, if the bit SPI_SSCTL.AUTOSS is set, the slave select signals will be generated automatically and output to SPI_SSB0 and SPI_SSB1 pins according to registers SPI_SSCTL.SS[0] and SPI_SSCTL.SS[1]. In this mode, SPI controller will assert SSB when transaction is triggered and de-assert when data transfer is finished. If the SPI_SSCTL.AUTOSS bit is cleared, the slave select output signals are asserted and de-asserted by manual setting and clearing the related bits in the SPI_SSCTL.SS[1:0] register. The active level of the slave

select output signals is specified by the SPI_SSCTL.SSLVL bit.

In Master mode, if the value of SUSPITV[3:0] is less than 3 and AUTOSS is enabled, the slave select signal will be kept in active state between two successive transactions.

In Slave mode, to recognize the inactive state of the slave select signal, the inactive period of the slave select signal must be larger than or equal to 3 engine clock periods between two successive transactions.

5.7.4.5 Serial Clock

In master mode, writing a divisor into the SPI_CLKDIV.DIVIDER register will program the output frequency of serial clock to the SPI_SCLK output port. In slave mode, the off-chip master device drives the serial clock through the SPI_SCLK.

5.7.4.6 Clock Polarity

The SPI_CTL.CLKP bit defines the serial clock idle state in master mode. If CLKP = 1, the output SPI_SCLK is high in idle state. If CLKP=0,it is low in idle state.

5.7.4.7 Transmit/Receive Bit Length

The bit length of a transfer word is defined in SPI_CTL.DWIDTH bit field. It is set to define the length of a transfer word and can be up to 32 bits in length. DWIDTH=0x0 enables 32bit word length.

5.7.4.8 LSB First

The SPI_CTL.LSB bit defines the **bit** order of data transmission. If LSB=0 then MSB of transfer word is sent first in time. If LSB=1 then LSB of transfer word is sent first in time. If REORDER is active, then the LSB=1 causes the bit order of each byte to be reversed, not the bit order of the short or word transmission.

For transmission, if DWIDTH is a byte multiple and LSB=1 , bytes are always reordered.

LSB is not valid and must be set to 0 for DUAL or QUAD SPI transactions.

5.7.4.9 Transmit Edge

The SPI_CTL.TXNEG bit determines whether transmit data is changed on the positive or negative edge of the SPI_SCLK serial clock. If TXNEG=0 then transmitted data will change state on the rising edge of SPI_SCLK. If TXNEG=1 then transmitted data will change state on the falling edge of SPI_SCLK.

5.7.4.10 Receive Edge

The SPI_CTL.RXNEG bit determines whether data is received at either the negative edge or positive edge of serial clock SPI_SCLK. If RXNEG=1 then data is clocked in on the falling edge of SPI_SCLK. If RXNEG=0 data is clocked in on the rising edge of SPI_SCLK. Note that RXNEG should be the inverse of TXNEG for standard SPI operation.

5.7.4.11 Word Suspend

The four bit field SUSPITV (SPI_CTL[7:4]) provides a configurable suspend interval of 0.5 ~ 15.5 SPI clock periods, between two successive transaction words in Master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SUSPITV is 0x3 (3.5 SPI clock cycles).

.

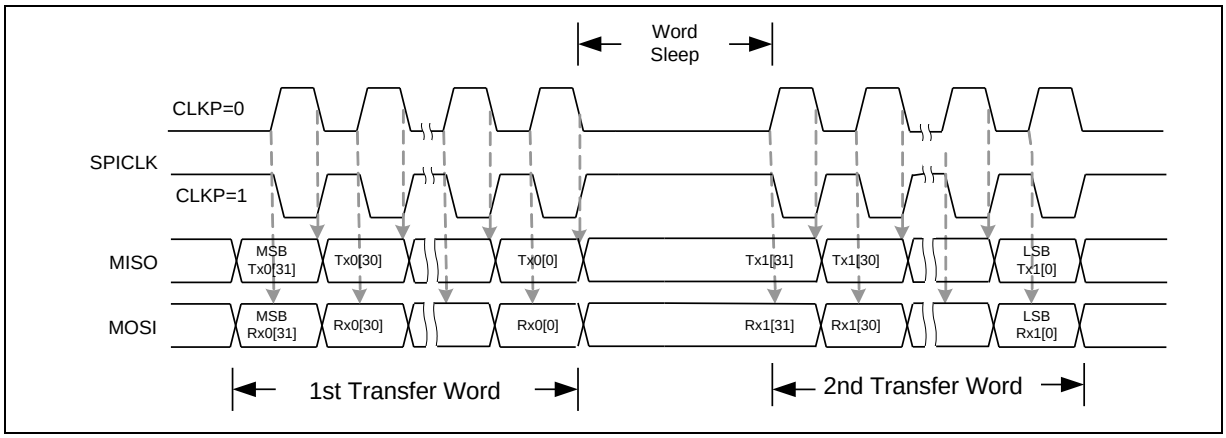


Figure 5.7-4 Word Sleep Suspend Mode

5.7.4.12 **Byte Reorder**

APB access to the SPI controller is via the 32bit wide TX and RX registers. When the transfer is set as MSB first (SPI_CTL.LSB = 0) and the SPI_CTL.REORDER bit is set, the data stored in the TX buffer and RX buffer will be rearranged such that the least significant physical byte is processed first. For DWIDTH =0 (32 bits transfer), the sequence of transmitted/received data will be BYTE0, BYTE1, BYTE2, and then BYTE3. If DWIDTH is set to 24-bits, the sequence will be BYTE0, BYTE1, and BYTE2. For Quad and Dual SPI transactions, REORDER is only valid for receive operation. For transmit in Dual/Quad modes, REORDER must be set to 0.

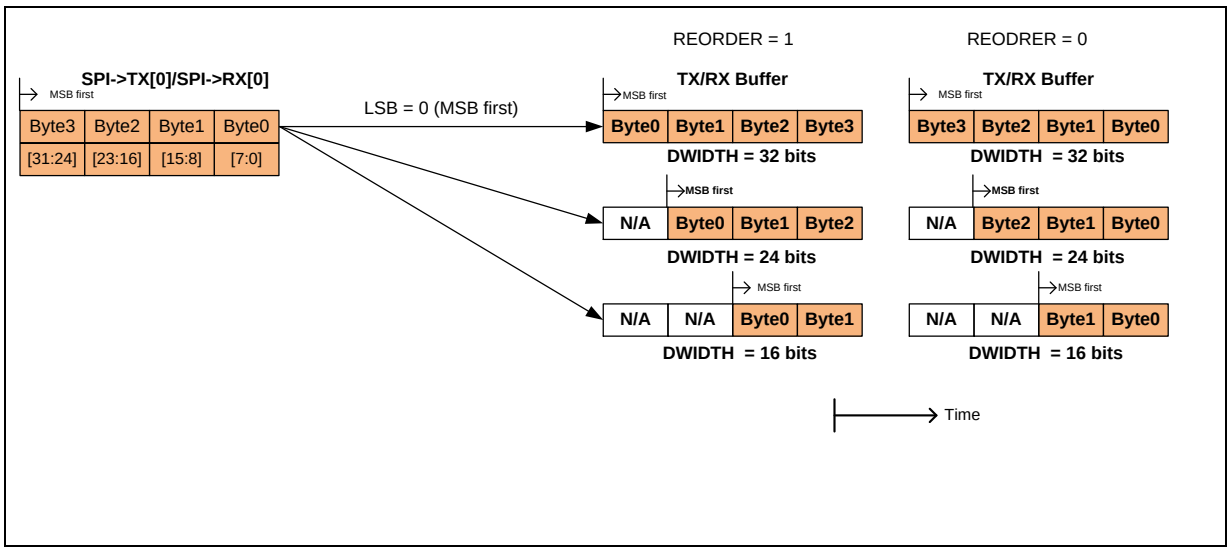


Figure 5.7-5 Byte Re-Ordering Transfer

Byte ordering can be a confusing issue when converting from arrays of data processed by the CPU for transmission out the SPI port. The CortexM0 stores data in a little endian format; that is the LSB of a multi-byte word or half-word are stored first in memory. Consider how the CortexM0 stores the following arrays in memory:

1. unsigned char ucSPI_DATA[]={0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08};
2. unsigned int uiSPI_DATA[]={0x01020304, 0x05060708};

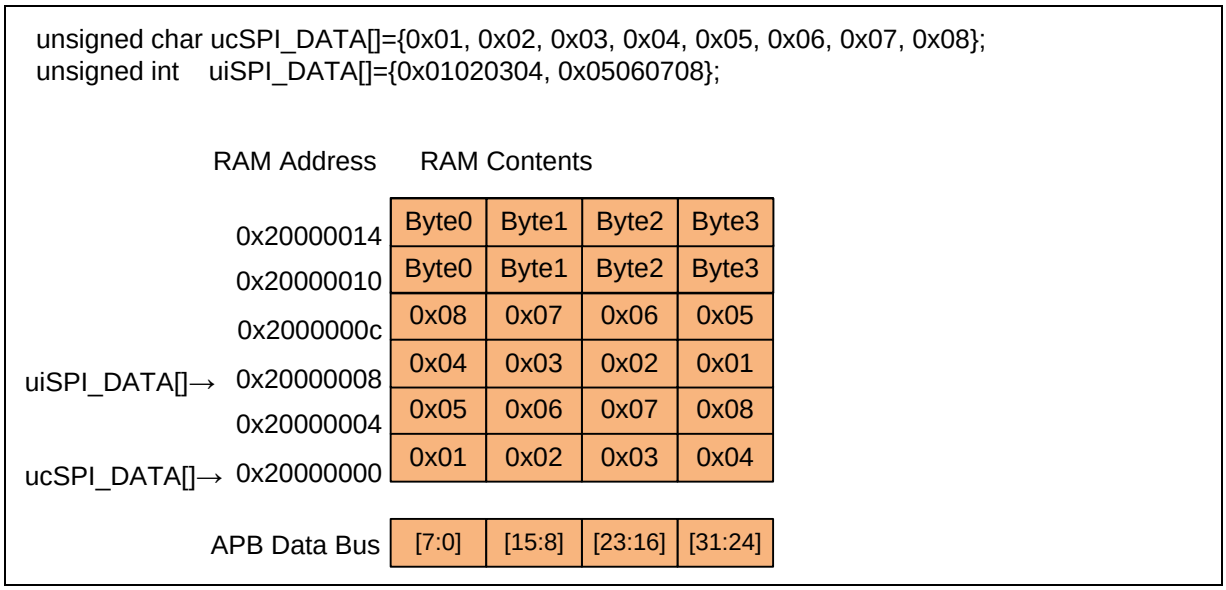


Figure 5.7-6 Byte Order in Memory

It can be seen from that byte order for an array of bytes is different than that of an array of words. Now consider if this data were to be sent to the SPI port; the user could:

1. Set DWIDTH=8 and send data byte-by-byte SPI_TX = ucSPI_DATA[i++]
2. Set DWIDTH=32 and send word-by-word SPI_TX = uiSPI_DATA[i++]

Both of these would result in the byte stream {0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08} being sent.

It would be common that a byte array of data is constructed but user, for efficiency, wishes to transfer data to SPI via word transfers. Consider the situation of where a int pointer points to the byte data array.

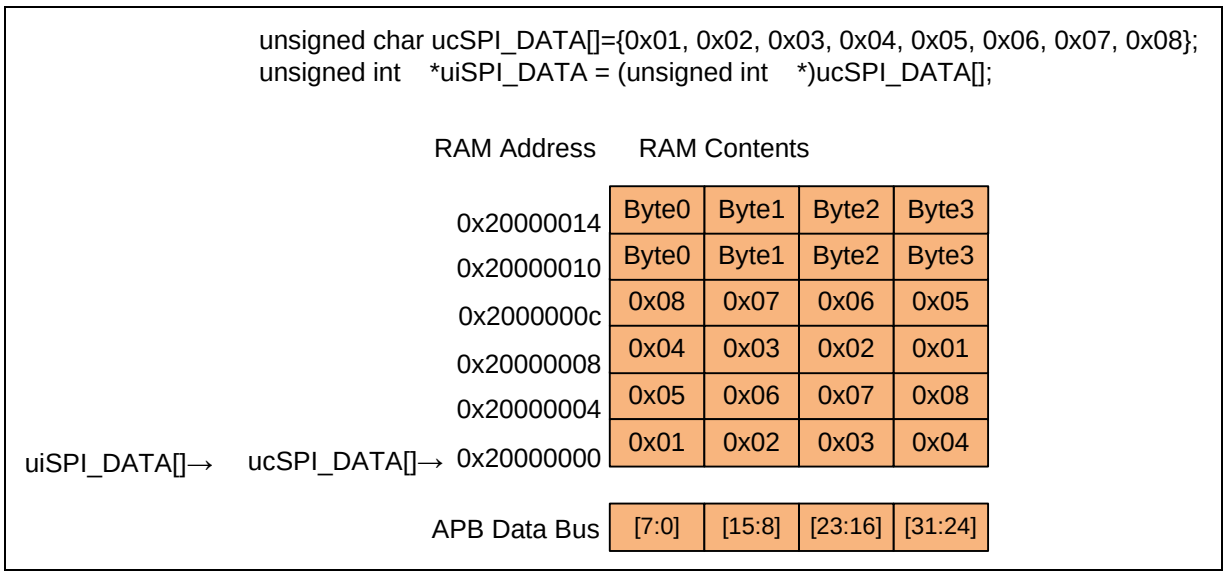


Figure 5.7-7 Byte Order in Memory

Now if we set DWIDTH=32 and sent word-by-word SPI_TX[0] = uiSPI_DATA[i++], the order transmitted would be {0x04, 0x03, 0x02, 0x01, 0x08, 0x07, 0x06, 0x05}. However if we set REORDER=1, we would reverse this order to the desired stream: {0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08}.

5.7.4.13 **Interrupt**

- SPI unit transfer interrupt

As the SPI controller finishes a unit transfer, the unit transfer interrupt flag IF (SPI_STATUS[1]) will be set to 1. The unit transfer interrupt event will generate an interrupt to CPU if the unit transfer interrupt enable bit IE (SPI_CTL[17]) is set. The unit transfer interrupt flag is cleared by writing 1 to it.

- SPI slave select interrupt

In slave mode, there are slave select active and in-active interrupt flag, SSACTIF and SSINAIF, will be set to 1 when the SPIEN and SLAVE bits were set to 1 and slave senses the slave select signal active or inactive. The SPI controller will issue an interrupt if the SSINAIEN or SSACTIEN, SPI_SSCTL[13:12], are set to 1.

- Slave Time-out interrupt

In Slave mode, there is slave time-out function for user to know that there is serial clock input but one transaction doesn't finish over the period of SLVTOCNT basing on engine clock.

When the Slave select is active and the value of SLVTOCNT is not 0, the Slave time-out counter in the SPI controller logic will start after the serial clock input. This counter will be clear after one transaction done or the SLVTOCNT is set to 0. If the value of the time-out counter greater or equal than the value of SLVTOCNT before one transaction done, the slave time-out event occurs and the SLVTOIF, SPI_STATUS[5], will be set to 1. The SPI controller will issue an interrupt if the SLVTOIEN, SPI_SSCTL[5], is set to 1.

- Slave Error 0 interrupt

In Slave mode, if the transmit/ receive bit count mismatch with the DWIDTH when the slave select line goes to inactive state, the Slave mode error 0, SLVBEIF, SPI_STATUS[6], will be set to 1. The SPI controller will issue an interrupt if the SLVBCEIEN, SPI_SSCTL[8], is set to 1.

Note: 1. In Slave transmit mode, if there is bit length transmit error (bit count mismatch), the user shall set the TXRST bit and write the transmit datum again to restart the next transaction.

2. If the slave select active but there is no any serial clock input, the SLVBEIF also active when the slave select goes to inactive state.

- Slave Under-run and Slave Error 1 interrupts

In Slave mode, if there is no any data is written to the SPI_TX register, the under-run event, TXUFIF (SPI_STATUS[19]) will active when the slave select active and the serial clock input this controller. The SPI controller will issue an interrupt if the SLVUDRIEN is set to 1.

Under the previous condition, the Slave mode error 1, SLVURIF, SPI_STATUS[7], will be set to 1 when SS goes to inactive state and transmit under-run occurs. The SPI controller will issue an interrupt if the SLVUDRIEN, SPI_SSCTL[9], is set to 1.

Note: In SLV3WIRE mode, the slave select bus active all the time so that the user shall polling the TXUFIF bit to know if there is transmit under-run event or not.

- Receive Over-run interrupt

In Slave mode, if the receive FIFO buffer contains 4 unread data, the RXFULL flag will be set to 1 and the RXOVIF will be set 1 if there is more serial data is received from SPIMOSI and the RXOVIF will be set to 1 and follow-up data will be dropped. The SPI controller will issue an interrupt if the RXOVIEN, SPI_FIFOCTL[5], is set to 1.

- Receive FIFO time-out interrupt

In FIFO mode, there is a time-out function to inform user. If there is a received data in the FIFO and it is not read by software over 64 SPI engine clock periods in Master mode or over 576 SPI engine clock periods in Slave mode, it will send a time-out interrupt to the system if the time-out interrupt enable bit, RXTOIEN, SPI_FIFOCTL[4], is set to 1.

- Transmit FIFO interrupt

In FIFO mode, if the valid data count of the transmit FIFO buffer is less than or equal to the setting value of TXTH, the transmit FIFO interrupt flag will be set to 1. The SPI controller will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, SPI_FIFOCTL[3], is set to 1.

- Receive FIFO interrupt

In FIFO mode, if the valid data count of the receive FIFO buffer is larger than the setting value of RXTH, the receive FIFO interrupt flag will be set to 1. The SPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, SPI_FIFOCTL[2], is set to 1.

5.7.4.14 **3-Wire Mode**

When the SLV3WIRE bit is set by software to enable the Slave 3-wire mode, the SPI controller can work with no slave select signal in Slave mode. The SLV3WIRE bit only takes effect in Slave mode. Only three pins, SPICLK, SPI_MISO0, and SPI_MOSI0, are required to communicate with a SPI master. The SPI_SS pin can be configured as a GPIO. When the SLV3WIRE bit is set to 1, the SPI slave will be ready to transmit/receive data after the SPIEN bit is set to 1.

5.7.4.15 **Dual/Quad I/O Mode**

The SPI controller supports dual and quad I/O transfer when setting the DUALIOEN bit or the QUADIOEN bit (SPI_CTL[21], SPI_CTL[22]) to 1. Many SPI Serial Flash devices support Dual/ Quad I/O transfer. The QDIODIR bit (SPI_CTL[20]) is used to define the direction of the transfer data. When the QDIODIR bit is set to 1, the controller will send the data to external device. When the QDIODIR bit is set to 0, the controller will read the data from the external device. This function supports transfers of 8, 16, 24, and 32-bits.

The Dual/Quad I/O mode is not supported in the Slave 3-wire mode. The byte REORDER function is only available in receive mode for Dual/Quad transactions.

For Dual I/O mode, if both the DUALIOEN and QDIODIR bits are set as 1, the SPI_MOSI0 is the even bit data output and the SPI_MISO0 will be set as the odd bit data output. If the DUALIOEN is set as 1 and QDIODIR is set as 0, both the SPI_MISO0 and SPI_MOSI0 will be set as data input ports.

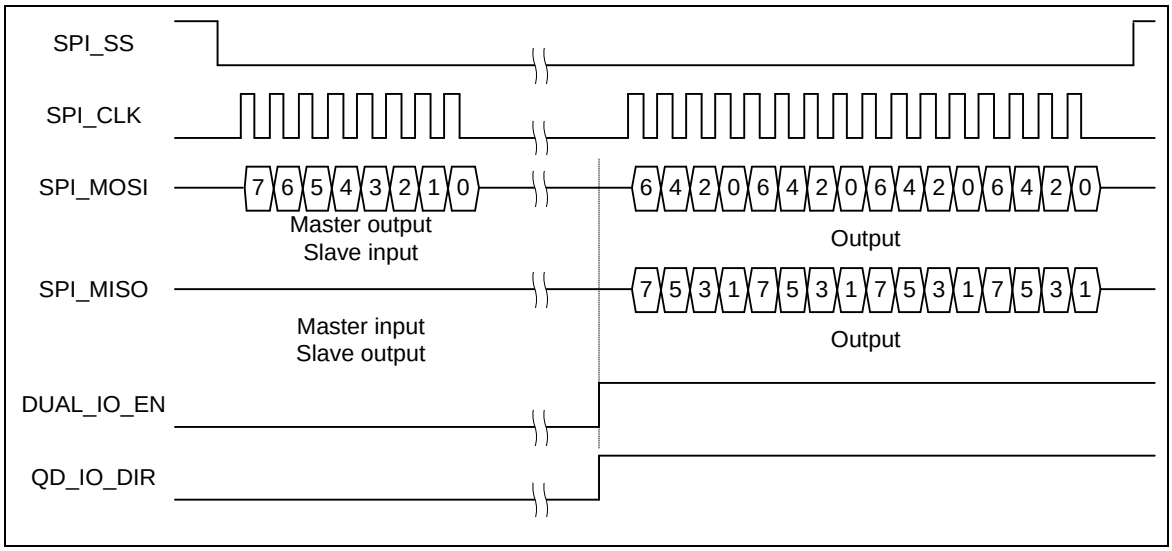


Figure 5.7-8 Bit Sequence of Dual Output Mode

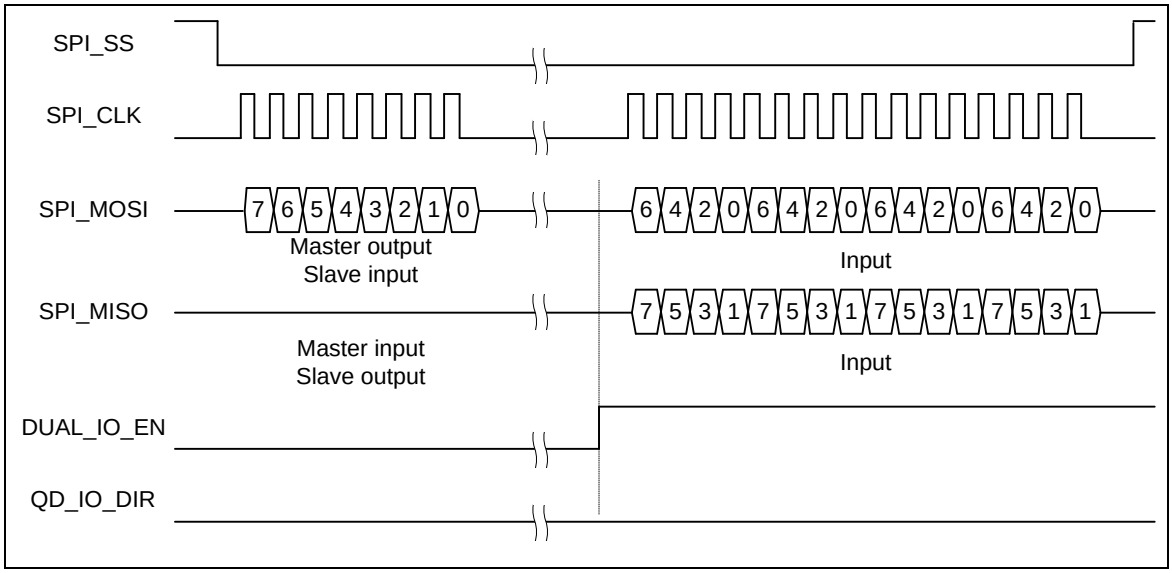


Figure 5.7-9 Bit Sequence of Dual Input Mode

For Quad I/O mode, if both the QUADIOEN and QDIODIR bits are set as 1, the SPI_MOSI0 and SPI_MOSI1 are the even bit data output and the SPI_MISO0 and SPI_MISO1 will be set as the odd bit data output. If the QUADIOEN is set as 1 and QDIODIR is set as 0, both the SPI_MISO0, SPI_MISO1, SPI_MOSI0 and SPI_MOSI1 will be set as data input ports.

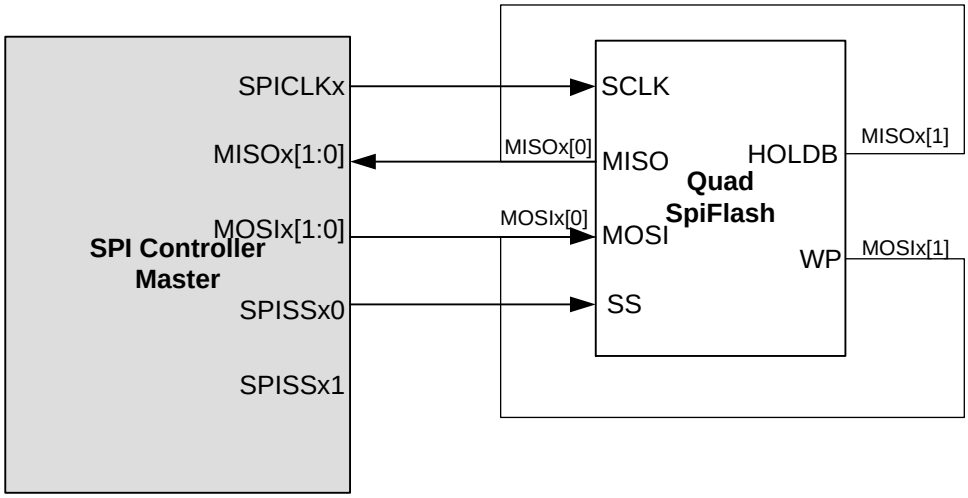


Figure 5.7-10 Quad Mode System Architecture

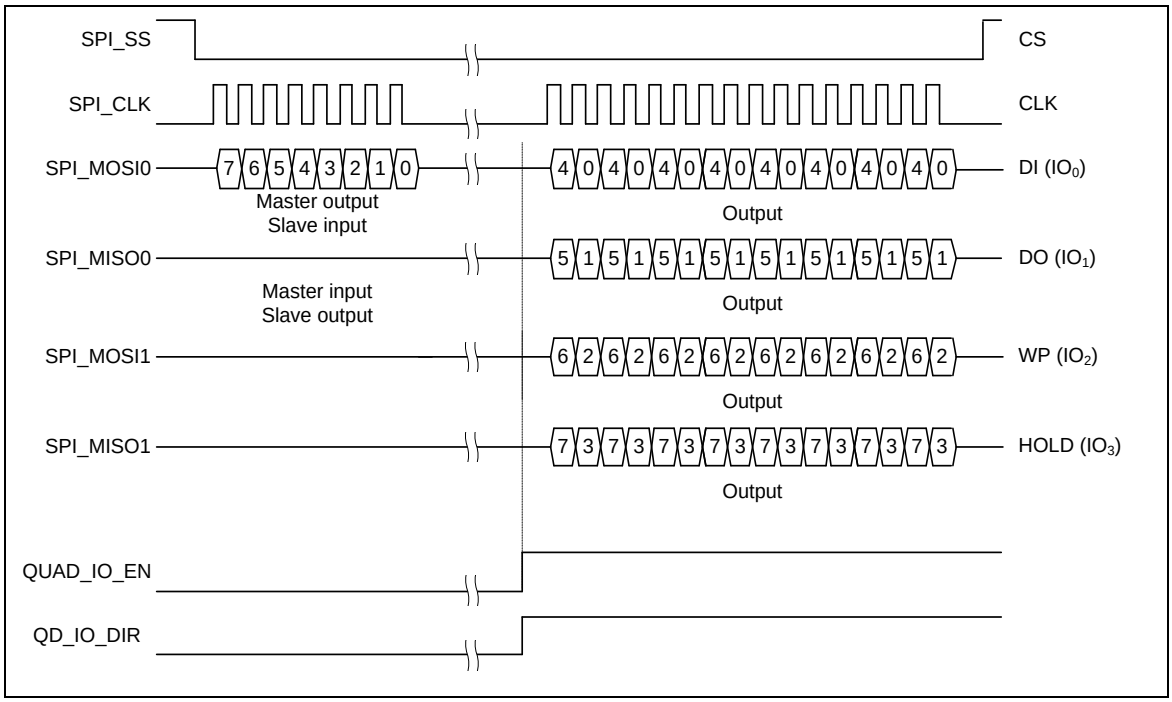


Figure 5.7-11 Bit Sequence of Quad Output Mode

5.7.4.16 **8-Level FIFO Buffer**

The SPI controller is equipped with eight 32-bit wide transmit and receive FIFO buffers.

The transmit FIFO buffer is an 8-level depth, 32-bit wide, first-in, first-out register buffer. 8 words of data can be written to the transmit FIFO buffer in advance through software by writing the SPI_TX register. The data stored in the transmit FIFO buffer will be read and sent out by the transmission control logic. If the 8-level transmit FIFO buffer is full, the TXFULL bit will be set to 1. When the SPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the 8-level transmit FIFO buffer is empty, the TXEMPTY bit will be set to 1. Notice that the TXEMPTY flag is set to 1 while the last transaction is still in progress. In Master mode, both the BUSY bit (SPI_STATUS[0]) and TXEMPTY bit should be checked by software to make sure whether the SPI is idle or not.

The received FIFO buffer is also an 8-level depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the received data to this buffer. The FIFO buffer data can be read from SPI_RX register by software. There are FIFO related status bits, like RXEMPTY and RXFULL, to indicate the current status of FIFO buffer.

The transmitting and receiving threshold can be set through software by setting the TXTH, SPI_FIFOCTL[29:28], and RXTH, SPI_FIFOCTL[25:24], settings. When the count of valid data stored in transmit FIFO buffer is less than or equal to TXTH setting, the TXTHIF, SPI_STATUS[18], bit will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RXTH setting, the RXTHIF, SPI_STATUS[10], bit will be set to 1.

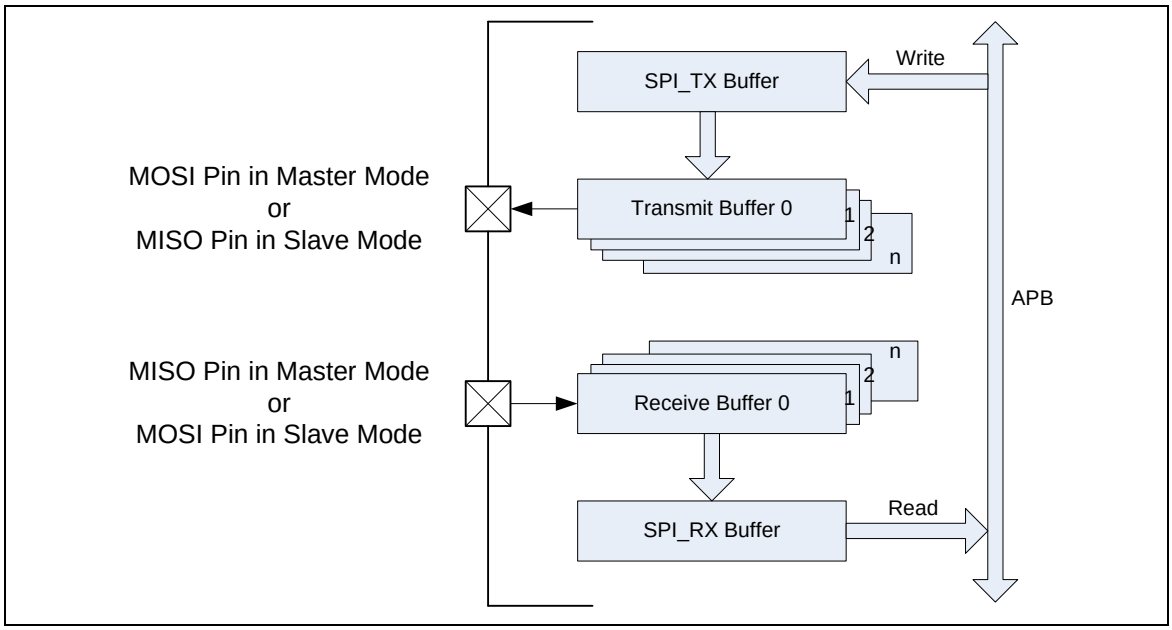


Figure 5.7-12 FIFO Mode Block Diagram

In Master mode, the first datum is written to the SPI_TX register, the TXEMPTY flag will be cleared to 0. The transmission immediately starts as long as the transmit FIFO buffer is not empty. User can write the next data into SPI_TX register immediately. The SPI controller will insert a suspend interval between two successive transactions and the period of suspend interval is decided by the setting of SUSPITV (SPI_CTL [7:4]). User can write data into SPI_TX register as long as the TXFULL flag is 0.

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPI_TX register is not updated after data transfer is done, the transfer will stop.

In Master mode, during receive operation, the serial data is received from SPI_MISO0/1 pin and stored to receive FIFO buffer. The RXEMPTY flag will be cleared to 0 while the receive FIFO buffer contains unread data. The received data can be read by software from SPI_RX register as long as the RXEMPTY flag is 0. If the receive FIFO buffer contains 8 unread data, the RXFULL flag will be set to 1. The SPI controller will stop receiving data until the SPI_RX register is read by software.

In Slave mode, during transmission operation, when data is written to the SPI_TX register by software, the data will be loaded into transmit FIFO buffer and the TXEMPTY flag will be set to 0. The transmission will start when the slave device receives clock signal from master. Data can be written to SPI_TX register as long as the TXFULL flag is 0. After all data have been drawn out by the SPI transmission logic unit and the SPI_TX register is not updated by software, the TXEMPTY flag will be set to 1.

If there is no any data is written to the SPI_TX register, the under-run event, TXUFIF (SPI_STATUS[19]) will active when the slave select active and the serial clock input this controller. Under the previous condition, the Slave mode error 1, SLVURIF, SPI_STATUS[7], will be set to 1 when SS goes to inactive state and transmit under-run occurs.

In Slave mode, during receiving operation, the serial data is received from SPI_MOSI0/1 pin and stored to SPI_RX register. The reception mechanism is similar to Master mode reception operation. If the receive FIFO buffer contains 8 unread data, the RXFULL flag will be set to 1 and the RXOVIF will be set 1 if there is more serial data is received from SPIMOSI and follow-up data will be dropped. If the receive bit counter mismatch with the DWIDTH when the slave select line goes to inactive state, the Slave mode error 0, SLVBEIF, SPI_STATUS[6], will be set to 1.

When the Slave select is active and the value of SLVTOCNT is not 0, the Slave time-out counter in the SPI controller logic will start after the serial clock input. This counter will be clear after one transaction done or the SLVTOCNT is set to 0. If the value of the time-out counter greater or equal than the value of SLVTOCNT before one transaction done, the slave time-out event occurs abd the SLVTOIF, SPI_STATUS[5], will be set to 1.

A receive time-out function is built-in in this controller. When the receive FIFO is not empty and no read operation in receive FIFO over 64 SPI clock period in Master mode or over 576 SPI engine clock period in Slave mode, the receive time-out occurs and the RXTOIF be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

5.7.4.17 DMA Receive Mode

The SPI controller supports DMA access to the transmit and receive FIFOs. When the DMA transmit interface is active, DMA sub-system fills the TX FIFO to trigger SPI interface. When only DMA receive function is required, an additional mode is provided to inform the SPI system of the number of transfers desired so that SPI system can read ahead of DMA requests. When SPI0_CTL.RXTCNTEN is set, the register SPI_RXTSNCNT holds the number of SPI transactions (total number of bytes is determined by SPI_CTL.DWIDTH value).

5.7.5 SPI Timing Diagram

In master/slave mode, the device address/slave select (SPI_SSB0/1) signal can be configured as active low or active high by the SPI_SSCTL.SSLVL bit.

The serial clock phase and polarity is controlled by CLKP, RXNEG and TXNEG bits. The bit length of a transfer word is configured by the DWIDTH parameter. Whether data transmission is MSB first or LSB first is controlled by the SPI_CTL.LSB bit. Four examples of SPI timing diagrams for master/slave operations and the related settings are shown as below.

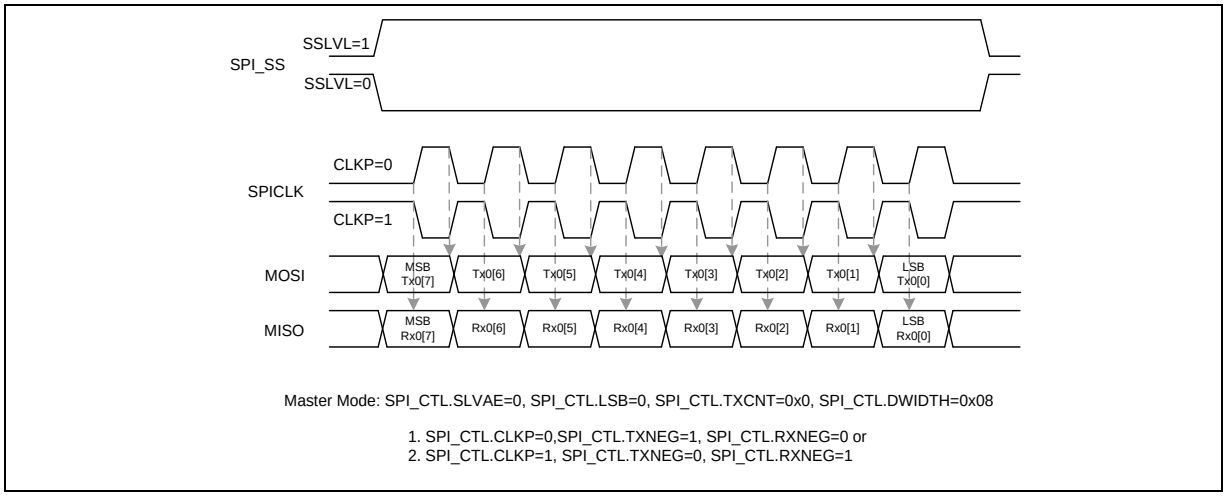
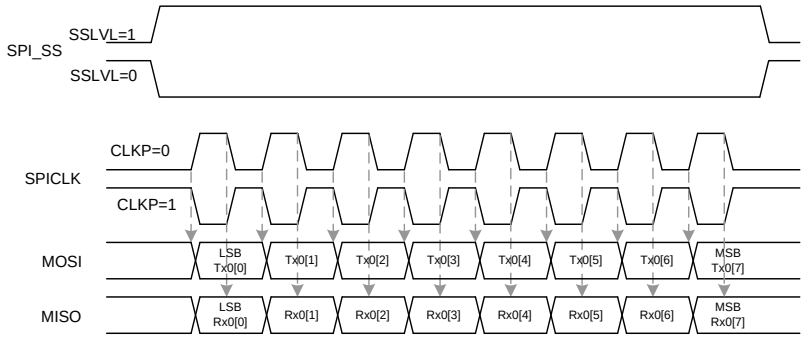


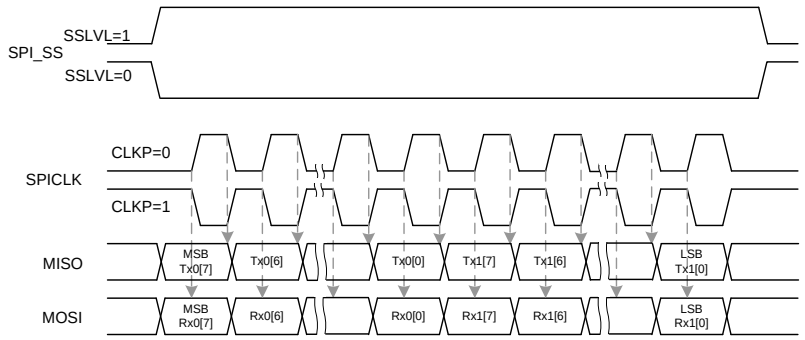
Figure 5.7-13 SPI Timing in Master Mode



Master Mode: SPI_CTL.SLVAE=0, SPI_CTL.LSB=1, SPI_CTL.TXCNT=0x0, SPI_CTL.DWIDTH=0x08

1. SPI_CTL.CLKP=0,SPI_CTL.TXNEG=0, SPI_CTL.RXNEG=1 or
2. SPI_CTL.CLKP=1, SPI_CTL.TXNEG=1, SPI_CTL.RXNEG=0

Figure 5.7-14 SPI Timing in Master Mode (Alternate Phase of SPICLK)



Master Mode: SPI_CTL.SLVAE=1, SPI_CTL.LSB=0, SPI_CTL.TXCNT=0x1, SPI_CTL.DWIDTH=0x08

1. SPI_CTL.CLKP=0,SPI_CTL.TXNEG=1, SPI_CTL.RXNEG=0 or
2. SPI_CTL.CLKP=1, SPI_CTL.TXNEG=0, SPI_CTL.RXNEG=1

Figure 5.7-15 SPI Timing in Slave Mode

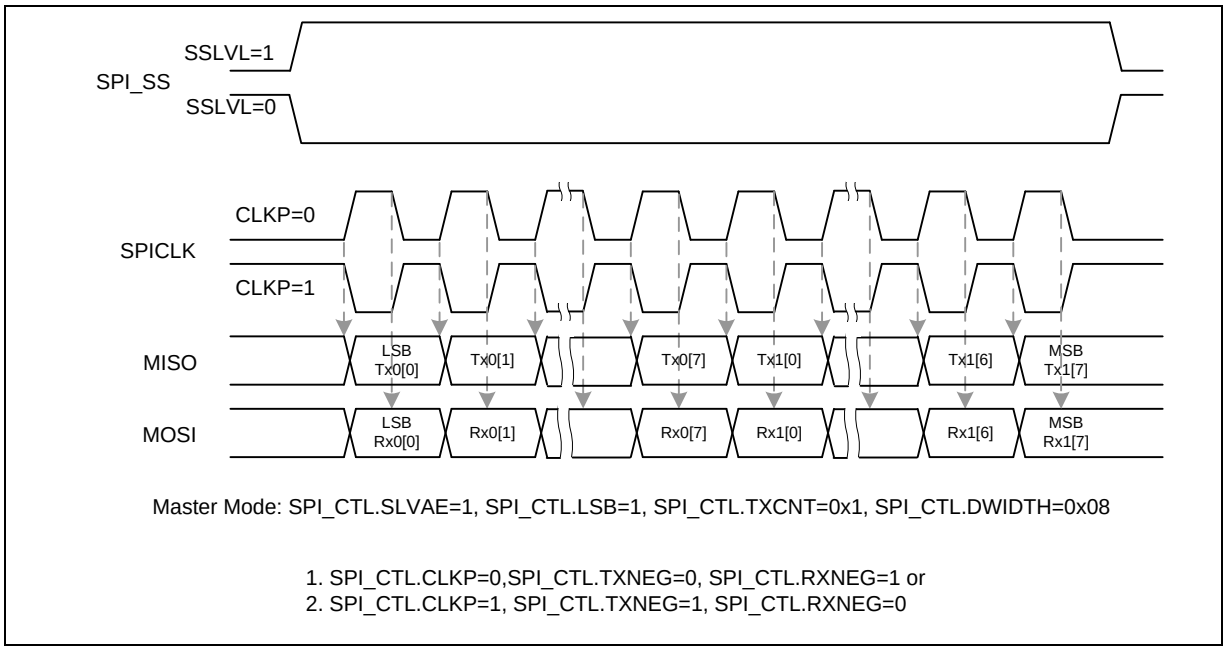


Figure 5.7-16 SPI Timing in Slave Mode (Alternate Phase of SPICLK)

5.7.6 SPI Configuration Examples

- Example 1, SPI controller is set as a master to access an off-chip slave device with following specifications:

- Data bit latched on positive edge of serial clock
- Data bit driven on negative edge of serial clock
- Data be transferred from MSB first
- SCLK low in idle state
- Only one byte data be transmitted/received in a transfer
- Slave select signal is active low
- SCLK frequency is 10MHz

To configure the SPI interface to the above specifications perform the following steps:

1) Write a divisor into the SPI_CLKDIV register to determine the output frequency of serial clock. Driver function DrvSPI_SetClock(0,10000000,0) can be used to achieve this.
2) Configure the SPI_SSCTL register to address device. For example to manually address, set SPI_SSCTL.AUTOSS=0, SPI_SSCTL.SSLVL =0 for active low SS. When software wishes to address device it will set SPI_SSCTL.SS=1 to output an active SS on SPI_SSB0 pin.
3) Configure the SPI_CTL register. Set SPI_CTL.SLAVE=0 for master mode, set SPI_CTL.CLKP=0 for SCLK polarity normally low, set SPI_CTL.TXNEG=1 so that data changes on falling edge of SCLK, set SPI_CTL.RXNEG=0 so that data is latched into device on positive edge of SCLK, set SPI_CTL.DWIDTH=8 for a single byte transfer and finally set SPI_CTL.LSB=0 for MSB first transfer.
4) If manually selecting slave device set SPI_SSCTL.SS=1.
5) To transmit one byte of data, write data to SPI_TX register. If only doing a receive, write a dummy byte to SPI_TX register.
6) Enable the SPI_CTL.SPIEN bit to start the data transfer over the SPI interface.
7) Wait for SPI transfer to finish. Can be interrupt driven (if the interrupt enable SPI_CTL.IE bit is set) or by polling the GO_BUSY bit which will be cleared to 0 by hardware automatically at end of transmission. --
8) Read out the received one byte data from SPI_RX
9) Go to 5) to continue another data transfer or set SPI_SSCTL.SS=0 to deactivate the off-chip slave devices.

- Example 2, SPI controller is set as a slave device that controlled by an off-chip master device with the following characteristics:

- Data bit latched on positive edge of serial clock
- Data bit driven on negative edge of serial clock
- Data be transferred from LSB first
- SCLK high in idle state
- Only one byte data be transmitted/received in a transfer
- Slave select signal is active high level trigger

To configure the SPI interface to the above specifications perform the following steps:

1) Configure the SPI_SSCTL register. SPI_SSCTL.SSACTPOL=1 for active high slave select, SPI_SSCTL.SS_LTRIG=1 for level sensitive trigger.
2) Configure the SPI_CTL register. Set SPI_CTL.SLAVE=1 for slave mode, set SPI_CTL.CLKPOL=1 for SCLK polarity idle high, set SPI_CTL.TXNEG=1 so that data changes on falling edge of SCLK, set SPI_CTL.RXNEG=0 so that data is latched into device on positive edge of SCLK, set SPI_CTL.DWIDTH=8 for a single byte transfer and finally set SPI_CTL.LSB=1 for LSB first transfer.
3) If SPI slave is to transmit one byte of data to the off-chip master device, write first byte to TX register. If no data to be transmitted write a dummy byte.
4) Enable the SPIEN bit to wait for the slave select trigger input and serial clock input from the off-chip master device to start the data transfer at the SPI interface.
5) Wait for SPI transfer to finish. Can be interrupt driven (if the interrupt enable SPI_CTL.IE bit is set) or by polling the GO_BUSY bit which will be cleared to 0 by hardware automatically at end of transmission. --
6) Read out the received data from RX register.
7) Go to 3) to continue another data transfer or disable the SPIEN bit to stop data transfer.

5.7.7 SPI0/1 Control Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPIx_BA = 0x4003_0000 + (0x1000 * x)** **x=0,1**				
SPI_CTL	SPIx_BA + 0x00	R/W	Control and Status Register	0x0000_0034
SPI_CLKDIV	SPIx_BA + 0x04	R/W	Clock Divider Register (Master Only)	0x0000_0000
SPI_SSCTL	SPIx_BA + 0x08	R/W	Slave Select Register	0x0000_0000
SPI_PDMACTL	SPIx_BA + 0x0C	R/W	SPI PDMA Control Register	0x0000_0000
SPI_FIFOCTL	SPIx_BA + 0x10	R/W	FIFO Control/Status Register	0x4400_0000
SPI_STATUS	SPIx_BA + 0x14	R/W	Status Register	0x0005_0110
SPI_RXTSNCNT	SPIx_BA + 0x18	R/W	Receive Transaction Count Register	0x0000_0000
SPI_TX	SPIx_BA + 0x20	W	FIFO Data Transmit Register	0x0000_0000
SPI_RX	SPIx_BA + 0x30	R	FIFO Data Receive Register	0x0000_0000

5.7.8 SPI0/1 Control Register Description

SPI Control and Status Register (SPI_CTL)

Register	Offset	R/W	Description	Reset Value
SPI_CTL	SPIx_BA + 0x00	R/W	Control and Status Register	0x0000_0034

31	30	29	28	27	26	25	24
Reserved							**RXMODEEN**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXTCNTEN	**QUADIOEN**	**DUALIOEN**	**QDIODIR**	**REORDER**	**SLAVE**	**IE**	**Reserved**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**LSB**	**DWIDTH**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SUSPITV				**CLKP**	**TXNEG**	**RXNEG**	**SPIEN**

Bits	Description	
[31:25]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**RXMODEEN**	**FIFO Receive Mode Enable** 0 = Disable function. 1 = Enable FIFO receive mode. In this mode SPI transactions will be continuously performed while RXFULL is not active. To stop transactions, set RXMODEEN to 0. **Note:** When RXMODEEN is 1'b, the value of SUSPITV must >=1 .
[23]	**RXTCNTEN**	**DMA Receive Transaction Count Enable** 0 = Disable function. 1 = Enable transaction counter for DMA receive only mode. SPI will perform the number of transfers specified in the SPI_RXTSNCNT register, allowing the SPI interface to read ahead of DMA controller.
[22]	**QUADIOEN**	**Quad I/O Mode Enable** 0 = Quad I/O mode Disabled. 1 = Quad I/O mode Enabled.
[21]	**DUALIOEN**	**Dual I/O Mode Enable** 0 = Dual I/O mode Disabled. 1 = Dual I/O mode Enabled.

[20]	**QDIODIR**	**Quad or Dual I/O Mode Direction Control** 0 = Quad or Dual Input mode. 1 = Quad or Dual Output mode.
[19]	**REORDER**	**Byte Reorder Function Enable** 0 = Byte reorder function Disabled. 1 = Byte reorder function Enabled. A byte suspend interval will be inserted between each byte. The period of the byte suspend interval depends on the setting of SUSPITV. **Note:** Byte reorder function is only available if DWIDTH is defined as 16, 24, and 32 bits. REORDER is only available for Receive mode in DUAL and QUAD transactions. And SUSPITV must be set to 0.
[18]	**SLAVE**	**Master Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**IE**	**Unit Transfer Interrupt Enable** 0 = Disable SPI Unit Transfer Interrupt. 1 = Enable SPI Unit Transfer Interrupt to CPU.
[16:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**LSB**	**LSB First** 0 = The MSB is transmitted/received first (which bit in TX and RX FIFO depends on the DWIDTH field). 1 = The LSB is sent first on the line (bit 0 of TX FIFO]), and the first bit received from the line will be put in the LSB position in the SPIn_RX FIFO (bit 0 SPIn_RX). **Note:** For DUAL and QUAD transactions with LSB must be set to 0.
[12:8]	**DWIDTH**	**DWIDTH – Data Word Bit Length** This field specifies how many bits are transmitted in one transmit/receive. Up to 32 bits can be transmitted. DWIDTH = 0x01 … 1 bit. DWIDTH = 0x02 … 2 bits. …… DWIDTH = 0x1f … 31 bits. DWIDTH = 0x00 … 32 bits.

[7:4]	**SUSPITV**	**Suspend Interval (Master Only)** The four bits provide configurable suspend interval between two successive transmit/receive transactions in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. SUSPITV is available for standard SPI transactions, it must be set to 0 for DUAL and QUAD mode transactions. (SUSPITV[3:0] + 0.5) * period of SPICLK clock cycle Example: SUSPITV = 0x0 … 0.5 SPICLK clock cycle. SUSPITV = 0x1 … 1.5 SPICLK clock cycle. …… SUSPITV = 0xE … 14.5 SPICLK clock cycle. SUSPITV = 0xF … 15.5 SPICLK clock cycle. Note: For DUAL and QUAD transactions with SUSPITV must be set to 0.
[3]	**CLKP**	**Clock Polarity** 0 = SCLK idle low. 1 = SCLK idle high.
[2]	**TXNEG**	**Transmit at Negative Edge** 0 = The transmitted data output signal is changed at the rising edge of SCLK. 1 = The transmitted data output signal is changed at the falling edge of SCLK.
[1]	**RXNEG**	**Receive at Negative Edge** 0 = The received data input signal is latched at the rising edge of SCLK. 1 = The received data input signal is latched at the falling edge of SCLK.
[0]	**SPIEN**	**SPI Transfer Enable** 0 = Disable SPI Transfer. 1 = Enable SPI Transfer. In Master mode, the transfer will start when there is data in the FIFO buffer after this is set to 1. In Slave mode, the device is ready to receive data when this bit is set to 1. Note: All configuration should be set before writing 1 to this SPIEN bit. (e.g.: TXNEG, RXNEG, DWIDTH, LSB, CLKP, and so on).

SPI Divider Register (SPI_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SPI_CLKDIV	SPIx_BA + 0x04	R/W	Clock Divider Register (Master Only)	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**DIVIDER**	**Clock Divider Register** The value in this field is the frequency divider for generating the SPI engine clock,Fspi_sclk, and the SPI serial clock of SPI master. The frequency is obtained according to the following equation. Fspi_sclk = Fspi_clockSRC / (DIVIDER+1). where Fspi_clockSRC is the SPI engine clock source, which is defined in the clock control, CLKSEL1 register. **Note:** SPI engine clock must smaller than PCLK/2

SPI Slave Select Register (SPI_SSCTL)

Register	Offset	R/W	Description	Reset Value
SPI_SSCTL	SPIx_BA + 0x08	R/W	Slave Select Register	0x0000_0000

31	30	29	28	27	26	25	24
SLVTOCNT[15:8]							
23	22	21	20	19	18	17	16
SLVTOCNT[7:0]							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVUDRIEN	SLVBCEIEN
7	6	5	4	3	2	1	0
Reserved	SLVTORST	SLVTOIEN	SLV3WIRE	AUTOSS	SSLVL	SS	

Bits	Description	
[31:16]	**SLVTOCNT**	**Slave Mode Time-out Period** In Slave mode, these bits indicate the time out period when there is serial clock input during slave select active. The clock source of the time out counter is Slave engine clock. If the value is 0, it indicates the slave mode time-out function is disabled.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable** 0 = Slave select inactive interrupt Disable. 1 = Slave select inactive interrupt Enable.
[12]	**SSACTIEN**	**Slave Select Active Interrupt Enable** 0 = Slave select active interrupt Disable. 1 = Slave select active interrupt Enable.
[11:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**SLVUDRIEN**	**Slave Mode Error 1 Interrupt Enable** 0 = Slave mode error 1 interrupt Disable. 1 = Slave mode error 1 interrupt Enable.
[8]	**SLVBCEIEN**	**Slave Mode Error 0 Interrupt Enable** 0 = Slave mode error 0 interrupt Disable.

		1 = Slave mode error 0 interrupt Enable.
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**SLVTORST**	**Slave Mode Time-out FIFO Clear** 0 = Function disabled. 1 = Both the FIFO clear function, TXRST and RXRST, are activated automatically when there is a slave mode time-out event.
[5]	**SLVTOIEN**	**Slave Mode Time-out Interrupt Enable** 0 = Slave mode time-out interrupt Disabled. 1 = Slave mode time-out interrupt Enabled.
[4]	**SLV3WIRE**	**Slave 3-wire Mode Enable** This is used to ignore the slave select signal in Slave mode. The SPI controller can work with 3-wire interface consisting of SPI_CLK, SPI_MISO, and SPI_MOSI. 0 = 4-wire bi-directional interface. 1 = 3-wire bi-directional interface.
[3]	**AUTOSS**	**Automatic Slave Select Function Enable (Master Only)** 0 = If this bit is cleared, slave select signals will be asserted/de-asserted by setting/clearing the corresponding bits of SPI_SSCTL[1:0]. 1 = If this bit is set, SPI_SS0/1 signals will be generated automatically. It means that device/slave select signal, which is set in SPI_SSCTL[1:0], will be asserted by the SPI controller when transmit/receive is started, and will be de-asserted after each transmit/receive is finished.
[2]	**SSLVL**	**Slave Select Active Level** This bit defines the active status of slave select signal (SPI_SS0/1). 0 = The slave select signal SPI_SS0/1 is active on low-level/falling-edge. 1 = The slave select signal SPI_SS0/1 is active on high-level/rising-edge.
[1:0]	**SS**	**Slave Select Control Bits (Master Only)** If AUTOSS bit is cleared, writing 1 to any bit of this field sets the proper SPISSx0/1 line to an active state and writing 0 sets the line back to inactive state. If the AUTOSS bit is set, writing 0 to any bit location of this field will keep the corresponding SPI_SS0/1 line at inactive state; writing 1 to any bit location of this field will select appropriate SPI_SS0/1 line to be automatically driven to active state for the duration of the transmit/receive, and will be driven to inactive state for the rest of the time. The active state of SPI_SS0/1 is specified in SSLVL. Note: SPI_SS0 is defined as the slave select input in Slave mode.

SPI DMA Control Register (SPI_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPI_PDMACTL	SPIx_BA + 0x0C	R/W	SPI PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:3]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	PDMARST	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logic of the SPI controller. This bit will be cleared to 0 automatically.
[1]	RXPDMAEN	**Receive PDMA Enable** Setting this bit to 1 will start the receive PDMA process. The SPI controller will issue request to PDMA controller automatically when the SPI receive buffer is not empty. This bit will be cleared to 0 by hardware automatically after PDMA transfer is done.
[0]	TXPDMAEN	**Transmit DMA Enable** Setting this bit to 1 will start the transmit PDMA process. SPI controller will issue request to PDMA controller automatically. Hardware will clear this bit to 0 automatically after PDMA transfer done.

SPI FIFO Control Register (SPI_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
SPI_FIFOCTL	SPIx_BA + 0x10	R/W	FIFO Control/Status Register	0x4400_0000

31	30	29	28	27	26	25	24
Reserved	**TXTH**			**Reserved**	**RXTH**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXUDFIEN	**TXUDFPOL**	**RXOVIEN**	**RXTOIEN**	**TXTHIEN**	**RXTHIEN**	**TXRST**	**RXRST**

Bits	Description	
[31]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[30:28]	**TXTH**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TXTH setting, the TXTHIF bit will be set to 1, else the TXTHIF bit will be cleared to 0. 000: 1 word will transmit 001: 2 word will transmit 010: 3 word will transmit 011: 4 word will transmit 100: 5 word will transmit 101: 6 word will transmit 110: 7 word will transmit 111: 8 word will transmit
[27]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[26:24]	**RXTH**	**Receive FIFO Threshold** If the valid data count of the receive FIFO buffer is larger than the RXTH setting, the RXTHIF bit will be set to 1, else the RXTHIF bit will be cleared to 0. 000: 1 word will transmit 001: 2 word will transmit 010: 3 word will transmit 011: 4 word will transmit 100: 5 word will transmit 101: 6 word will transmit 110: 7 word will transmit 111: 8 word will transmit
[23:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**TXUDFIEN**	**Slave Transmit Under Run Interrupt Enable** 0 = Slave Transmit FIFO under-run interrupt Disabled. 1 = Slave Transmit FIFO under-run interrupt Enabled.
[6]	**TXUDFPOL**	**Transmit Under-run Data Out** 0 = The SPI data out is 0 if there is transmit under-run event in Slave mode. 1 = The SPI data out is 1 if there is transmit under-run event in Slave mode. Note: The under run event is active after the serial clock input and the hardware synchronous, so that the first 1~3 bit (depending on the relation between system clock and the engine clock) data out will be the last transaction data. Note: If the frequency of system clock approach the engine clock, they may be a 3-bit time to report the transmit under-run data out.
[5]	**RXOVIEN**	**Receive FIFO Overrun Interrupt Enable** 0 = Receive FIFO overrun interrupt Disabled. 1 = Receive FIFO overrun interrupt Enabled.
[4]	**RXTOIEN**	**Slave Receive Time-out Interrupt Enable** 0 = Receive time-out interrupt Disabled. 1 = Receive time-out interrupt Enabled.
[3]	**TXTHIEN**	**Transmit FIFO Threshold Interrupt Enable** 0 = TX FIFO threshold interrupt Disabled. 1 = TX FIFO threshold interrupt Enabled.
[2]	**RXTHIEN**	**Receive FIFO Threshold Interrupt Enable** 0 = RX FIFO threshold interrupt Disabled. 1 = RX FIFO threshold interrupt Enabled.

[1]	**TXRST**	**Clear Transmit FIFO Buffer** 0 = No effect. 1 = Clear transmit FIFO buffer. The TXFULL bit will be cleared to 0 and the TXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clocks + 3 SPI engine clock after it is set to 1. Note: If there is slave receive time out event, the TXRST will be set 1 when the SPI_SSCTL.SLVTORST, is enabled.
[0]	**RXRST**	**Clear Receive FIFO Buffer** 0 = No effect. 1 = Clear receive FIFO buffer. The RXFULL bit will be cleared to 0 and the RXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clocks + 3 SPI engine clock after it is set to 1. Note: If there is slave receive time out event, the RXRST will be set 1 when the SPI_SSCTL.SLVTORST, is enabled.

SPI Status Register (SPI_STATUS)

Register	Offset	R/W	Description	Reset Value
SPI_STATUS	SPIx_BA + 0x14	R/W	Status Register	0x0005_0110

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	SLVTOIF	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**FIFO CLR Status (Read Only)** 0 = Done the FIFO buffer clear function of TXRST and RXRST. 1 = Doing the FIFO buffer clear function of TXRST or RXRST. Note: Both the TXRST, RXRST, need 3 system clock + 3 engine clocks, the status of this bit allows the user to monitor whether the clear function is busy or done.
[22:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[19]	**TXUFIF**	**Slave Transmit FIFO Under-run Interrupt Status (Read Only)** When the transmit FIFO buffer is empty and further serial clock pulses occur, data transmitted will be the value of the last transmitted bit and this under-run bit will be set. Note: This bit will be cleared by writing 1 to itself.

[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count of the transmit FIFO buffer is larger than the setting value of TXTH. 1 = The valid data count of the transmit FIFO buffer is less than or equal to the setting value of TXTH. Note: If TXTHIEN = 1 and TXTHIF = 1, the SPI controller will generate a SPI interrupt request.
[17]	**TXFULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.
[16]	**TXEMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[15]	**SPIENSTS**	**SPI Enable Bit Status (Read Only)** 0 = Indicate the transmit control bit is disabled. 1 = Indicate the transfer control bit is active. Note: The clock source of SPI controller logic is engine clock, it is asynchronous with the system clock. In order to make sure the function is disabled in SPI controller logic, this bit indicates the real status of SPIEN in SPI controller logic for user.
[14:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12]	**RXTOIF**	**Receive Time-out Interrupt Status** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI clock period in Master mode or over 576 SPI engine clock period in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. Note: This bit will be cleared by writing 1 to itself.
[11]	**RXOVIF**	**Receive FIFO Overrun Status** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. Note: This bit will be cleared by writing 1 to itself.
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the Rx FIFO buffer is smaller than or equal to the setting value of RXTH. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RXTH. Note: If RXTHIEN = 1 and RXTHIF = 1, the SPI controller will generate a SPI interrupt request.

[9]	**RXFULL**	**Receive FIFO Buffer Full Indicator (Read Only)** 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[8]	**RXEMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[7]	**SLVURIF**	**Slave Mode Error 1 Interrupt Status (Read Only)** In Slave mode, transmit under-run occurs when the slave select line goes to inactive state. 0 = No Slave mode error 1 event. 1 = Slave mode error 1 occurs.
[6]	**SLVBEIF**	**Slave Mode Error 0 Interrupt Status (Read Only)** In Slave mode, there is bit counter mismatch with DWIDTH when the slave select line goes to inactive state. 0 = No Slave mode error 0 event. 1 = Slave mode error 0 occurs. Note: If the slave select active but there is no any serial clock input, the SLVBEIF also active when the slave select goes to inactive state.
[5]	**SLVTOIF**	**Slave Time-out Interrupt Status (Read Only)** When the Slave Select is active and the value of SLVTOCNT is not 0 and the serial clock input, the slave time-out counter in SPI controller logic will be start. When the value of time-out counter greater or equal than the value of SPI_SSCTL.SLVTOCNT, during before one transaction done, the slave time-out interrupt event will active. 0 = Slave time-out is not active. 1 = Slave time-out is active. Note: If the DWIDTH is set 16, one transaction is equal 16 bits serial clock period.
[4]	**SSLINE**	**Slave Select Line Bus Status (Read Only)** 0 = Indicates the slave select line bus status is 0. 1 = Indicates the slave select line bus status is 1. Note: If SPI_SSCTL.SSLVL is set 0, and the SSLINE is 1, the SPI slave select is in inactive status.
[3]	**SSINAIF**	**Slave Select Inactive Interrupt Status** 0 = Slave select inactive interrupt is clear or not occur. 1 = Slave select inactive interrupt event has occur. Note: This bit will be cleared by writing 1 to itself.
[2]	**SSACTIF**	**Slave Select Active Interrupt Status** 0 = Slave select active interrupt is clear or not occur. 1 = Slave select active interrupt event has occur. Note: This bit will be cleared by writing 1 to itself.

[1]	**UNITIF**	**Unit Transfer Interrupt Status** 0 = No transaction has been finished since this bit was cleared to 0. 1 = SPI controller has finished one unit transfer. Note: This bit will be cleared by writing 1 to itself.
[0]	**BUSY**	**SPI Unit Bus Status (Read Only)** 0 = No transaction in the SPI bus. 1 = SPI controller unit is in busy state. The following listing are the bus busy conditions: SPIEN = 1 and the TXEMPTY = 0. For SPI Master, the TXEMPTY = 1 but the current transaction is not finished yet. For SPI Slave receive mode, the SPIEN = 1 and there is serial clock input into the SPI core logic when slave select is active. For SPI Slave transmit mode, the SPIEN = 1 and the transmit buffer is not empty in SPI core logic event if the slave select is inactive.

SPI Receive Transaction Count (SPI_RXTSNCNT)

Register	Offset	R/W	Description	Reset Value
SPI_RXTSNCNT	SPIx_BA + 0x18	R/W	Receive Transaction Count Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**RXTSNCNT**	**DMA Receive Transaction Count** When using DMA to receive SPI data without transmitting data, this register can be used in conjunction with the control bit SPI_CTL.RXTCNTEN to set number of transactions to perform. Without this, the SPI interface will only initiate a transaction when it receives a request from the DMA system, resulting in a lower achievable data rate.

SPI Data Transmit Register (SPI_TX)

Register	Offset	R/W	Description	Reset Value
SPI_TX	SPIx_BA + 0x20	W	FIFO Data Transmit Register	0x0000_0000

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

Bits	Description	
[31:0]	**TX**	**Data Transmit Register** A write to the data transmit register pushes data onto into the 8-level transmit FIFO buffer. The number of valid bits depends on the setting of transmit bit width field of the SPI_CTL register. For example, if DWIDTH is set to 0x08, the bits TX[7:0] will be transmitted. If DWIDTH is set to 0, the SPI controller will perform a 32-bit transfer.

SPI Data Receive Register (SPI_RX)

Register	Offset	R/W	Description	Reset Value
SPI_RX	SPIx_BA + 0x30	R	FIFO Data Receive Register	0x0000_0000

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

Bits	Description	
[31:0]	**RX**	**Data Receive Register** A read from this register pops data from the 8-level receive FIFO. Valid data is present if the SPI_STATUS. RXEMPTY bit is not set to 1. This is a read-only register.

5.8 I²S Controller (I²S0)

5.8.1 Overview

The I2S controller consists of I2S protocol to interface with external audio CODEC. There is one set of I2S controller, with two 16-level depth FIFO for reading path and writing path respectively and is capable of handling 8/16/24/32 bits audio data sizes. I2S controller supports the PDMA controller handles the data movement between FIFO and memory.

5.8.2 Features

- Support Master mode and Slave mode
- Capable of handling 8, 16, 24 and 32 bits data sizes in each audio channel
- Supports monaural and stereo audio data
- Supports I²S protocols: Philips standard, MSB-justified, and LSB-justified data format
- Supports PCM protocols: PCM standard, MSB-justified, and LSB-justified data format
- Each I²S controller provides two 16-level FIFO data buffers, one for transmitting and the other for receiving
- Generates interrupt requests when buffer levels cross a programmable boundary
- I²S controller supports two PDMA requests, one for transmitting and the other for receiving

5.8.3 Block Diagram

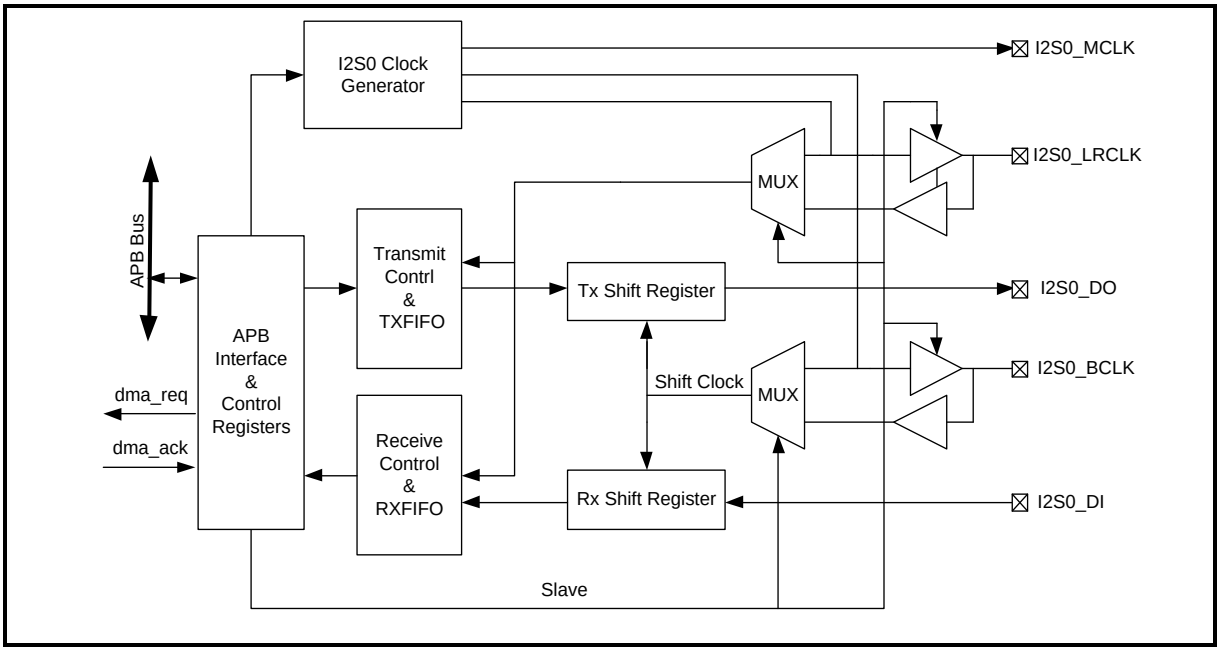


Figure 5.8-1 I2S0 Block Diagram

5.8.4 Functional Description

5.8.4.1 I²S Clock

The I2S controller has five clock sources selected by I2S0SEL (CLK_CLKSEL1[22:20]). The I2S clock rate must be slower than or equal to system clock rate.

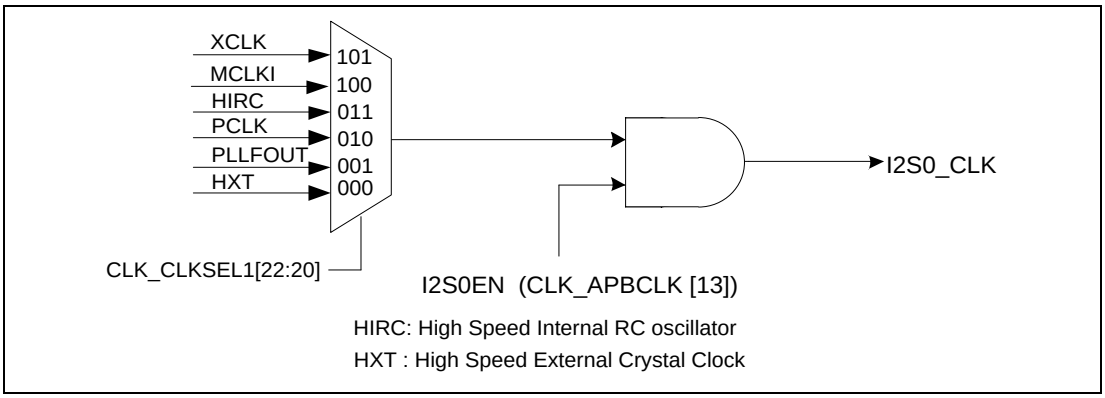


Figure 5.8-2 I²S0 Clock Control Diagram

5.8.4.2 Master/Slave Interface

The I2S function can operate as master or slave mode by setting SLAVE (I2S0_CTL0[8]) to communicate with other I2S slave or master. The serial bus clock I2S_BCLK is permanently generated by the master even through there is no transferring data bit at the moment. The word select signal I2S_LRCLK is also generated by the master and it indicates the beginning of a new data word and the targeted audio channel. Both the I2S_LRCLK and the transmitting data change synchronously to the falling edges of I2S_BCLK.

In some applications, especially for Audio-ADC or Audio-DAC, a master clock signal, I2S_MCLK, is required with a fixed phase relation to the I2S_BCLK. The I2S_MCLK is enabled by MCLKEN (I2S0_CTL0[15]). In Master mode, the I2S_MCLK, I2S_BCLK, I2S_LRCLK is output to device slave. And if in slave mode, the I2S_MCLK is output to master, and I2S_BCLK or I2S_LRCLK is input from master.

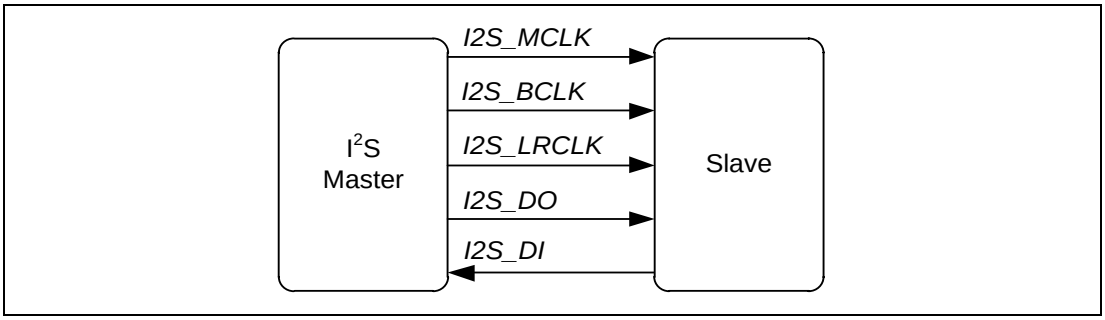


Figure 5.8-3 Master mode Interface Block Diagram

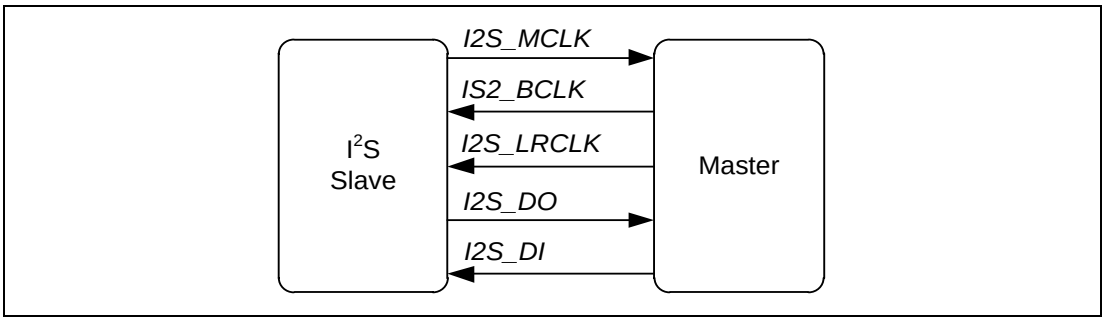


Figure 5.8-4 Slave mode Interface Block Diagram

5.8.4.3 I²S Operation

The I2S controller supports MSB-justified, LSB-justified, and I2S Philips standard data format. The I2S_LRCLK signal indicates which audio channel is in transferring. The bit count of an audio channel is defined by CHWIDTH (I2S0_CTL0[29:28]), and the bit-width of data word in an audio channel is determined by DATWIDTH (I2S0_CTL0[5:4]). If CHWIDTH (I2S0_CTL0[29:28]) is less than DATWIDTH (I2S0_CTL0[5:4]), the hardware will set the channel bit-width to be same as data bit-width. However, there will be redundant zero bits in each audio channel if CHWIDTH (I2S0_CTL0[29:28]) is greater than DATWIDTH (I2S0_CTL0[5:4]).

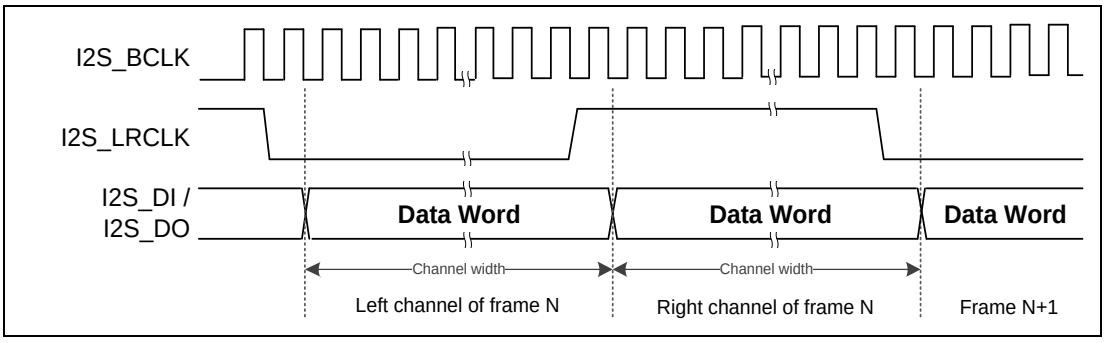


Figure 5.8-5 I²S Channel Width and Data Width (CHWIDTH≦DATWIDTH)

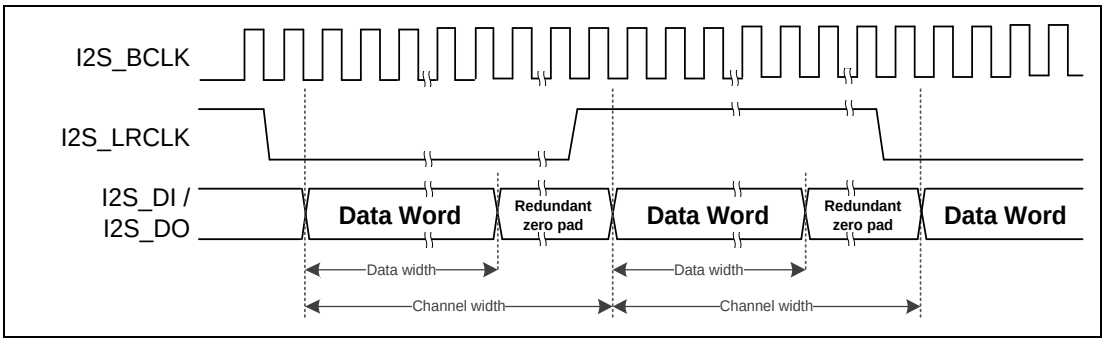


Figure 5.8-6 I²S Channel Width and Data Width (CHWIDTH > DATWIDTH)

The transferring data sequence is always started from the MSB (most significance bit) to the LSB (least significance bit). As the Figure 5.8-7 I2S Data Format Timing Diagram (FORMAT = 0x0 ; CHWIDTH≦DATWIDTH)

, transmitting data are read at rising edge of I2S_BCLK and sent out at falling edge of I2S_BCLK in I2S protocol.

In I2S data format, the MSB is sent and latched at the next falling edge of I2S_BCLK cycle after the transition of I2S_LRCLK. In MSB justified data format, the I2S_LRCLK changes the polarity at the transmitting of the first data bit (MSB) in each audio channel. In LSB justified data format, the LSB is sent and latched at the last I2S_BCLK cycle of an audio channel. The MSB justified and LSB justified data format of I2S protocol can be selected by FORMAT (I2S0_CTL0[26:24]).

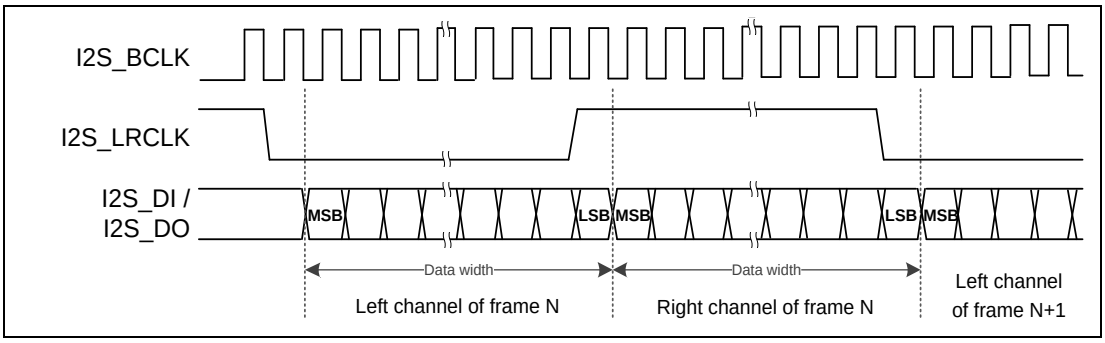


Figure 5.8-7 I²S Data Format Timing Diagram (FORMAT = 0x0 ; CHWIDTH≦DATWIDTH)

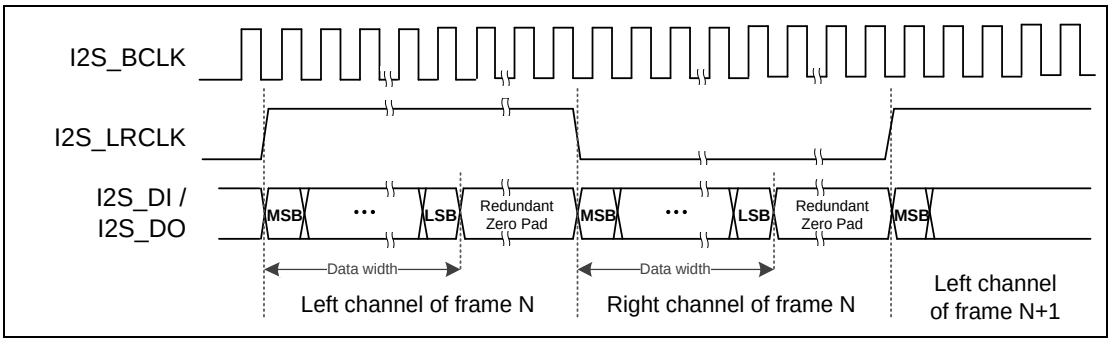


Figure 5.8-8 MSB Justified Data Format (FORMAT = 0x1 ; CHWIDTH > DATWIDTH)

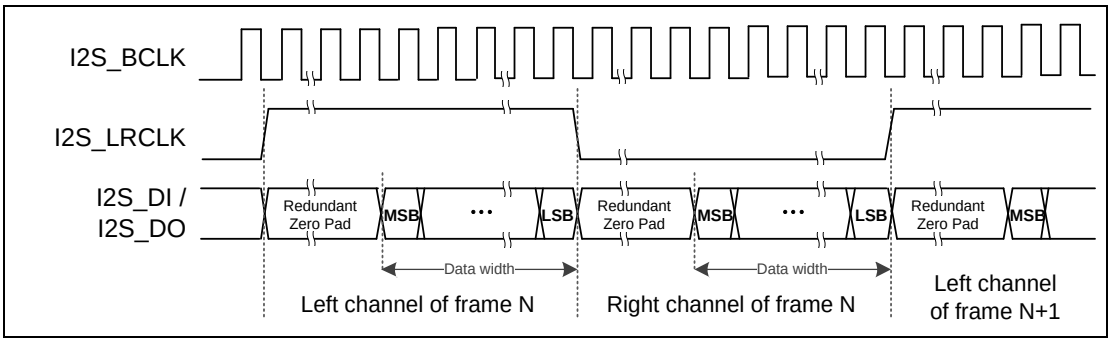


Figure 5.8-9 LSB Justified Data Format (FORMAT = 0x2 ; CHWIDTH > DATWIDTH)

The I2S controller also supports PCM audio transmission which can be selected by FORMAT (I2S0_CTL0[26:24]). In PCM protocol, the function of I2S_LRCLK is simply to identify the beginning of an audio sample (or audio frame) and it is always indicated by the rising edge of the pulse. Therefore, the I2S_LRCLK in PCM protocol may be also called “frame start” or “frame sync” signal. In master device, there are two common representations for the width of the frame start pulse which can use PCMSYNC (I2S0_CTL0[27]) to choose: One is equivalent to the period of a channel width and the other is equivalent to a single period of the I2S_BCLK.

Same as I2S protocol, the DATWIDTH (I2S0_CTL0[5:4]) and CHWIDTH (I2S0_CTL0[29:28]) can be used to

configure the data bit-width and channel bit-width in PCM protocol. Besides, FORMAT (I2S0_CTL0[26:24]) can also be used to select the different data formats of PCM standard mode, PCM with MSB justified, and PCM with LSB justified data format.

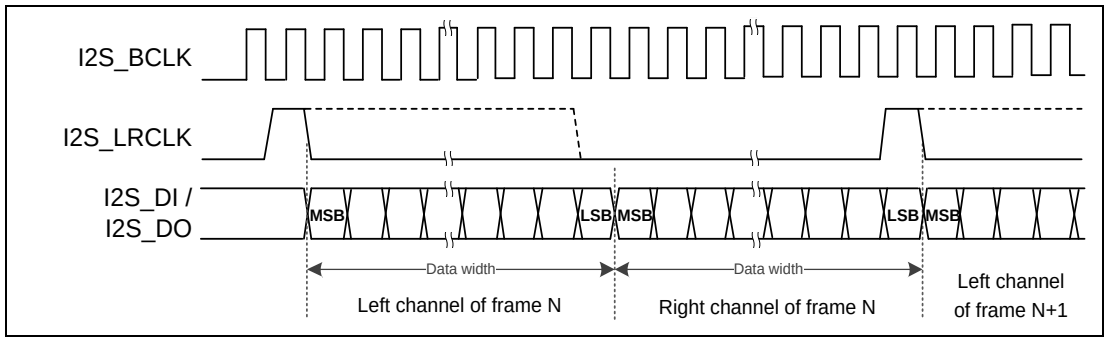


Figure 5.8-10 Standard PCM Audio Timing Diagram (FORMAT = 0x4 ; CHWIDTH≦DATWIDTH)

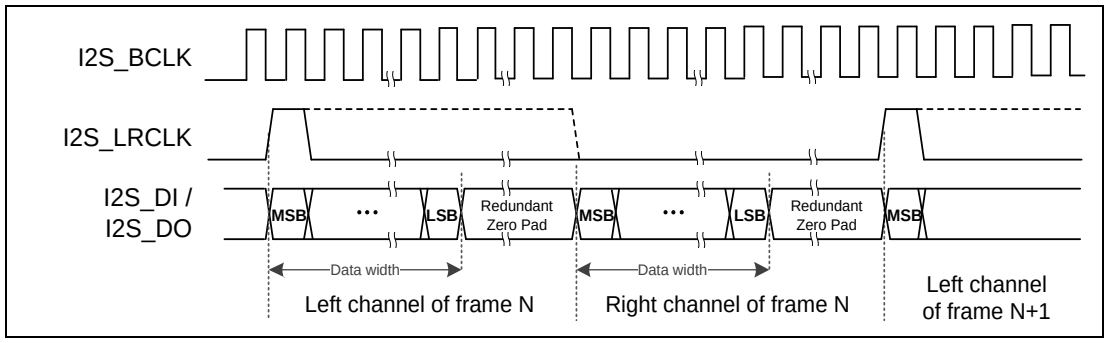


Figure 5.8-11 PCM with MSB Justified Data Format (FORMAT = 0x5 ; CHWIDTH > DATWIDTH)

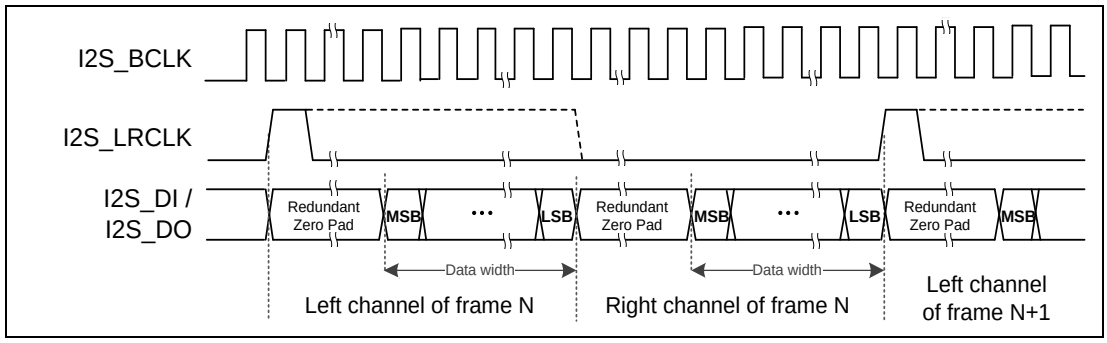


Figure 5.8-12 PCM with LSB Justified Data Format (FORMAT = 0x6 ; CHWIDTH > DATWIDTH)

5.8.4.4 Zero Crossing

When playing the audio by I2S controller, the output transmitting data comes from the memory by PDMA or by CPU. However, there may be some pop noise which induces the uncomfortable hearing if the playing sound volume is changed greatly by user. The zero-crossing event of audio data means the playing sound is relatively silent at the moment. Therefore, the zero-cross interrupt can be used for the indication of gain level adjustment in order to prevent the huge variance of sound volume.

If zero-cross detection of individual audio channel is enabled by the corresponding control bit CH0ZCEN and CH1ZCEN (I2S0_CTL1[0] and I2S0_CTL1[1]), the hardware will detect the next transferring data word of the corresponding audio channel whether it is zero or its MSB has been changed. If zero value or MSB (sign bit) changing of the transmitting audio data has been detected while zero-cross detection is enabled, the hardware will set the corresponding status bit CH0ZCIF and CH1ZCIF (I2S0_STATUS1[0] and I2S0_STATUS1[1]) for the audio channel and then keep the output audio data silent (all data bit zero) automatically until the corresponding event status bit is cleared by software.

Therefore, if user wants to modify the audio playing gain, users can enable the zero crossing interrupt function, CH0ZCIEN and CH1ZCIEN (I2S0_IEN[16:17]) , to indicate the zero crossing time and to change the audio gain. This will reduce the pop noise.

5.8.4.5 PDMA Mode

The I2S function can use PDMA function for transmitting or receiving data access. If the PDMA function of transmitting data is enabled by TXPDMAEN (I2S0_CTL0[20]), the I2S controller will generate the request signal and then get transmitting audio data from memory by PDMA IP automatically while TX FIFO is not full. If the PDMA function of receiving data is enabled by RXPDMAEN (I2S0_CTL0[19]), the I2S controller will generate the request signal and then the receiving data will be moved into memory by PDMA hardware automatically while the RX FIFO is not empty. Therefore, using PDMA function will save the CPU loading to service other functions.

5.8.4.6 I²S Interrupt Sources

The I2S controller supports zero-cross interrupt of individual audio channel, transmit FIFO threshold level interrupt, transmit FIFO overflow interrupt and transmit FIFO underflow interrupt in transmit operation. In receive operation, it supports receive FIFO threshold level interrupt, receive FIFO overflow interrupt and receive FIFO underflow interrupt. When I2S interrupt occurs, user can check I2STXINT (I2S0_STATUS0[2]) and I2SRXINT (I2S0_STATUS0[1]) flags to recognize the interrupt sources.

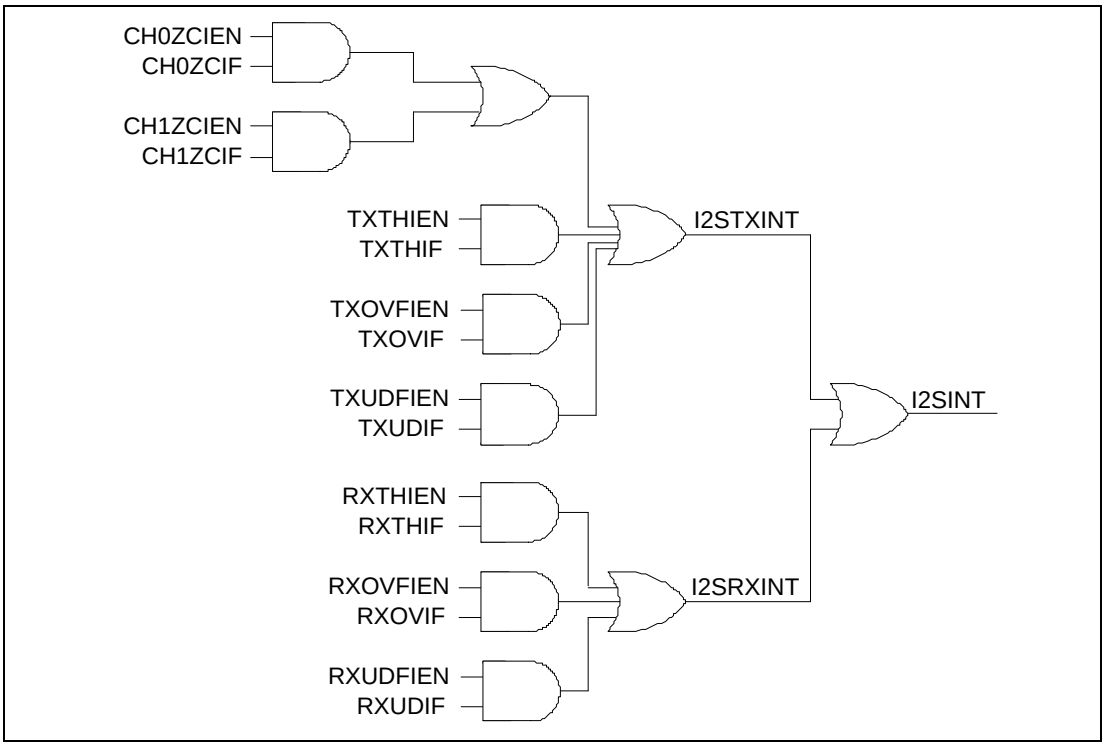


Figure 5.8-13 I²S Interrupts

5.8.4.7 **FIFO Operation**

In 2-channel I2S or PCM protocol, the bit-width of audio data in a channel block can be 8, 16, 24, or 32 bits. The memory arrangements of audio data for various settings are shown in Figure 5.8-14 FIFO Contents for Various 2-channel Audio Modes

.

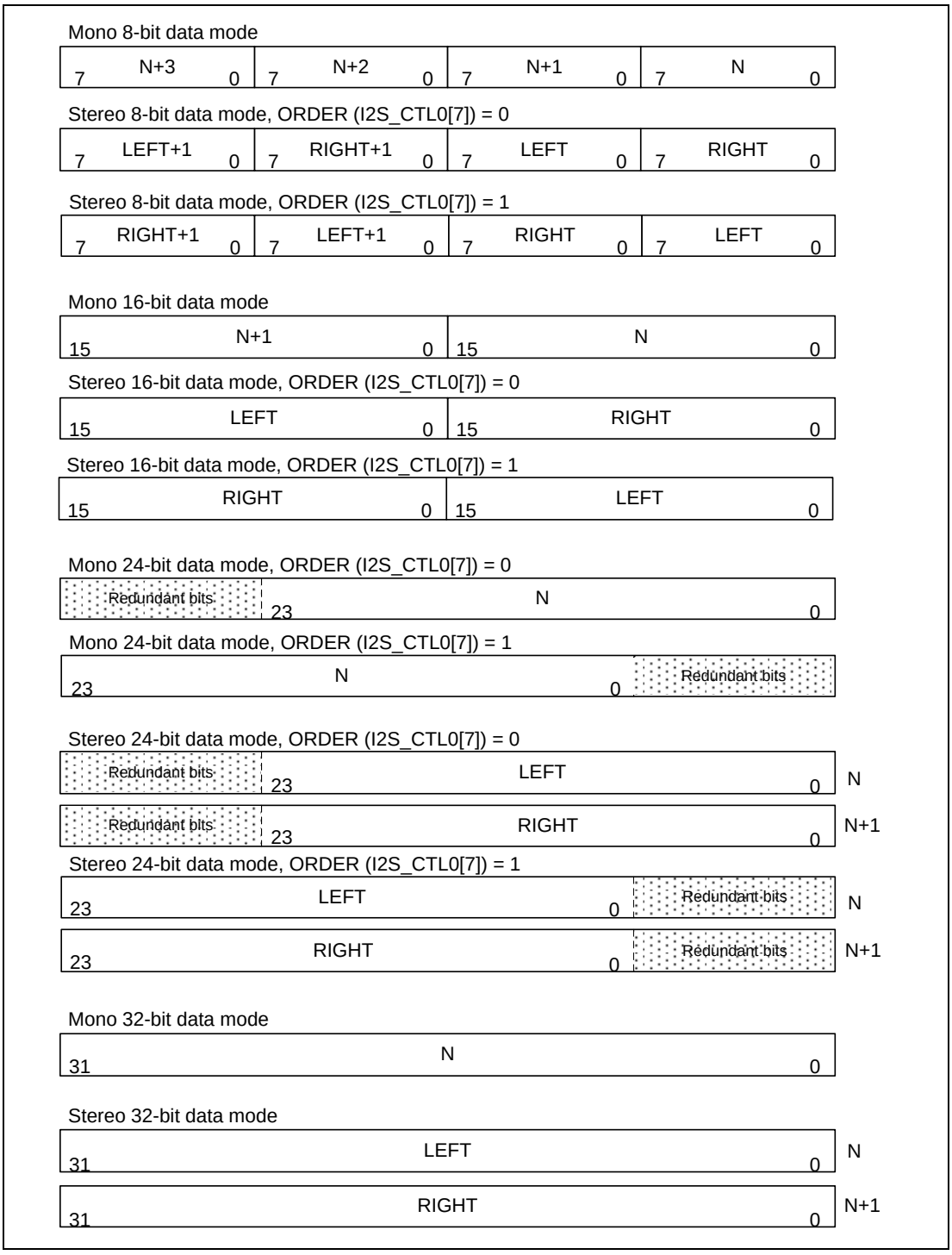


Figure 5.8-14 FIFO Contents for Various 2-channel Audio Modes

5.8.5 I2S Control Register Map

R: Read only, **W**: Write only, **R/W**: Both read and write

Register	Offset	R/W	Description	Reset Value
I2S0 Base Address **I2S0_BA = 0x4009_0000**				
I2S0_CTL0	I2S0_BA+0x00	R/W	I^2S0 Control Register 0	0x0000_0000
I2S0_CTL1	I2S0_BA+0x20	R/W	I^2S0 Control Register 1	0x0000_0000
I2S0_CLKDIV	I2S0_BA+0x04	R/W	I^2S0 Clock Divider Register	0x0000_0000
I2S0_IEN	I2S0_BA+0x08	R/W	I^2S0 Interrupt Enable Register	0x0000_0000
I2S0_STATUS0	I2S0_BA+0x0C	R/W	I^2S0 Status Register 0	0x0014_1000
I2S0_STATUS1	I2S0_BA+0x24	R/W	I^2S0 Status Register 1	0x0000_0000
I2S0_TXFIFO	I2S0_BA+0x10	W	I^2S0 Transmit FIFO Register	0x0000_0000
I2S0_RXFIFO	I2S0_BA+0x14	R	I^2S0 Receive FIFO Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

5.8.6 I2S Control Register Description

I2S0 Control Register 0 (I2S0_CTL0)

Register	Offset	R/W	Description	Reset Value
I2S0_CTL0	I2S0_BA+0x00	R/W	I^2S0 Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		**CHWIDTH**		**Reserved**	**FORMAT**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXLCH	**Reserved**	**RXPDMAEN**	**TXPDMAEN**	**RXFBCLR**	**TXFBCLR**	**FLZCDEN**	**FRZCDEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MCLKEN	**Reserved**						**SLAVE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ORDER	**MONO**	**DATWIDTH**		**MUTE**	**RXEN**	**TXEN**	**I2SEN**

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29:28]	**CHWIDTH**	**Channel Width** This bit fields are used to define the length of audio channel. If CHWIDTH < DATWIDTH, the hardware will set the real channel length as the bit-width of audio data which is defined by DATWIDTH. 00 = The bit-width of each audio channel is 8-bit. 01 = The bit-width of each audio channel is 16-bit. 10 = The bit-width of each audio channel is 24-bit. 11 = The bit-width of each audio channel is 32-bit.
[27]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[26:24]	**FORMAT**	**Data Format Selection** 000 = I^2S standard data format. 001 = I^2S with MSB justified. 010 = I^2S with LSB justified. 011 = Reserved. Do not use. 100 = PCM standard data format. 101 = PCM with MSB justified. 110 = PCM with LSB justified. 111 = Reserved. Do not use.

[23]	**RXLCH**	**Receive Left Channel Enable Control** When monaural format is selected (MONO = 1), I^{2}S will receive channel1 data if RXLCH is set to 0, and receive channel0 data if RXLCH is set to 1. 0 = Receives channel1 data in MONO mode. 1 = Receives channel0 data in MONO mode.
[22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**RXPDMAEN**	**Receive PDMA Enable Control** 0 = Receiver PDMA function Disabled. 1 = Receiver PDMA function Enabled.
[20]	**TXPDMAEN**	**Transmit PDMA Enable Control** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.
[19]	**RXFBCLR**	**Receive FIFO Buffer Clear** 0 = No Effect. 1 = Clear RX FIFO. **Note1:** Write 1 to clear receive FIFO, internal pointer is reset to FIFO start point, and RXCNT (I2S0_STATUS1[20:16]) returns 0 and receive FIFO becomes empty. **Note2:** This bit is cleared by hardware automatically, read it return zero.
[18]	**TXFBCLR**	**Transmit FIFO Buffer Clear** 0 = No Effect. 1 = Clear TX FIFO. **Note1:** Write 1 to clear transmit FIFO, internal pointer is reset to FIFO start point, and TXCNT (I2S0_STATUS1[12:8]) returns 0 and transmit FIFO becomes empty but data in transmit FIFO is not changed. **Note2:** This bit is clear by hardware automatically, read it return zero.
[17]	**FLZCDEN**	**Force Left Channel Zero Cross Data Option Bit** If this bit set to 1, when channel data sign bit changes or next shift data bits are all 0 then the channel ZCIF flag in I2S0_STATUS1 register is set to 1 and channel data will force zero. This function is only available in transmit operation. 0 = Keep channel data, when zero crossing flag on. 1 = Force channel data to zero, when zero crossing flag on.
[16]	**FRZCDEN**	**Force Right Channel Zero Cross Data Option Bit** If this bit set to 1, when channel data sign bit changes or next shift data bits are all 0 then the channel ZCIF flag in I2S0_STATUS1 register is set to 1 and channel data will force zero. This function is only available in transmit operation. 0 = Keep channel data, when zero crossing flag on. 1 = Force channel data to zero, when zero crossing flag on.

[15]	**MCLKEN**	**Master Clock Enable Control** If MCLKEN is set to 1, I²S controller will generate master clock on I2S_MCLK pin for external audio devices. 0 = Master clock Disabled. 1 = Master clock Enabled.
[14:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**SLAVE**	**Slave Mode Enable Control** 0 = Master mode. 1 = Slave mode. **Note:** I²S can operate as master or slave. For Master mode, I2S_BCLK and I2S_LRCLK pins are output mode and send out bit clock to Audio CODEC chip. In Slave mode, I2S_BCLK and I2S_LRCLK pins are input mode and I2S_BCLK and I2S_LRCLK signals are received from outer Audio CODEC chip.
[7]	**ORDER**	**Stereo Data Order in FIFO** In 8-bit/16-bit data width, this bit is used to select whether the even or odd channel data is stored in higher byte. In 24-bit data width, this is used to select the left/right alignment method of audio data which is stored in data memory consisted of 32-bit FIFO entries. 0 = Even channel data at high byte in 8-bit/16-bit data width. LSB of 24-bit audio data in each channel is aligned to right side in 32-bit FIFO entries. 1 = Even channel data at low byte. MSB of 24-bit audio data in each channel is aligned to left side in 32-bit FIFO entries.
[6]	**MONO**	**Monaural Data Control** 0 = Data is stereo format. 1 = Data is monaural format. **Note:** when chip records data, RXLCH (I2S0_CTL0[23]) indicates which channel data will be saved if monaural format is selected.
[5:4]	**DATWIDTH**	**Data Width** This bit field is used to define the bit-width of data word in each audio channel 00 = The bit-width of data word is 8-bit. 01 = The bit-width of data word is 16-bit. 10 = The bit-width of data word is 24-bit. 11 = The bit-width of data word is 32-bit.
[3]	**MUTE**	**Transmit Mute Enable Control** 0 = Transmit data is shifted from buffer. 1 = Send zero on transmit channel.

[2]	**RXEN**	**Receive Enable Control** 0 = Data receiving Disabled. 1 = Data receiving Enabled.
[1]	**TXEN**	**Transmit Enable Control** 0 = Data transmission Disabled. 1 = Data transmission Enabled.
[0]	**I2SEN**	**I²S Controller Enable Control** 0 = I²S controller Disabled. 1 = I²S controller Enabled.

I2S0 Control Register 1 (I2S0_CTL1)

Register	Offset	R/W	Description	Reset Value
I2S0_CTL1	I2S0_BA+0x20	R/W	I^2S0 Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						**PB16ORD**	**PBWIDTH**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**RXTH**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**TXTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**CH1ZCEN**	**CH0ZCEN**

Bits	Description	
[31:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25]	**PB16ORD**	**FIFO Read/Write Order in 16-bit Width of Peripheral Bus** When PBWIDTH = 1, the data FIFO will be increased or decreased by two peripheral bus access. This bit is used to select the order of FIFO access operations to meet the 32-bit transmitting/receiving FIFO entries. 0 = Low 16-bit read/write access first. 1 = High 16-bit read/write access first. **Note:** This bit is available while PBWIDTH = 1.
[24]	**PBWIDTH**	**Peripheral Bus Data Width Selection** This bit is used to choice the available data width of APB bus. It must be set to 1 while PDMA function is enable and it is set to 16-bit transmission mode 0 = 32 bits data width. 1 = 16 bits data width. **Note1:** If PBWIDTH=1, the low 16 bits of 32-bit data bus are available. **Note2:** If PBWIDTH=1, the transmitting FIFO level will be increased after two FIFO write operations. **Note3:** If PBWIDTH=1, the receiving FIFO level will be decreased after two FIFO read operations.
[23:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[19:16]	**RXTH**	**Receive FIFO Threshold Level** 0000 = 1 data word in receive FIFO. 0001 = 2 data words in receive FIFO. 0010 = 3 data words in receive FIFO. …. 1110 = 15 data words in receive FIFO. 1111 = 16 data words in receive FIFO. **Note:** When received data word number in receive buffer is greater than threshold level then RXTHIF (I2S0_STATUS0[10]) flag is set.
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**TXTH**	**Transmit FIFO Threshold Level** 0000 = 0 data word in transmit FIFO. 0001 = 1 data word in transmit FIFO. 0010 = 2 data words in transmit FIFO. .... 1110 = 14 data words in transmit FIFO. 1111 = 15 data words in transmit FIFO. **Note:** If remain data word number in transmit FIFO is the same or less than threshold level then TXTHIF (I2S0_STATUS0[18]) flag is set.
[7:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**CH1ZCEN**	**Channel1 Zero-cross Detect Enable Control** 0 = channel1 zero-cross detect Disabled. 1 = channel1 zero-cross detect Enabled. **Note1:** Channel1 also means right audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode. **Note2:** If this bit is set to 1, when channel1 data sign bit change or next shift data bits are all zero then CH1ZCIF(I2S0_STATUS1[1]) flag is set to 1. **Note3:** If CH1ZCIF Flag is set to 1, the channel1 will be mute.
[0]	**CH0ZCEN**	**Channel0 Zero-cross Detection Enable Control** 0 = channel0 zero-cross detect Disabled. 1 = channel0 zero-cross detect Enabled. **Note1:** Channel0 also means left audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode. **Note2:** If this bit is set to 1, when channel0 data sign bit change or next shift data bits are all zero then CH0ZCIF(I2S0_STATUS1[0]) flag is set to 1. **Note3:** If CH0ZCIF Flag is set to 1, the channel0 will be mute.

I2S0 Clock Divider (I2S0_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2S0_CLKDIV	I2S0_BA+0x04	R/W	I²S0 Clock Divider Register	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17:8]	**BCLKDIV**	**Bit Clock Divider** The I²S controller will generate bit clock in Master mode. Software can program these bit fields to generate sampling rate clock frequency. F_BCLK= F_I2SCLK / (2*(BCLKDIV + 1)). **Note:** F_BCLK is the frequency of BCLK and F_I2SCLK is the frequency of I2S_CLK
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6:0]	**MCLKDIV**	**Master Clock Divider** If chip external crystal frequency is (2xMCLKDIV)*256fs then software can program these bits to generate 256fs clock frequency to audio codec chip. If MCLKDIV is set to 0, MCLK is the same as external clock input. For example, sampling rate is 24 kHz and chip external crystal clock is 12.288 MHz, set MCLKDIV = 1. F_MCLK = F_I2SCLK/(2x(MCLKDIV)) (When MCLKDIV is >= 1). F_MCLK = F_I2SCLK (When MCLKDIV is set to 0). **Note:** F_MCLK is the frequency of MCLK, and F_I2SCLK is the frequency of the I2S_CLK

I2S0 Interrupt Enable Register (I2S0_IEN)

Register	Offset	R/W	Description	Reset Value
I2S0_IEN	I2S0_BA+0x08	R/W	I^2S0 Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**CH1ZCIEN**	**CH0ZCIEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**TXTHIEN**	**TXOVFIEN**	**TXUDFIEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**RXTHIEN**	**RXOVFIEN**	**RXUDFIEN**

Bits	Description	
[31:18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17]	**CH1ZCIEN**	**Channel1 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel1 zero-cross **Note2:** Channel1 also means right audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode.
[16]	**CH0ZCIEN**	**Channel0 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel0 zero-cross **Note2:** Channel0 also means left audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode.
[15:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**TXTHIEN**	**Transmit FIFO Threshold Level Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** Interrupt occurs if this bit is set to 1 and data words in transmit FIFO is less than TXTH (I2S0_CTL1[11:8]).

[9]	**TXOVFIEN**	**Transmit FIFO Overflow Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** Interrupt occurs if this bit is set to 1 and TXOVIF (I2S0_STATUS0[17]) flag is set to 1
[8]	**TXUDFIEN**	**Transmit FIFO Underflow Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** Interrupt occur if this bit is set to 1 and TXUDIF (I2S0_STATUS0[16]) flag is set to 1.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**RXTHIEN**	**Receive FIFO Threshold Level Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** When data word in receive FIFO is equal or higher than RXTH (I2S0_CTL1[19:16]) and the RXTHIF (I2S0_STATUS0[10]) bit is set to 1. If RXTHIEN bit is enabled, interrupt occur.
[1]	**RXOVFIEN**	**Receive FIFO Overflow Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** Interrupt occurs if this bit is set to 1 and RXOVIF (I2S0_STATUS0[9]) flag is set to 1
[0]	**RXUDFIEN**	**Receive FIFO Underflow Interrupt E Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** If software reads receive FIFO when it is empty then RXUDIF (I2S0_STATUS0[8]) flag is set to 1.

I2S0 Status Register 0 (I2S0_STATUS0)

Register	Offset	R/W	Description	Reset Value
I2S0_STATUS0	I2S0_BA+0x0C	R/W	I²S0 Status Register 0	0x0014_1000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**TXEMPTY**	**TXFULL**	**TXTHIF**	**TXOVIF**	**TXUDIF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**RXEMPTY**	**RXFULL**	**RXTHIF**	**RXOVIF**	**RXUDIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**DATACH**	**I2STXINT**	**I2SRXINT**	**I2SINT**

Bits	Description	
[31:21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[20]	**TXEMPTY**	**Transmit FIFO Empty (Read Only)** This bit reflect data word number in transmit FIFO is zero 0 = Not empty. 1 = Empty.
[19]	**TXFULL**	**Transmit FIFO Full (Read Only)** This bit reflect data word number in transmit FIFO is 12 0 = Not full. 1 = Full.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Flag (Read Only)** 0 = Data word(s) in FIFO is higher than threshold level. 1 = Data word(s) in FIFO is equal or lower than threshold level. **Note:** When data word(s) in transmit FIFO is equal or lower than threshold value set in TXTH (I2S0_CTL1[11:8]) the TXTHIF bit becomes to 1. It keeps at 1 till TXCNT (I2S0_STATUS1[12:8]) is higher than TXTH (I2S0_CTL1[11:8]) after software write TXFIFO register.
[17]	**TXOVIF**	**Transmit FIFO Overflow Interrupt Flag** 0 = No overflow. 1 = Overflow. **Note1:** Write data to transmit FIFO when it is full and this bit set to 1 **Note2:** Write 1 to clear this bit to 0.

[16]	**TXUDIF**	**Transmit FIFO Underflow Interrupt Flag** 0 = No underflow. 1 = Underflow. **Note1:** This bit will be set to 1 when shift logic hardware read data from transmitting FIFO and the filling data level in transmitting FIFO is not enough for one audio frame. **Note2:** Write 1 to clear this bit to 0.
[15:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12]	**RXEMPTY**	**Receive FIFO Empty (Read Only)** 0 = Not empty. 1 = Empty. **Note:** This bit reflects data words number in receive FIFO is zero
[11]	**RXFULL**	**Receive FIFO Full (Read Only)** 0 = Not full. 1 = Full. **Note:** This bit reflects data words number in receive FIFO is 12.
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Flag (Read Only)** 0 = Data word(s) in FIFO is not higher than threshold level. 1 = Data word(s) in FIFO is higher than threshold level. **Note:** When data word(s) in receive FIFO is higher than threshold value set in RXTH (I2S0_CTL1[19:16]) the RXTHIF bit becomes to 1. It keeps at 1 till RXCNT (I2S0_STATUS1[20:16]) is not higher than RXTH (I2S0_CTL1[19:16]) after software read RXFIFO register.
[9]	**RXOVIF**	**Receive FIFO Overflow Interrupt Flag** 0 = No overflow occur. 1 = Overflow occur. **Note1:** When receive FIFO is full and receive hardware attempt to write data into receive FIFO then this bit is set to 1, data in 1st buffer is overwrote. **Note2:** Write 1 to clear this bit to 0.
[8]	**RXUDIF**	**Receive FIFO Underflow Interrupt Flag** 0 = No underflow occur. 1 = Underflow occur. **Note1:** When receive FIFO is empty, and software reads the receive FIFO again. This bit will be set to 1, and it indicates underflow situation occurs. **Note2:** Write 1 to clear this bit to zero
[7:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[3]	**DATACH**	**Transmission Data Channel (Read Only)** This bit fields are used to indicate which audio channel is current transmit data belong. 0 = channel0 (means left channel while 2-channel I2S/PCM mode). 1 = channel1 (means right channel while 2-channel I2S/PCM mode).
[2]	**I2STXINT**	**I^2S Transmit Interrupt (Read Only)** 0 = No transmit interrupt. 1 = Transmit interrupt.
[1]	**I2SRXINT**	**I^2S Receive Interrupt (Read Only)** 0 = No receive interrupt. 1 = Receive interrupt.
[0]	**I2SINT**	**I^2S Interrupt Flag (Read Only)** 0 = No I^2S interrupt. 1 = I^2S interrupt. **Note:** It is wire-OR of I2STXINT and I2SRXINT bits.

I2S0 Status Register 1 (I2S0_STATUS1)

Register	Offset	R/W	Description	Reset Value
I2S0_STATUS1	I2S0_BA+0x24	R/W	I²S0 Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**RXCNT**				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**TXCNT**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**CH1ZCIF**	**CH0ZCIF**

Bits	Description	
[31:21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[20:16]	**RXCNT**	**Receive FIFO Level (Read Only)** These bits indicate the number of available entries in receive FIFO 00000 = No data. 00001 = 1 word in receive FIFO. 00010 = 2 words in receive FIFO. …. 01110 = 14 words in receive FIFO. 01111 = 15 words in receive FIFO. 10000 = 16 words in receive FIFO. Others are reserved. Do not use.
[15:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[12:8]	**TXCNT**	**Transmit FIFO Level (Read Only)** These bits indicate the number of available entries in transmit FIFO 00000 = No data. 00001 = 1 word in transmit FIFO. 00010 = 2 words in transmit FIFO. …. 01110 = 14 words in transmit FIFO. 01111 = 15 words in transmit FIFO. 10000 = 16 words in transmit FIFO. Others are reserved. Do not use.
[7:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**CH1ZCIF**	**Channel1 Zero-cross Interrupt Flag** It indicates channel1 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel1. 1 = Channel1 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** Channel1 also means right audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode.
[0]	**CH0ZCIF**	**Channel0 Zero-cross Interrupt Flag** It indicates channel0 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel0. 1 = Channel0 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** Channel0 also means left audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode.

I2S0 Transmit FIFO (I2S0_TXFIFO)

Register	Offset	R/W	Description	Reset Value
I2S0_TXFIFO	I2S0_BA+0x10	W	I²S0 Transmit FIFO Register	0x0000_0000

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

Bits	Description	
[31:0]	**TXFIFO**	**Transmit FIFO Bits** I²S contains 16 words (16x32 bit) data buffer for data transmit. Write data to this register to prepare data for transmit. The remaining word number is indicated by TXCNT (I2S0_STATUS1[12:8]).

I2S0 Receive FIFO (I2S0_RXFIFO)

Register	Offset	R/W	Description	Reset Value
I2S0_RXFIFO	I2S0_BA+0x14	R	I^2S0 Receive FIFO Register	0x0000_0000

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

Bits	Description	
[31:0]	**RXFIFO**	**Receive FIFO Bits** I^2S contains 16 words (16x32 bit) data buffer for data receive. Read this register to get data in FIFO. The remaining data word number is indicated by RXCNT (I2S0_STATUS1[20:16]).

5.9 Timer Controller

5.9.1 Overview

ISD91500 timer module includes three channels, Timer0, Timer1 and Timer2 which allow user to easily implement a counting scheme for use. The timer can perform functions like frequency measurement, event counting, interval measurement, clock generation, delay timing, and so on. The timer possesses features such as adjustable resolution, programmable counting period, and detailed information. The timer can generate an interrupt signal upon timeout, or provide the current value of count during operation.

Besides the general timer function, Timer1 also can be used to generate IR carrier output.

5.9.2 Features

The general timer (Timer0/1/2) controller includes the following features

- AMBA APB interface compatible
- Each channel with an 8-bit pre-scale counter, a 16-bit up-counter and an interrupt request signal.
- Independent clock source for each channel.
- Provides one-shot, periodic and continuous counting operation modes
- 16-bit up counter value is readable through CNT register
- Time out period = (Period of timer clock input) * (Prescale + 1) * (TCMP)
- Maximum counting cycle time = (1/49.152M) * (2^8) * (2^16), if TMRx_CLK=49.152 MHz.

5.9.3 Timer Controller Block Diagram

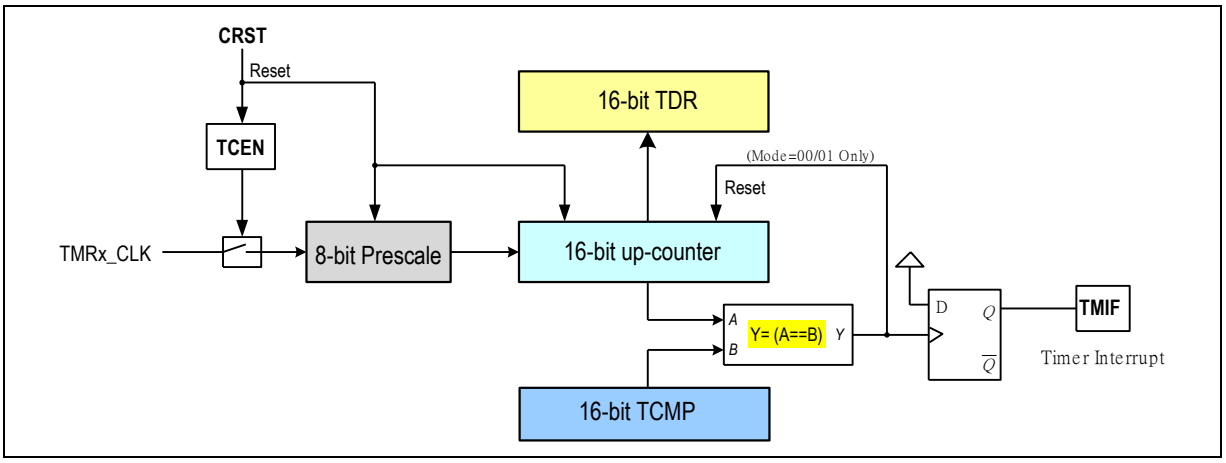


Figure 5.9-1 Timer0/1/2 Block Diagram

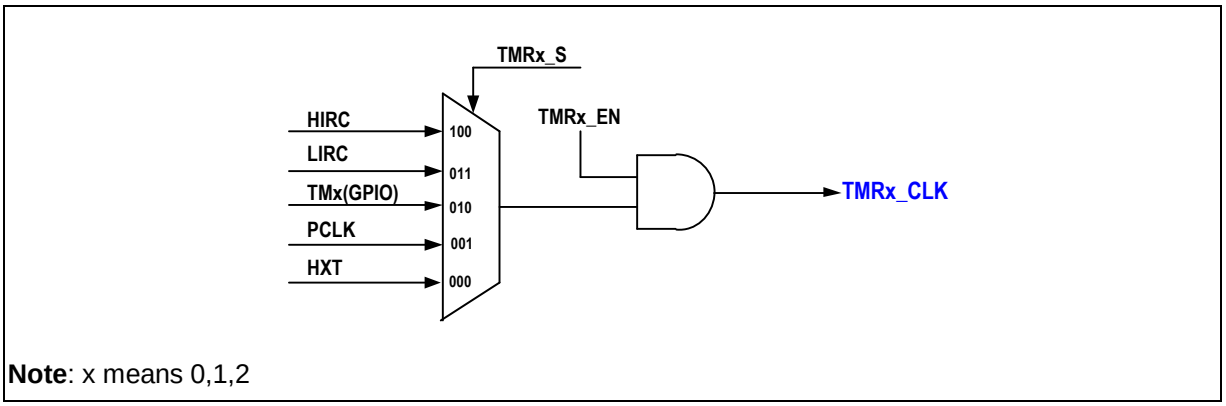


Figure 5.9-2 Clock Source of Timer0/1/2

5.9.4 Timer Functional Description

5.9.4.1 Timer Interrupt Flag

In timer mode, Timer controller can generate the interrupt:

- Timer Interrupt: TIF (TIMERx_INTSTS[0]) bit will be set when timer counter value CNT (TIMERx_CNT[15:0]) matches the timer compared value CMPDAT (TIMERx_CMP[15:0]);

5.9.4.2 Timer Counting Mode

Timer controller provides three timer counting modes: one-shot, periodic and continuous counting operation modes.

Timer input clock or event source is divided by (PSC+1) before it is fed to the 16 bit up counter. By default the 8-bit prescaler PSC (TIMERx_CTL[7:0]) value is 0.

5.9.4.3 One–shot Mode

Writing 0b00 into TIMERx_CTL[28:27] selects one-shot mode for that timer.

CNTEN (TIMERx_CTL[30]) is the timer enable bit. Once enabled, the timer counter starts up counting. Once the CNT (TIMERx_CNT[15:0]) value reaches CMPDAT (TIMERx_CMP[15:0]), TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value and CNTEN bit is automatically cleared by hardware and timer counting operation stops. A timer interrupt will be triggered if INTEN (TIMERx_CTL[29]) is enabled.

5.9.4.4 Periodic Mode

Writing 0b01 into TIMERx_CTL[28:27] selects periodic mode for that timer.

CNTEN (TIMERx_CTL[30]) is the timer enable bit. Once enabled, the timer counter starts to count. Once the CNT (TIMERx_CNT[15:0]) value reaches CMPDAT (TIMERx_CMP[15:0]), TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value will be automatically cleared by hardware and timer counter start to count from 0 again. In the meantime, if INTEN (TIMERx_CTL[29]) is enabled a timer interrupt will be generated.

In this mode, timer controller keeps counting and comparing with CMPDAT value continuously, raises TIF periodically until the CNTEN bit is cleared.

5.9.4.5 Continuous Counting Mode

Writing 0b11 into TIMERx_CTL[28:27] selects continuous counting mode.

Under continuous counting mode, timer starts up counting once CNTEN (TIMERx_CTL[30]) is enabled.

When CNT (TIMERx_CNT[15:0]) value reaches CMPDAT (TIMERx_CMP[15:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1, hence a timer interrupt generated if enabled, and counting continues. Software can change CMPDAT to a different value immediately without stopping/resetting timer.

For example, if CMPDAT value is initially set as 80, when CNT reaches 80, timer raises TIF and generates interrupt (if interrupt enabled), and counting continues. Now assume software clears TIF and re-set CMPDAT value to 200, then when CNT reaches 200 timer will raise TIF and generate interrupt again.

In this mode, timer counter counts endlessly from 0 to 216 -1, then from 0 again, until CNTEN is disabled.

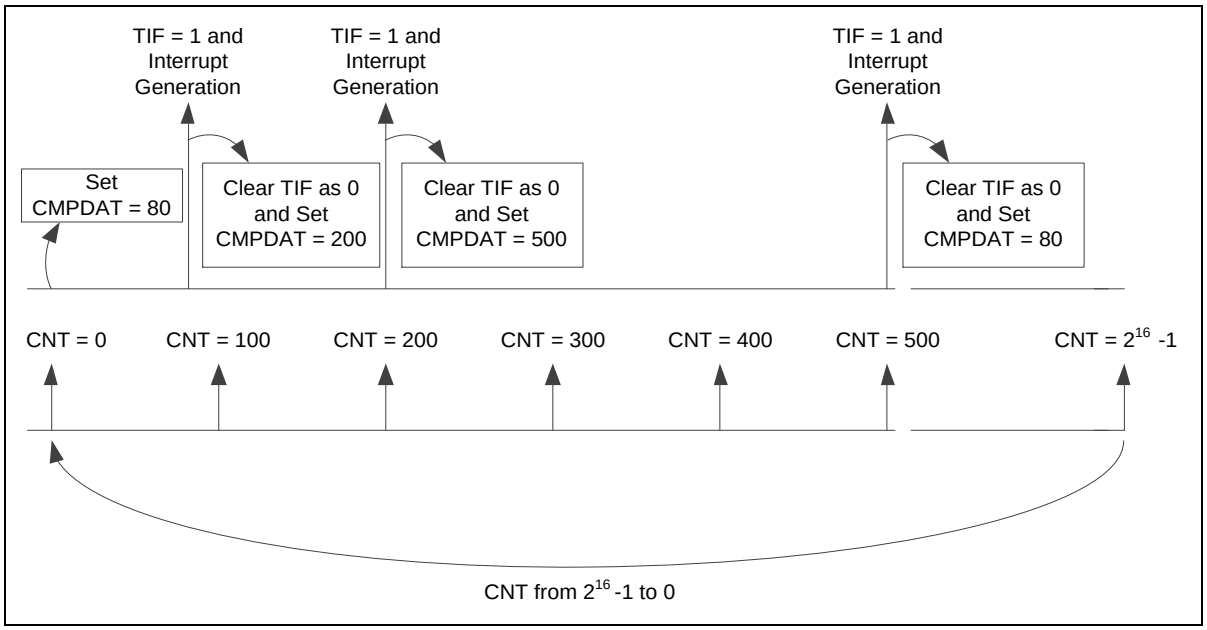


Figure 5.9-3 Continuous Counting Mode

5.9.5 Timer Controller Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
TMR Base Address: **TMR_BA=0x4001_0000**				
TIMER0_CTL	TMR_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TIMER0_CMP	TMR_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TIMER0_INTSTS	TMR_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER0_CNT	TMR_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TIMER1_CTL	TMR_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TIMER1_CMP	TMR_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000
TIMER1_INTSTS	TMR_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER1_CNT	TMR_BA+0x2C	R	Timer1 Data Register	0x0000_0000
IR_CTL	TMR_BA+0x34	R/W	IR Carrier Output Control Register	0x0000_0000
TIMER2_CTL	TMR_BA+0x40	R/W	Timer2 Control and Status Register	0x0000_0005
TIMER2_CMP	TMR_BA+0x44	R/W	Timer2 Compare Register	0x0000_0000
TIMER2_INTSTS	TMR_BA+0x48	R/W	Timer2 Interrupt Status Register	0x0000_0000
TIMER2_CNT	TMR_BA+0x4C	R	Timer2 Data Register	0x0000_0000

5.9.6 Timer Controller Register Description

Timer Control Register (TIMERx_CTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_CTL	TMR_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TIMER1_CTL	TMR_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TIMER2_CTL	TMR_BA+0x40	R/W	Timer2 Control and Status Register	0x0000_0005

31	30	29	28	27	26	25	24
Reserved	**CNTEN**	**INTEN**	**OPMODE**		**RSTCNT**	**ACTSTS**	**Reserved**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC							

Bits	Description	
[31]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[30]	**CNTEN**	**Counter Enable Bit** 0 = Stop/Suspend counting. 1 = Start counting. **Note:** This bit is auto-cleared by hardware in one-shot mode (OPMODE = 00b).
[29]	**INTEN**	**Interrupt Enable Bit** 0 = Disable TIMER Interrupt. 1 = Enable TIMER Interrupt. If timer interrupt is enabled, and time-out flag (TIF) is 1’b .The timer asserts its interrupt signal to CPU.

[28:27]	**OPMODE**	**Timer Operating Mode** 00 = The Timer is operating in the one-shot mode. The associated interrupt signal is generated once (if INTEN is 1) and CNTEN is automatically cleared by hardware. 01 = The Timer is operating in the periodic mode. The associated interrupt signal is generated periodically (if INTEN is 1). 10 = Reserved. 11 = The Timer is operating in continuous counting mode. The associated interrupt signal is generated when TIMERx_CNT = TIMERx_CMP (if INTEN is 1); however, the 16-bit up-counter counts continuously without reset. **Note:** When changing the Timer Operating Mode, the CNTEN bit should be set to 0 disable first.
[26]	**RSTCNT**	**Counter Reset Bit** Set this bit will reset the Timer counter, pre-scale and also force CNTEN to 0. 0 = No effect. 1 = Reset Timer's pre-scale counter, internal 16-bit up-counter and CNTEN bit.
[25]	**ACTSTS**	**Timer Active Status Bit (Read Only)** This bit indicates the counter status of Timer. 0 = Timer is not active. 1 = Timer is active.
[24:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**PSC**	**Timer Clock Prescaler** Clock input is divided by (PSC+1) before it is fed to the counter. If PSC = 0, then there is no scaling. **Note:** No matter CNTEN is 0 or 1, whenever software writes a new value into this register, TIMER will restart counting by using this new value and abort previous count.

Timer Compare Register (TIMERx_CMP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CMP	TMR_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TIMER1_CMP	TMR_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000
TIMER2_CMP	TMR_BA+0x44	R/W	Timer2 Compare Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**CMPDAT**	**Timer Comparison Value** CMPDAT is a 16-bit comparison register. When the 16-bit up-counter is enabled and its value is equal to CMPDAT value, a Timer out flag (TIF) is requested. **Note 1:** Never set CMPDAT to 0x000 or 0x001. Timer will not function correctly. **Note 2:** No matter CNTEN is 0 or 1, whenever software writes a new value into this register, TIMER will restart counting by using this new value and abort previous count.

Timer Interrupt Status Register (TIMERx_INTSTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_INTSTS	TMR_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER1_INTSTS	TMR_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER2_INTSTS	TMR_BA+0x48	R/W	Timer2 Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**TIF**	**Timer Interrupt Flag (Read Only)** This bit indicates the interrupt status of Timer. TIF bit is set by hardware when the 16-bit counter matches the timer comparison value (CMPDAT). It is cleared by writing 1 to itself 0 = No effect. 1 = CNT (TIMERx_CNT [15:0]) value matches the CMPDAT (TIMERx_CMP[15:0]) value.

Timer Data Register (TIMERx_CNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_CNT	TMR_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TIMER1_CNT	TMR_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TIMER2_CNT	TMR_BA+0x4C	R	Timer2 Data Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**CNT**	**Timer Data Register** User can read this register for the current up-counter value while TIMERx_CTL.CNTEN is set to 1,

IR Carrier Output Control Register (IR_CTL)

Register	Offset	R/W	Description	Reset Value
IR_CTL	TMR_BA+0x34	R/W	IR Carrier Output Control Register	0x0000_0000

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

Timer1 time-out signal is used to toggle IROUT output. Before IR carrier output is enabled, user needs to setup Timer1 according to output frequency of IR carrier.

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**IRCEN**	**IR carrier output enable** 0 = Disable IR carrier output, 1 = Enable IR carrier output. Timer1 time out will toggle the output state on IROUT pin.
[0]	**NONCS**	**Non-carrier state** 0 = IROUT keeps low when IRCEN is 0, 1 = IROUT keeps high when IRCEN is 0.

5.10 Watchdog Timer

5.10.1 Overview

The purpose of Watchdog Timer (WDT) is to perform a system reset if system runs into an unknown state. This prevents system from hanging for an indefinite period of time.

5.10.2 Features

- 19-bit free running up counter for WDT time-out interval
- Selectable time-out interval (2^4 ~ 2^{18}) WDT_CLK
- Supports to force WDT enabled after chip power on or reset by setting CWDTEN in Config0 register.
- Supports WDT time-out wake-up function under STOP mode only if WDT clock source is selected as 10 KHz

5.10.3 Block diagram

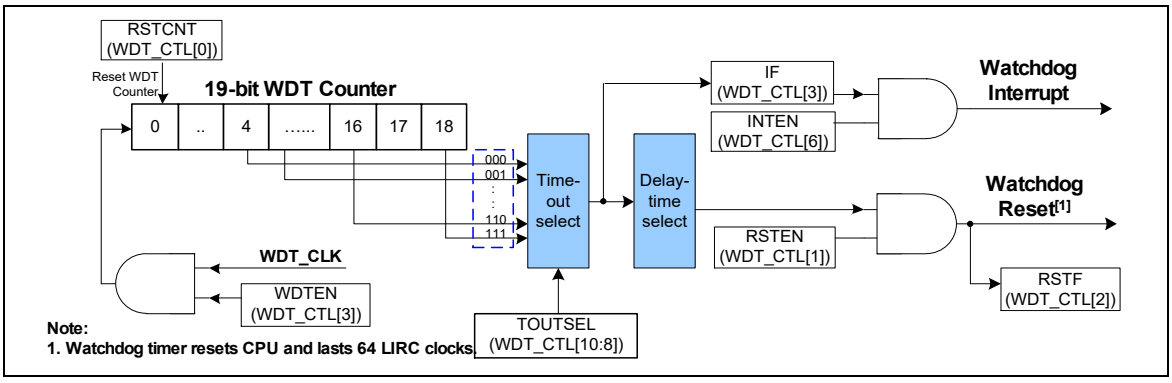


Figure 5.10-1 Watchdog Timer Block Diagram

5.10.4 Clock Control

The WDT clock control is shown as following.

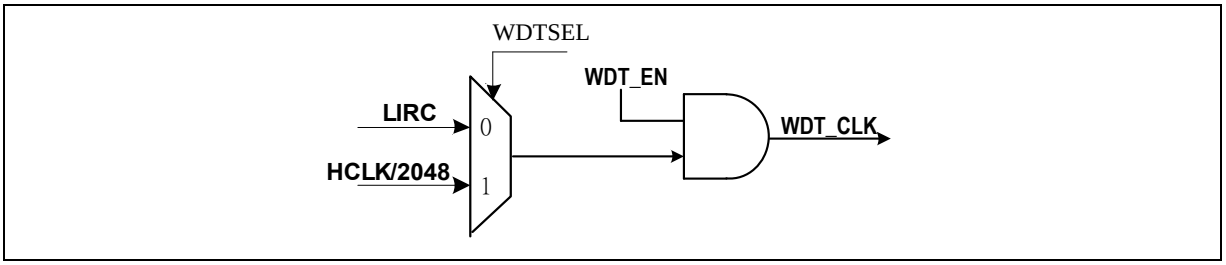


Figure 5.10-2 WDT Clock Control Diagram

5.10.5 Functional Description

The watchdog timer includes a 19-bit free running counter with programmable time-out intervals. Setting WDTEN (WDT_CTL [7]) enables the watchdog timer and the WDT counter starts counting up. When the counter reaches the selected time-out interval, Watchdog timer interrupt flag IF (WDT_CTL [3]) will be set immediately to request a WDT interrupt if the watchdog timer interrupt enable bit INTEN(WDT_CTL [6]) is set, in the meantime, a specified delay time follows the time-out event. User must set RSTCNT (WDT_CTL [0]) (Watchdog timer reset) high to reset the 19-bit WDT counter to prevent Watchdog timer reset before the delay time expires. RSTCNT (WDT_CTL [0]) bit is auto cleared by hardware after WDT counter is reset. There are eight time-out intervals with specific delay time which are selected by Watchdog timer interval select bits TOUTSEL (WDT_CTL [10:8]). If the WDT counter has not been cleared after the specific delay time expires, the watchdog timer will set Watchdog Timer Reset Flag RSTF (WDT_CTL [2]) high and reset CPU. This reset will last 64 LIRC clocks then CPU restarts executing program from reset vector (0x0000 0000). RSTF (WDT_CTL [2]) will not be cleared by Watchdog reset. User may poll RSTF (WDT_CTL [2]) by software to recognize the reset source.

Table 5.10-1 Watchdog Timeout Interval Selection

TOUTSEL	WDT Interrupt Timeout	WDT Reset Timeout	WDT Reset Timeout Interval (WDT_CLK=10 KHz(LIRC))
000	2^4 WDT_CLK	(2^4 + 1024) WDT_CLK	104 ms
001	2^6 WDT_CLK	(2^6 + 1024) WDT_CLK	108.8 ms
010	2^8 WDT_CLK	(2^8 + 1024) WDT_CLK	128 ms
011	2^{10} WDT_CLK	(2^{10} + 1024) WDT_CLK	204.8 ms
100	2^{12} WDT_CLK	(2^{12} + 1024) WDT_CLK	512 ms
101	2^{14} WDT_CLK	(2^{14} + 1024) WDT_CLK	1740.8 ms
110	2^{16} WDT_CLK	(2^{16} + 1024) WDT_CLK	6656 ms
111	2^{18} WDT_CLK	(2^{18} + 1024) WDT_CLK	26316.8 ms

5.10.6 Watchdog Timer Control Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
WDT Base Address: **WDT_BA = 0x4000_4000**				
WDT_CTL	WDT_BA+0x00	R/W	Watchdog Timer Control Register	0x0000_0700

5.10.7 Watchdog Timer Control Register Description

Watchdog Timer Control Register (WDT_CTL)

This is a protected register, to write to register, first issue the unlock sequence (see Register Lock Control Register (SYS_REGLCTL)). Only flag bits IF and RSTF are unprotected and can be write-cleared at any time.

Register	Offset	R/W	Description	Reset Value
WDT_CTL	WDT_BA+0x00	R/W	Watchdog Timer Control Register	0x0000_0700

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**TOUTSEL**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WDTEN	**INTEN**	**Reserved**		**IF**	**RSTF**	**RSTEN**	**RSTCNT**

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[10:8]	**TOUTSEL**	**Watchdog Timer Interval Select** These three bits select the timeout interval for the Watchdog timer, a watchdog reset will occur 1024 clock cycles later if Watchdog timer is not reset. The WDT interrupt timeout is given by: 000 = 2^4 * WDT_CLK. 001 = 2^6 * WDT_CLK. 010 = 2^8 * WDT_CLK. 011 = 2^{10} * WDT_CLK. 100 = 2^{12} * WDT_CLK. 101 = 2^{14} * WDT_CLK. 110 = 2^{16} * WDT_CLK. 111 = 2^{18} * WDT_CLK. WDT reset timeout = (WDT interrupt timeout +1024) * WDT_CLK. Where WDT_CLK is the period of the Watchdog Timer clock source.
[7]	**WDTEN**	**Watchdog Timer Enable** 0 = Disable the WDT(Watchdog timer) (This action will reset the internal counter). 1 = Enable the WDT(Watchdog timer).
[6]	**INTEN**	**Watchdog Time-Out Interrupt Enable** 0 = Disable the WDT time-out interrupt. 1 = Enable the WDT time-out interrupt.
[5:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3]	**IF**	**Watchdog Timer Interrupt Flag** If the Watchdog timer interrupt is enabled, then the hardware will set this bit to indicate that the Watchdog timer interrupt has occurred. If the Watchdog timer interrupt is not enabled, then this bit indicates that a timeout period has elapsed. 0 = Watchdog timer interrupt has not occurred. 1 = Watchdog timer interrupt has occurred. **Note:** This bit is cleared by writing 1 to this bit.
[2]	**RSTF**	**Watchdog Timer Reset Flag** When the Watchdog timer initiates a reset, the hardware will set this bit. This flag can be read by software to determine the source of reset. Software is responsible to clear it manually by writing 1 to it. If RSTEN is disabled, then the Watchdog timer has no effect on this bit. 0 = Watchdog timer reset has not occurred. 1= Watchdog timer reset has occurred. **Note:** This bit is cleared by writing 1 to this bit.

[1]	**RSTEN**	**Watchdog Timer Reset Enable(Write Protected)** Setting this bit will enable the Watchdog timer reset function. 0 = Disable Watchdog timer reset function. 1= Enable Watchdog timer reset function. **Note:** This bit is writing protected. Refer to the SYS_REGLCTL.
[0]	**RSTCNT**	**Clear Watchdog Timer (Write Protected)** Set this bit will clear the Watchdog timer. 0 = Writing 0 to this bit has no effect. 1 = Reset the contents of the Watchdog timer. **Note1:** This bit will be automatically cleared by hardware. **Note2:** This bit is writing protected. Refer to the SYS_REGLCTL.

5.11 I2C Serial Interface Controller (Master/Slave)

5.11.1 Overview

ISD91500 supports two I2C controller – I2C0,I2C1. I2C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The I2C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously. Serial, 8-bit oriented, bi-directional data transfers can be made up 1.0 Mbps.

5.11.2 Features

- Up to two I2C Controller.
- Master/Slave mode
- Bidirectional data transfer between masters and slaves
- Multi-master bus (no central master).
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus.
- Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer.
- Built-in a 14-bit time-out counter will request the I2C interrupt if the I2C bus hangs up and timer-out counter overflows.
- External pull-up are needed for high output
- Programmable clock allowing versatile rate control.
- Supports 7-bit addressing mode
- Supports multiple address recognition (four slave address with mask option)

5.11.3 Functional Description

On I2C bus, data is transferred between a Master and a Slave synchronously to SCL on the SDA line on a byte-by-byte basis. Each data byte is 8 bits long. There is one SCL clock pulse for each data bit with the MSB being transmitted first. An acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to Figure 5.11-1 for more detail I2C BUS Timing.

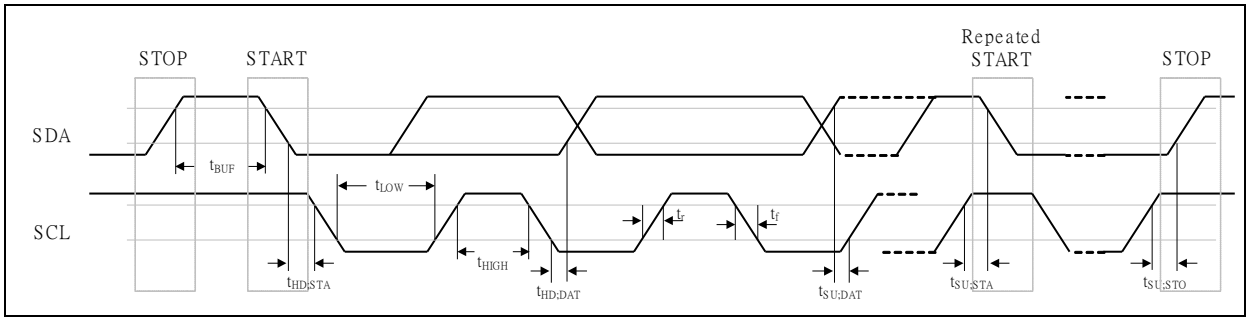


Figure 5.11-1 I2C Bus Timing

The device's on-chip I2C logic provides the serial interface that meets the I2C bus standard mode specification. The I2C port handles byte transfers autonomously. To enable this port, the bit I2CEN in I2C_CTL should be set to '1'. The I2C H/W interfaces to the I2C bus via two pins: I2C_SDA and I2C_SCL.See chapter0 for alternate GPIO pin functions. Pull up resistor is needed for these pins for I2C operation as these are open drain pins.

5.11.3.1 I²C Protocol

Normally, a standard communication consists of four parts:

1) START or Repeated START signal generation
2) Slave address transfer
3) Data transfer
4) STOP signal generation

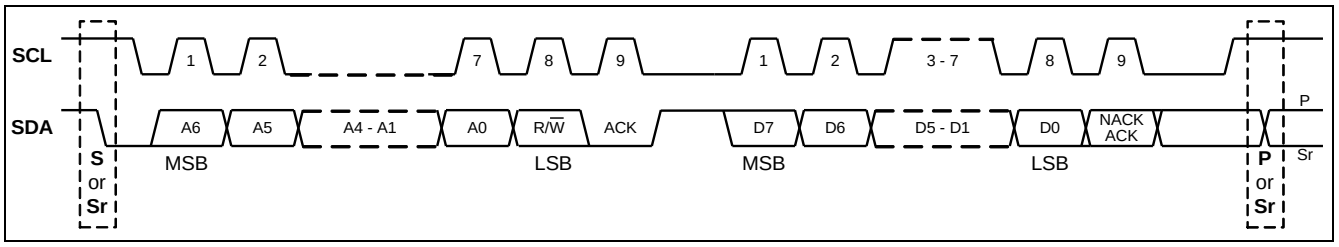


Figure 5.11-2 I2C Protocol

5.11.3.2 Data transfer on the I2C-bus

A master-transmitter always begins by addressing a slave receiver with a 7-bit address. For a transaction where the master-transmitter is sending data to the slave, the transfer direction is not changed, master is always transmitting and slave acknowledges the data, see Figure 5.11-3 .

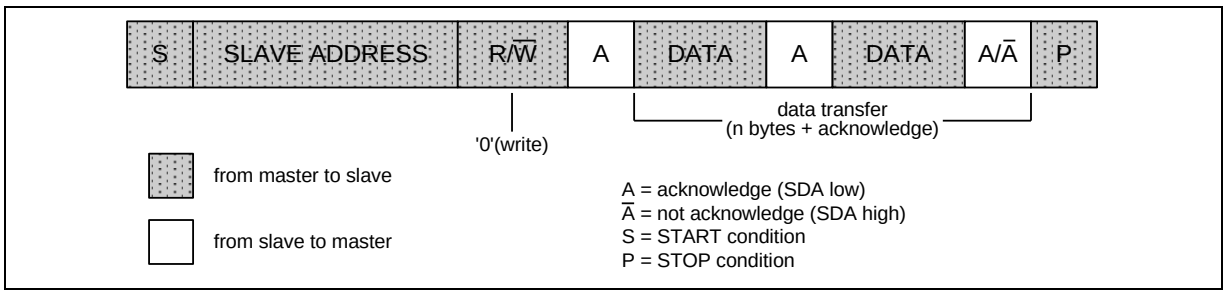


Figure 5.11-3 Master Transmits Data to Slave

For a master to read data from a slave, master addresses slave with the R/W bit set to ‘1’, immediately after the first byte (address) is acknowledged by the slave the transfer direction is changed and slave sends data to the master and master acknowledges the data transfer.

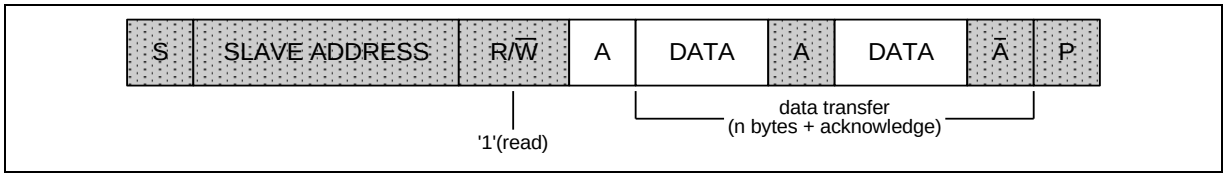


Figure 5.11-4 Master Reads Data from Slave

5.11.3.3 START or Repeated START signal

When the bus is free/idle, meaning no master device is engaging the bus (both SCL and SDA lines are high), a master can initiate a transfer by sending a START signal. A START signal, usually referred to as the S-bit, is defined as a HIGH to LOW transition on the SDA line while SCL is HIGH. The START signal denotes the beginning of a new data transfer.

A Repeated START (Sr) is a START signal without first generating a STOP signal. The master uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

STOP signal

The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the P-bit, is defined as a LOW to HIGH transition on the SDA line while SCL is HIGH.

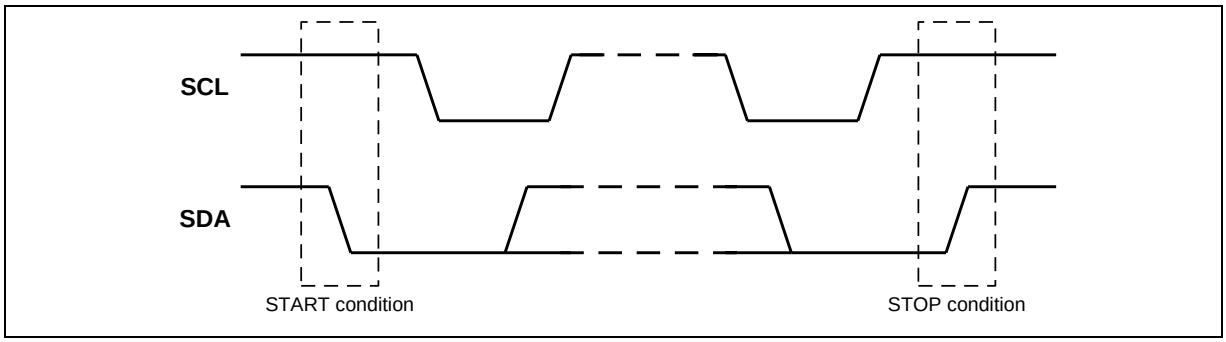


Figure 5.11-5 START and STOP condition

5.11.3.4 Slave Address Transfer

The first byte of data transferred by the master immediately after the START signal is the slave address. This is a 7-bits calling address followed by a RW bit. The RW bit signals the slave the data transfer direction. No two slaves in the system can have the same address. Only the slave with an address that matches the one transmitted by the master will respond by returning an acknowledge bit by pulling the SDA low at the 9th SCL clock cycle.

5.11.3.5 Data Transfer

Once successful slave addressing has been achieved, the data transfer can proceed on a byte-by-byte basis in the direction specified by the RW bit sent by the master. Each transferred byte is followed by an acknowledge bit on the 9th SCL clock cycle. If the slave signals a Not Acknowledge (NACK), the master can generate a STOP signal to abort the data transfer or generate a Repeated START signal and start a new transfer cycle.

If the master, as the receiving device, does Not Acknowledge (NACK) the slave, the slave releases the SDA line for the master to generate a STOP or Repeated START signal.

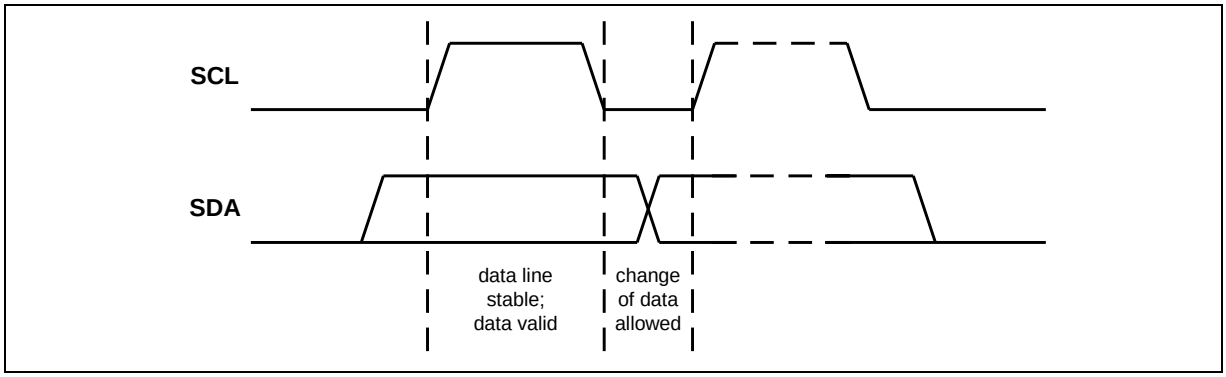


Figure 5.11-6 Bit Transfer on the I2C bus

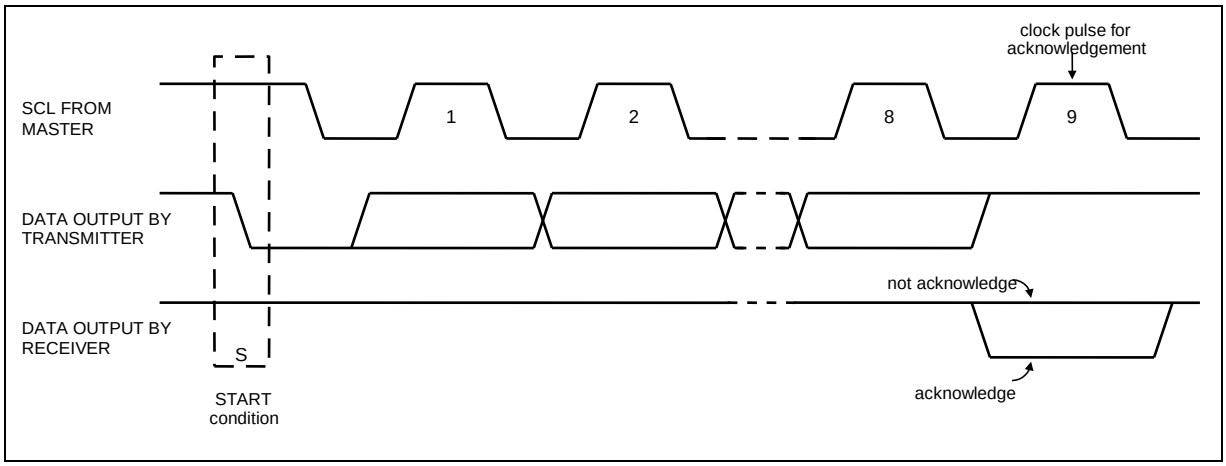


Figure 5.11-7 Acknowledge on the I2C bus

5.11.4 Modes of Operation

The on-chip I2C ports support five operation modes, Master transmitter, Master receiver, Slave transmitter, Slave receiver, and GC call.

In a given application, I2C port may operate as a master or as a slave. In the slave mode, the I2C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master(by setting the AA bit), an acknowledge pulse will be transmitted out on the 9th clock. An interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, the hardware waits until the bus is free before the master mode is entered so that a possible slave action is not interrupted. If bus arbitration is lost in the master mode, I2C port switches to the slave mode immediately and can detect its own slave address in the same serial transfer.

5.11.4.1 Master Transmitter Mode

Serial data output through SDA while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the data direction bit. In this case the data direction bit (R/W) will be logic 0, and it is represented by “W” in the flow diagrams. Thus the first byte transmitted is SLA+W. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an acknowledge bit is received. START and STOP conditions are output to indicate the beginning and the end of a serial transfer.

5.11.4.2 Master Receiver Mode

In this case the data direction bit (R/W) will be logic 1, and it is represented by “R” in the flow diagrams. Thus the first byte transmitted is SLA+R. Serial data is received via SDA while SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are output to indicate the beginning and end of a serial transfer.

5.11.4.3 Slave Receiver Mode

Serial data and the serial clock are received through SDA and SCL. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and direction bit.

5.11.4.4 Slave Transmitter Mode

The first byte is received and handled as in the slave receiver mode. However, in this mode, the direction bit will indicate that the transfer direction is reversed. Serial data is transmitted via SDA while the serial clock is input through SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer.

5.11.5 Data Transfer Flow in Five Operating Modes

The five operating modes are: Master/Transmitter, Master/Receiver, Slave/Transmitter, Slave/Receiver and GC Call. Bits STA, STO and AA in I2C_CTL register will determine the next state of the SIO hardware after SI flag is cleared. Upon completion of the new action, a new status code will be updated and the SI flag will be set. If the I2C interrupt control bit INTEN (I2C_CTL[7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

Data transfers in each mode are shown in the following figures.

*** Legend for the following five figures:

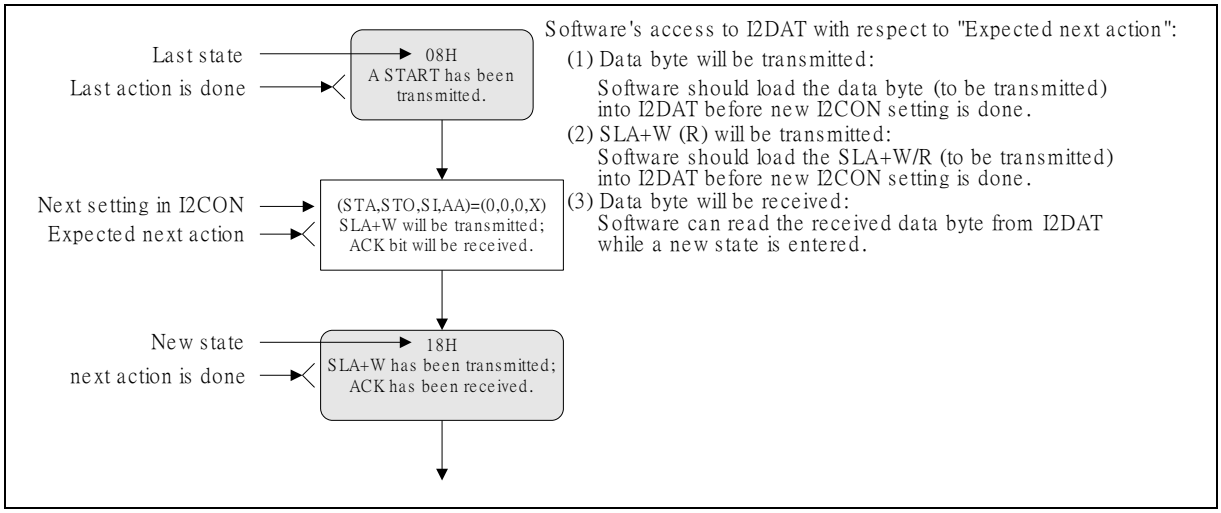


Figure 5.11-8 Legend for the following four figures

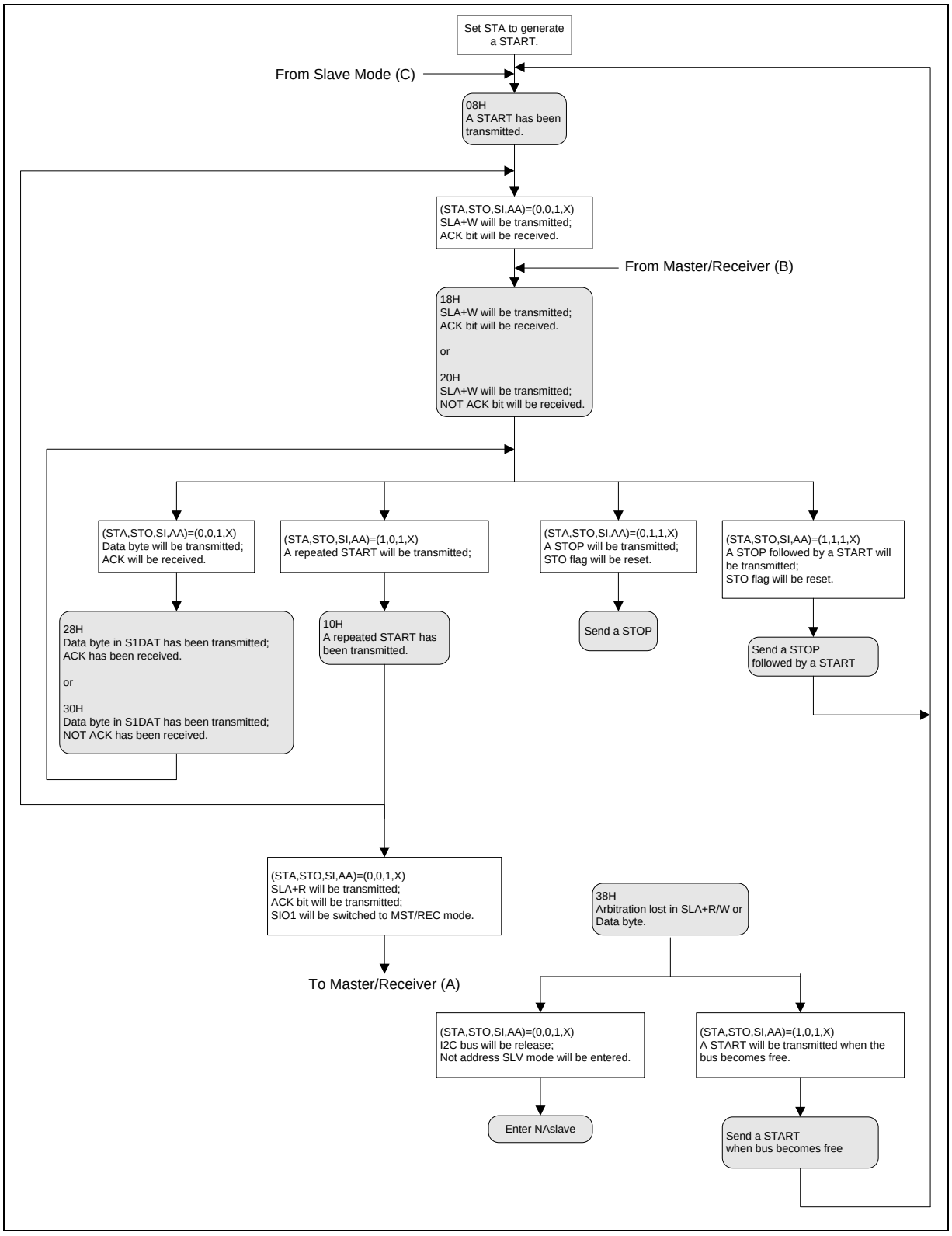


Figure 5.11-9 Master Transmitter Mode

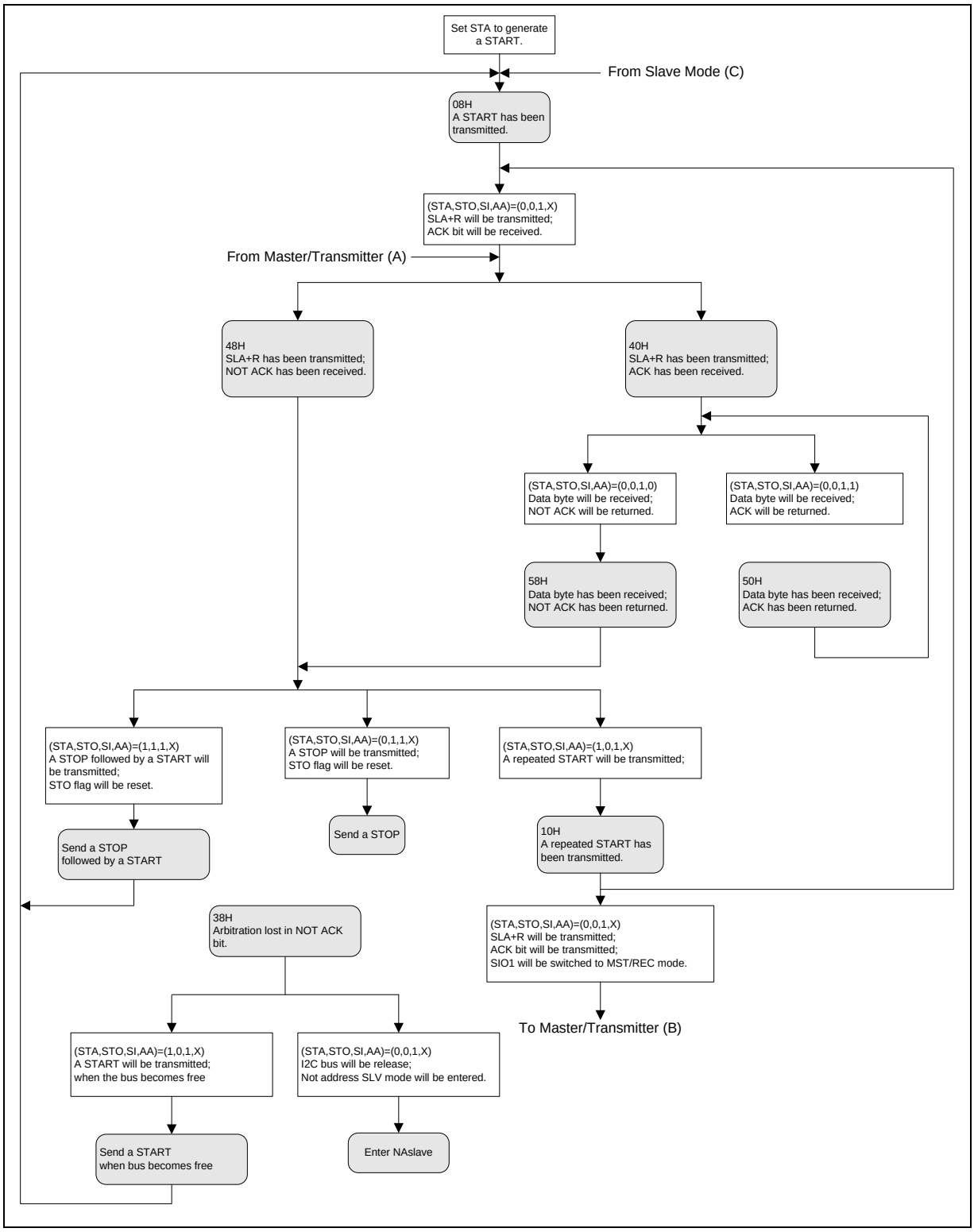


Figure 5.11-10 Master Receiver Mode

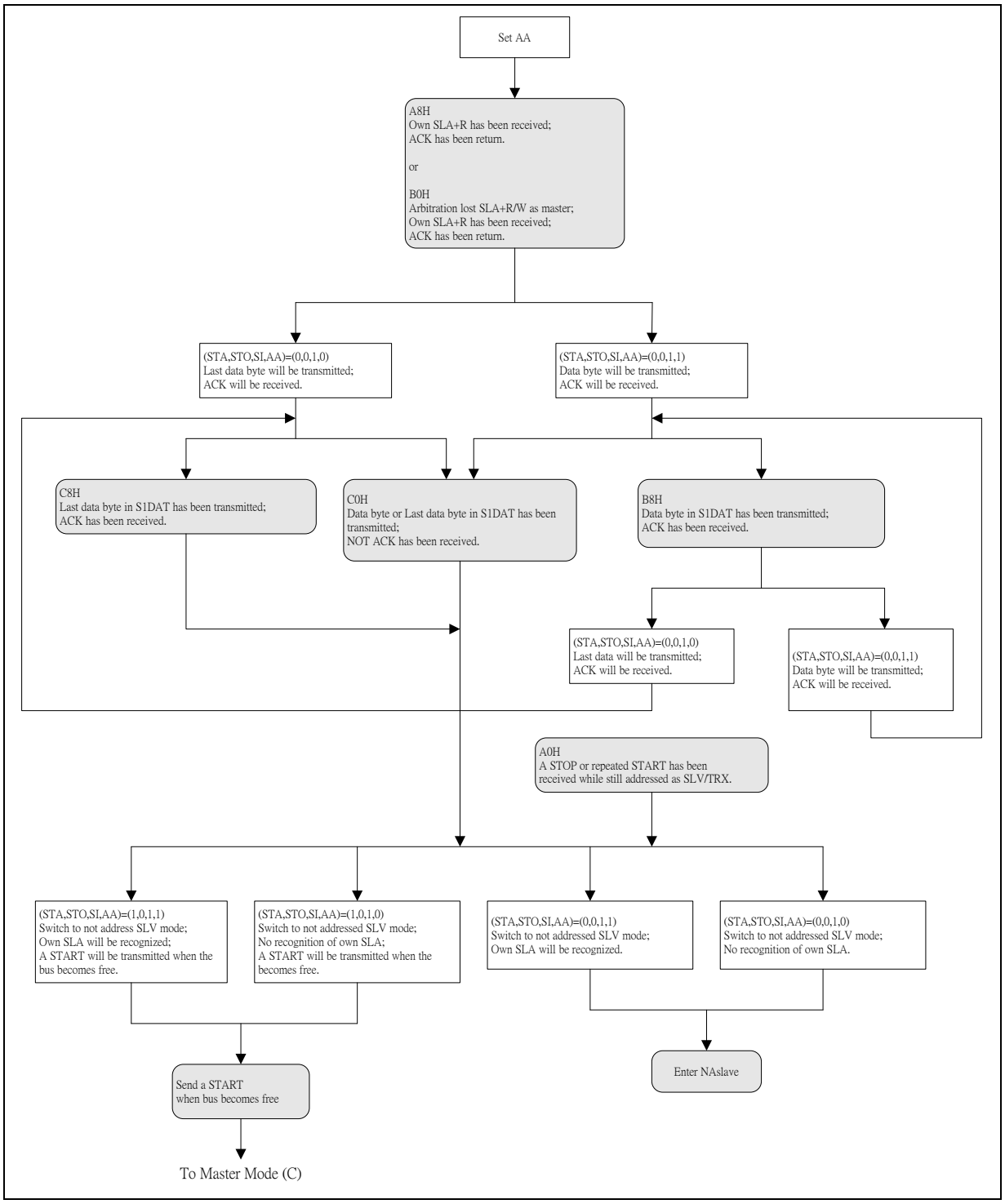


Figure 5.11-11 Slave Transmitter Mode

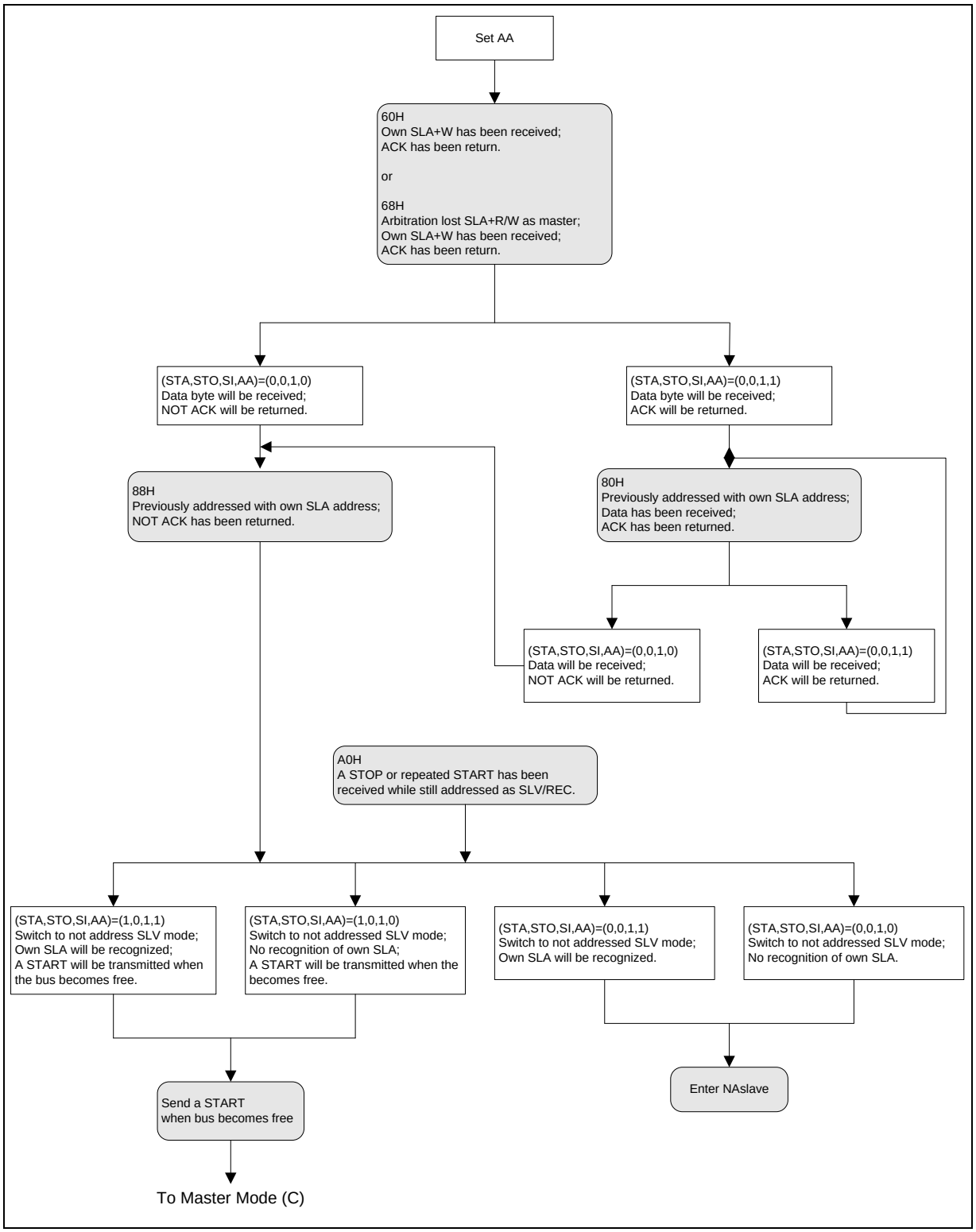


Figure 5.11-12 Slave Receiver Mode

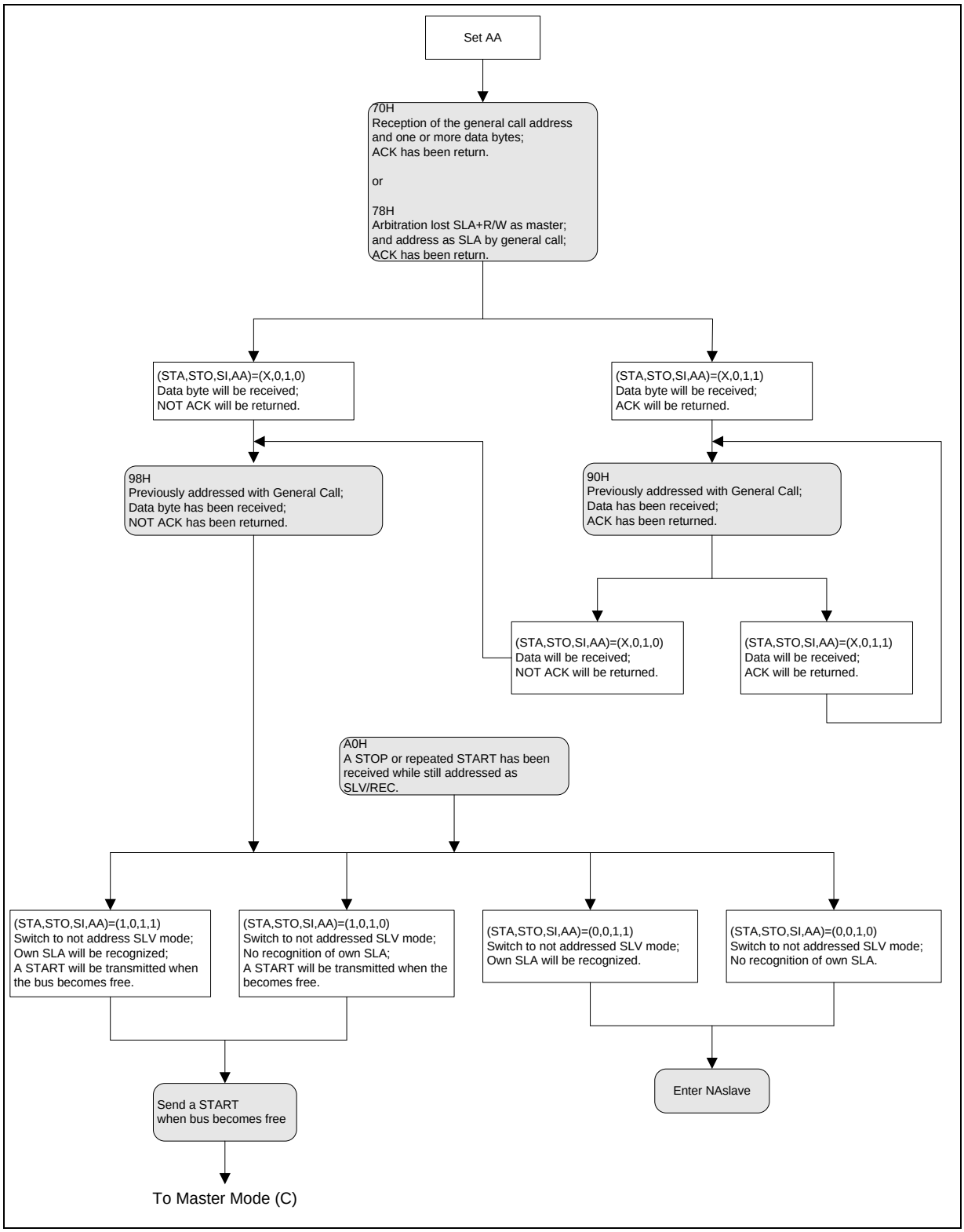


Figure 5.11-13 GC Mode

5.11.6 I2C Protocol Registers

The CPU interfaces to the SIO port through the following thirteen special function registers: I2C_CTL (control register), I2C_STATUS (status register), I2C_DAT (data register), I2C_ADDRn (address registers, n=0~3), I2C_ADDRMSKn (address mask registers, n=0~3), I2C_CLKDIV (clock rate register) and I2C_TOCTL (Time-out counter register). Bits 31~ bit 8 of these I2C special function registers are reserved. These bits do not have any functions and are all zero if read back.

When I2C port is enabled by setting I2CEN (I2C_CTL[6]) to high, the internal states will be controlled by I2C_CTL and I2C logic hardware. Once a new status code is generated and stored in I2C_STATUS, the I2C Interrupt Flag bit SI (I2C_CTL[3]) will be set automatically. If the Enable Interrupt bit EI (I2C_CTL[7]) is set high at this time, the I2C interrupt will be generated. The bit field I2C_STATUS[7:3] stores the internal state code, the lowest 3 bits of I2C_STATUS are always zero and the contents are stable until SI is cleared by software. The base address of the I2C peripheral on the ISD91500 is 0x4002_0000.

5.11.6.1 Address Registers (I2C_ADDR)

I2C port is equipped with four slave address registers I2C_ADDRn (n=0~3). The contents of the register are irrelevant when I2C is in master mode. In the slave mode, the bit field I2C_ADDRn[7:1] must be loaded with the MCU's own slave address. The I2C hardware will react if the contents of I2C_ADDR are matched with the received slave address.

The I2C ports support the "General Call" function. If the GC bit (I2C_ADDRn[0]) is set the I2C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.

When GC bit is set, the I2C is in Slave mode, it can be received the general call address by 00H after Master send general call address to I2C bus, then it will follow status of GC mode. If it is in master mode, the AA bit (I2C_CTL[2], Assert Acknowledge control bit) must be cleared when it will send general call address of 00H to I2C bus.

I2C-bus controllers support multiple address recognition with four address mask registers I2ADRMn (n=0~3). When the bit in the address mask register is set to one, it means the received corresponding address bit is don't-care. If the bit is set to zero, that means the received corresponding register bit should be exact the same as address register.

5.11.6.2 Data Register (I2C_DAT)

This register contains a byte of serial data to be transmitted or a byte which has just been received. The CPU can read from or write to this 8-bit (I2C_DAT[7:0]) directly addressable SFR while it is not in the process of shifting a byte. This occurs when SIO is in a defined state and the serial interrupt flag (SI) is set. Data in I2C_DAT[7:0] remains stable as long as SI bit is set. While data is being shifted out, data on the bus is simultaneously being shifted in; I2C_DAT[7:0] always contains the last data byte present on the bus. Thus, in the event of arbitration lost, the transition from master transmitter to slave receiver is made with the correct data in I2C_DAT[7:0].

I2C_DAT[7:0] and the acknowledge bit form a 9-bit shift register, the acknowledge bit is controlled by the SIO hardware and cannot be accessed by the CPU. Serial data is shifted through the acknowledge bit into I2C_DAT[7:0] on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into I2C_DAT[7:0], the serial data is available in I2C_DAT[7:0], and the acknowledge bit (ACK or NACK) is returned by the control logic during the ninth clock pulse. Serial data is shifted out from I2C_DAT[7:0] on the falling edges of SCL clock pulses, and is shifted into I2C_DAT[7:0] on the rising edges of SCL clock pulses.

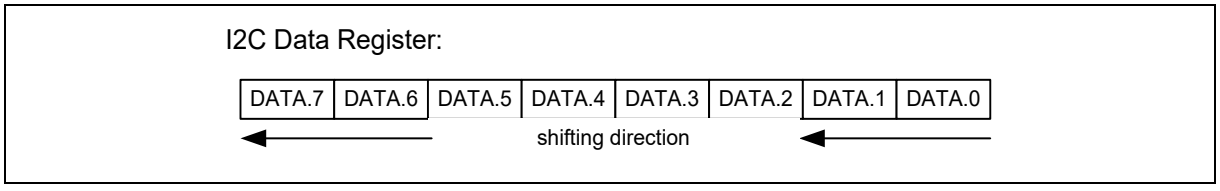


Figure 5.11-14 I2C Data Shift Direction

5.11.6.3 Control Register (I2C_CTL)

The CPU can read from and write to this 8-bit field of I2C_CTL[7:0]. Two bits are affected by hardware: the SI bit is set when the I2C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when I2CEN = "0".

INTEN Enable Interrupt.

I2CEN Set to enable I2C serial function block. When I2CEN=1 the I2C serial function is enabled.

STA I2C START Control Bit. Setting STA to logic 1 enters master mode, the I2C hardware sends a START or repeat START condition to bus when the bus is free.

STO I2C STOP Control Bit. In master mode, setting STO transmits a STOP condition to the bus. The I2C hardware will check the bus condition and if a STOP condition is detected this flag will be cleared by hardware. In a slave mode, setting STO resets I2C hardware to the defined “not addressed” slave mode. This means it is NO LONGER in the slave receiver mode to receive data from the master transmit device.

SI I2C Interrupt Flag. When a new SIO state is present in the I2C_STATUS register, the SI flag is set by hardware, and if bit INTEN (I2C_CTL[7]) is set, the I2C interrupt is requested. SI must be cleared by software. Clear SI is by writing one to this bit.

AA Assert Acknowledge Control Bit. When AA=1 prior to address or data received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when:

1.) A slave is acknowledging the address sent from master,
2.) A receiver device is acknowledging the data sent by a transmitter.

When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.

5.11.6.4 Status Register (I2C_STATUS)

I2C_STATUS[7:0] is an 8-bit read-only register. The three least significant bits are always 0. The bit field I2C_STATUS[7:3] contains the status code. There are 26 possible status codes. When I2C_STATUS[7:0] contains F8H, no serial interrupt is requested. All other I2C_STATUS[7:3] values correspond to defined SIO states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2C_STATUS[7:3] one machine cycle after SI is set by hardware and is still present one machine cycle after SI has been reset by software.

In addition, state 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the format frame. Examples of illegal positions are during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I2C from bus error, STO should be set and SI should be clear to enter not addressed slave mode. Then clear STO to release bus and to wait new communication. I2C bus cannot recognize stop condition during this action when bus error occurs.

5.11.6.5 I2C Clock Baud Rate Bits (I2C_CLKDIV)

The data baud rate of I2C is determined by I2C_CLKDIV[7:0] register when SIO is in a master mode. It is not important when SIO is in a slave mode. In the slave modes, SIO will automatically synchronize with any clock frequency up to 1M Hz from master I2C device.

Data Baud Rate of I2C = PCLK /(4x(I2C_CLKDIV[7:0]+1)). If PCLK=16MHz, the I2C_CLKDIV[7:0] = 40 (28H), data baud rate of I2C = 16MHz/(4x(40 +1)) = 97.5Kbits/sec.

5.11.6.6 The I2C Time-out Counter Register (I2C_TOCTL)

There is a 14-bit time-out counter which can be configured to deal with an I2C bus hang-up. If the time-out counter is enabled, the counter starts up-counting until it overflows (TOIF=1) and generates I2C interrupt to CPU or stops counting by clearing TOCEN to 0. When time-out counter is enabled, setting flag SI to high will reset counter. Counter will re-start after SI is cleared. If the I2C bus hangs up, counter will overflow and generate a CPU interrupt. Refer to Figure 5.11-15 for the 14-bit time-out counter. User can clear TOIF by writing one to this bit.

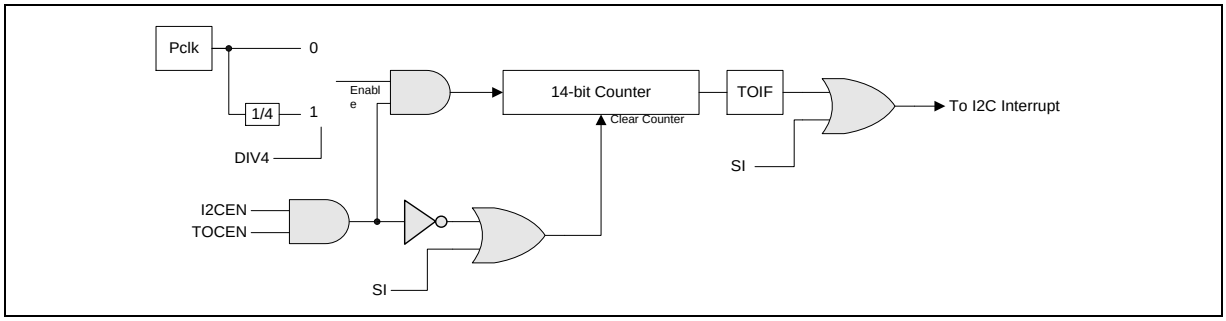


Figure 5.11-15 I2C Time-out Count Block Diagram

5.11.7 I2C Control Register Mapping

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
I2C Base Address: **I2Cx_BA = 0x4002_0000 + (0x1000 * x)** **x=0,1**				
I2C_CTL	I2Cx_BA+0x00	R/W	I2C Control Register	0x0000_0000
I2C_ADDR0	I2Cx_BA+0x04	R/W	I2C Slave address Register0	0x0000_0000
I2C_DAT	I2Cx_BA+0x08	R/W	I2C DATA Register	0x0000_0000
I2C_STATUS	I2Cx_BA+0x0C	R	I2C Status Register	0x0000_00F8
I2C_CLKDIV	I2Cx_BA+0x10	R/W	I2C clock divided Register	0x0000_0000
I2C_TOCTL	I2Cx_BA+0x14	R/W	I2C Time out control Register	0x0000_0000
I2C_ADDR1	I2Cx_BA+0x18	R/W	I2C Slave address Register1	0x0000_0000
I2C_ADDR2	I2Cx_BA+0x1C	R/W	I2C Slave address Register2	0x0000_0000
I2C_ADDR3	I2Cx_BA+0x20	R/W	I2C Slave address Register3	0x0000_0000
I2C_ADDRMSK0	I2Cx_BA+0x24	R/W	I2C Slave address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2Cx_BA+0x28	R/W	I2C Slave address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2Cx_BA+0x2C	R/W	I2C Slave address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2Cx_BA+0x30	R/W	I2C Slave address Mask Register3	0x0000_0000

5.11.8 I2C Control Register Description

I2C CONTROL REGISTER (I2C_CTL)

Register	Offset	R/W	Description	Reset Value
I2C_CTL	I2Cx_BA+0x00	R/W	I2C Control Register	0x0000_0000

7	6	5	4	3	2	1	0
INTEN	**I2CEN**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**INTEN**	**Enable Interrupt** 0 = Disable interrupt. 1 = Enable interrupt CPU.
[6]	**I2CEN**	**I2C Controller Enable Bit** 0 = Disable. 1 = Enable. Set to enable I2C serial function block.
[5]	**STA**	**I2C START Control Bit** Setting STA to logic 1 will enter master mode, the I2C hardware sends a START or repeat START condition to bus when the bus is free.
[4]	**STO**	**I2C STOP Control Bit** In master mode, set STO to transmit a STOP condition to bus. I2C hardware will check the bus condition, when a STOP condition is detected this bit will be cleared by hardware automatically. In slave mode, setting STO resets I2C hardware to the defined “not addressed” slave mode. This means it is NO LONGER in the slave receiver mode able receive data from the master transmit device.
[3]	**SI**	**I2C Interrupt Flag** When a new SIO state is present in the I2C_STATUS register, the SI flag is set by hardware, and if bit I2CEN (I2C_CTL[7]) is set, the I2C interrupt is requested. SI must be cleared by software. Clear SI is by writing one to this bit.

[2]	**AA**	**Assert Acknowledge Control Bit** When AA=1 prior to address or data received, an acknowledge (ACK - low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when:. 1. A slave is acknowledging the address sent from master, 2. The receiver devices are acknowledging the data sent by transmitter. When AA = 0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.
[1:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

I2C DATA REGISTER (I2C_DAT)

Register	Offset	R/W	Description	Reset Value
I2C_DAT	I2Cx_BA+0x08	R/W	I2C DATA Register	0x0000_0000

7	6	5	4	3	2	1	0
DAT[7:0]							

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**DAT**	**I2C Data Register** During master or slave transmit mode, data to be transmitted is written to this register. During master or slave receive mode, data that has been received may be read from this register.

I2C STATUS REGISTER (I2C_STATUS)

Register	Offset	R/W	Description	Reset Value
I2C_STATUS	I2Cx_BA+0x0C	R	I2C Status Register	0x0000_00F8

7	6	5	4	3	2	1	0
STATUS[7:0]							

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**STATUS**	**I2C Status Register** The status register of I2C: The three least significant bits are always 0. The five most significant bits contain the status code. There are 26 possible status codes. When I2C_STATUS contains F8H, no serial interrupt is requested. All other I2C_STATUS values correspond to defined I2C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2C_STATUS one PCLK cycle after SI is set by hardware and is still present one PCLK cycle after SI has been reset by software. In addition, states 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit.

I2C DATA BAUD RATE CONTROL REGISTER (I2C_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2C_CLKDIV	I2Cx_BA+0x10	R/W	I2C clock divided Register	0x0000_0000

7	6	5	4	3	2	1	0
DIVIDER[7:0]							

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**DIVIDER**	**I2C Clock Divided Register** The I2C clock rate bits: Data Baud Rate of I2C = PCLK /(4x(I2C_CLKDIV+1)). **Note:** I2C_CLKDIV must >2

I2C TIME-OUT COUNTER REGISTER (I2C_TOCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TOCTL	I2Cx_BA+0x14	R/W	I2C Time out control Register	0x0000_0000

7	6	5	4	3	2	1	0
Reserved					TOCEN	TOCDIV4	TOIF

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**TOCEN**	**Time-out Counter Control Bit** 0 = Disable. 1 = Enable. When enabled, the 14 bit time-out counter will start counting when SI is clear. Setting flag SI to high will reset counter and re-start up counting after SI is cleared.
[1]	**TOCDIV4**	**Time-out Counter Input Clock Divide by 4** 0 = Disable. 1 = Enable. When enabled, the time-out clock is PCLK/4.
[0]	**TOIF**	**Time-out Flag** 0 = No time-out. 1 = Time-out flag is set by H/W. It can interrupt CPU. Write 1 to clear..

I2C SLAVE ADDRESS REGISTER (I2CADDRn)

Register	Offset	R/W	Description	Reset Value
I2C_ADDR0	I2Cx_BA+0x04	R/W	I2C Slave address Register0	0x0000_0000
I2C_ADDR1	I2Cx_BA+0x18	R/W	I2C Slave address Register1	0x0000_0000
I2C_ADDR2	I2Cx_BA+0x1C	R/W	I2C Slave address Register2	0x0000_0000
I2C_ADDR3	I2Cx_BA+0x20	R/W	I2C Slave address Register3	0x0000_0000

7	6	5	4	3	2	1	0
ADDR[7:1]							**GC**

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:1]	**ADDR**	**I2C Address Register** The content of this register is irrelevant when I2C is in master mode. In the slave mode, the seven most significant bits must be loaded with the MCU's own address. The I2C hardware will react if any of the addresses are matched.
[0]	**GC**	**General Call Function** 0 = Disable General Call Function. 1 = Enable General Call Function.

I2C SLAVE ADDRESS MASK REGISTER (I2CADMn)

Register	Offset	R/W	Description	Reset Value
I2C_ADDRMSK0	I2Cx_BA+0x24	R/W	I2C Slave address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2Cx_BA+0x28	R/W	I2C Slave address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2Cx_BA+0x2C	R/W	I2C Slave address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2Cx_BA+0x30	R/W	I2C Slave address Mask Register3	0x0000_0000

7	6	5	4	3	2	1	0
ADDRMSK							**RESERVED**

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:1]	**ADDRMSK**	**I2C Address Mask Register** 0 = Mask disable. 1 = Mask enable (the received corresponding address bit is don't care.). I2C bus controllers support multiple-address recognition with four address mask registers. Bits in this field mask the ADDRx registers masking bits from the address comparison.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

5.12 12-bit Analog-to-Digital Converter (SARADC)

5.12.1 Overview

ISD91500 contains one 12-bit successive approximation analog-to-digital converters (SARADC). SARADC can be programmed operation independently. SARADC has 12-channel external single-ended inputs. The A/D converter supports three operation modes: single, single-cycle scan and continuous scan mode. Each of A/D converters can be started by software.

5.12.2 Features

- There are max 15 times sharing SARADC channel input. The Ch0-Ch11 is related with external input.
- Analog input voltage range: 0~VCCA
- 12-bits resolution and 10-bits accuracy is guaranteed
- Up to 200KSPS conversion rate
- Three operating modes
  - Single mode: Single channel A/D conversion
  - Single-cycle scan mode: Conversion on all enabled channels once
  - Continuous scan mode: Repetitive conversion on enabled channels
- An A/D conversion can be started by
  - Software write SWTRG bit
- Conversion results are held in data registers for assign channel with valid and overrun indicators
- Conversion result can be compared with specified value and user can select whether to generate an interrupt when conversion result is less than/great or equal to the compare register setting.

5.12.3 SARADC Block Diagram

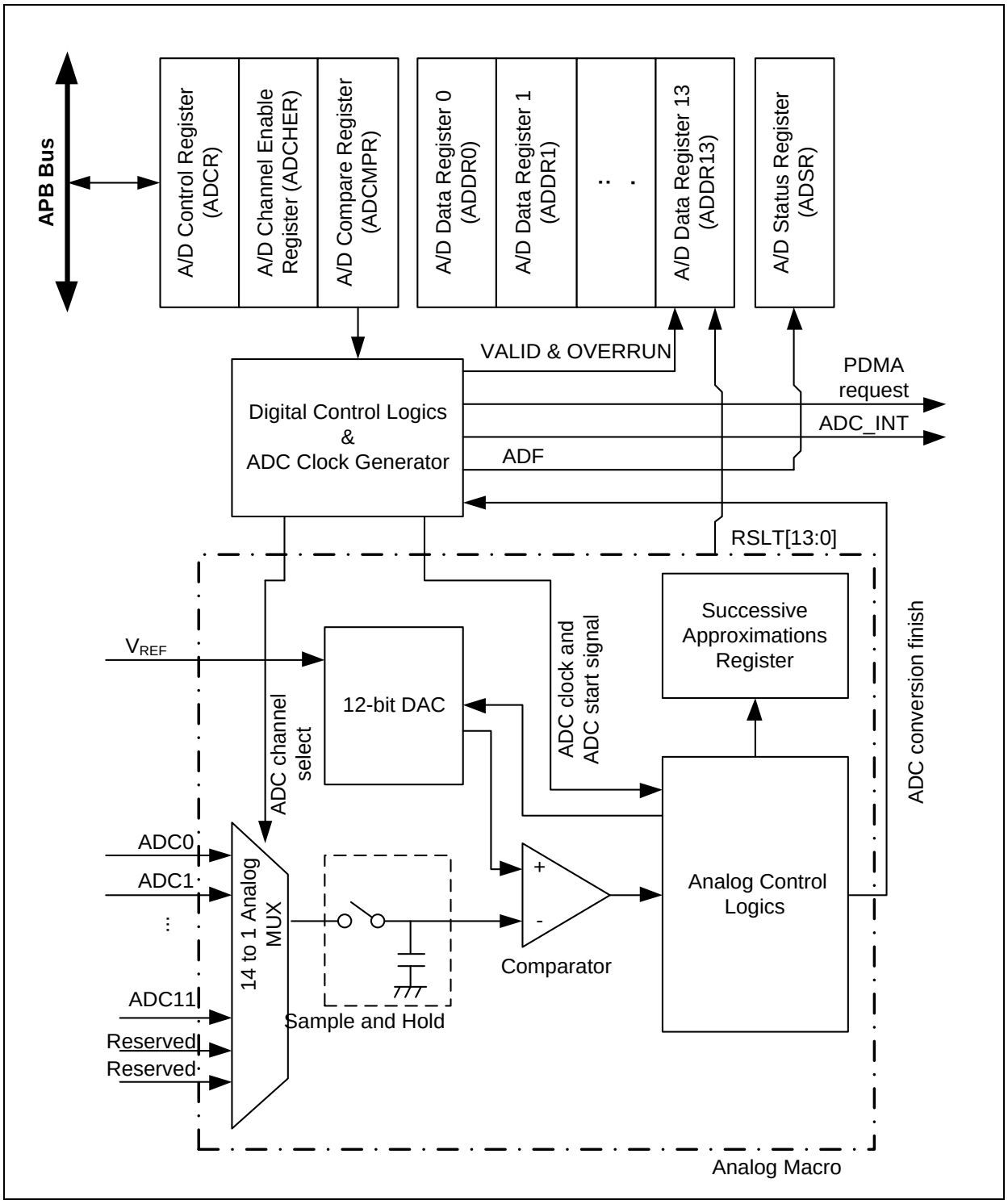


Figure 5.12-1 ADC Controller Block Diagram

5.12.4 SARADC Inputs

SAR INPUT	Signal
Ain0	GPIOA_0
Ain1	GPIOA_1
Ain2	GPIOA_2
Ain3	GPIOA_3
Ain4	GPIOA_4
Ain5	GPIOA_5
Ain6	GPIOA_6
Ain7	GPIOA_7
Ain8	GPIOA_8
Ain9	GPIOA_9
Ain10	GPIOA_10
Ain11	GPIOA_11

5.12.5 SARADC Operation Procedure

The A/D converter operates by successive approximation with 12-bit resolution. The SARADCs have three operation modes: single mode, single-cycle scan mode and continuous mode.

When changing the operating mode or analog input channelsetting, in order to prevent incorrect operation, software must clear SWTRG bit to 0 in SARADC_CTL register. The A/D converter discards current conversion immediately and enters idle state while SWTRG bit is cleared

5.12.5.1 SARADC Clock Generator

The maximum sampling rate is up to 200 KHz and the conversion time is less 2μS. The SARADC clock source can be from HXT, PCLK, PLLFOUT or HIRC oscillator. The selected clock source is divided by (SARADCDIV+1) to produce the clock for setup sampling rate.

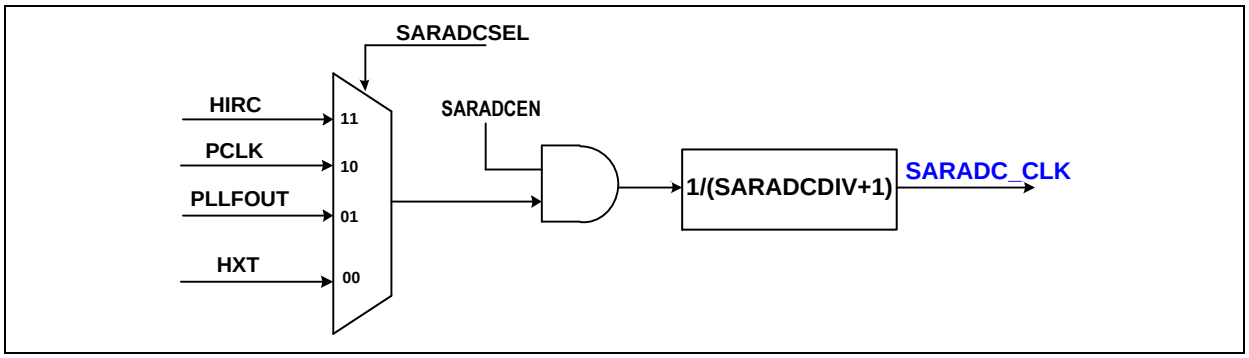


Figure 5.12-2 SARADC Clock Source

5.12.5.2 Single Mode

In single mode, A/D conversion is to be performed only once on the specified channel. The operations are as follows:

1. The channel defined in CHSEQ0 is the channel to be converted.
2. A/D conversion is started when the SWTRG bit in ADC_CTL is set to 1 by software.
3. When A/D conversion is finished, the result is transferred to the A/D data register ADC_DATn corresponding to the channel.
4. On completion of conversion, the ADIF bit in ADC_STATUS is set to 1. If the ADCIE bit is 1, an ADINT interrupt request is generated.
5. The SWTRG bit remains “1” during A/D conversion. When A/D conversion ends, the SWTRG bit is automatically cleared to 0 and the A/D converter enters the idle state.
6. When the SWTRG bit is cleared to 0 during A/D conversion, A/D conversion stops and the A/D converter enters the idle state.

5.12.5.3 Continuous Scan Mode

In continuous scan mode, A/D conversion is to be performed sequentially on the specified channels that are defined by CHSEQx bits in SARADC_CHSEQ0/ SARADC_CHSEQ1 register (12 channels maximum). The operations are as follows:

1. When the SWTRG bit in SARADC_CTL is set to 1 by software, A/D conversion starts on the channel selected by CHSEQ0.
2. When A/D conversion for each channel selected by channel sequence register (SARADC_CHSEQ0/ SARADC_CHSEQ1) is completed, the converted result is sequentially transferred to the A/D data register corresponding to channel sequence.
3. When the conversion for all the selected channels is completed, the ADIF bit in SARADC_STATUS0 is set to 1. If the ADCIE bit is 1, an ADINT interrupt is requested after A/D conversion ends. Conversion of the channel defined in CHSEQ0 starts again.
4. Steps 2 to 3 are repeated as long as the SWTRG bit remains to 1. When the SWTRG bit is cleared to 0, A/D conversion stops and the A/D converter enters the idle state.

An example timing diagram for continuous scan mode on 3 channels (CHSEQ0=4, CHSEQ1=3, CHSEQ2=4, CHSEQ3=2 and CHSEQ4=F) is shown as below:

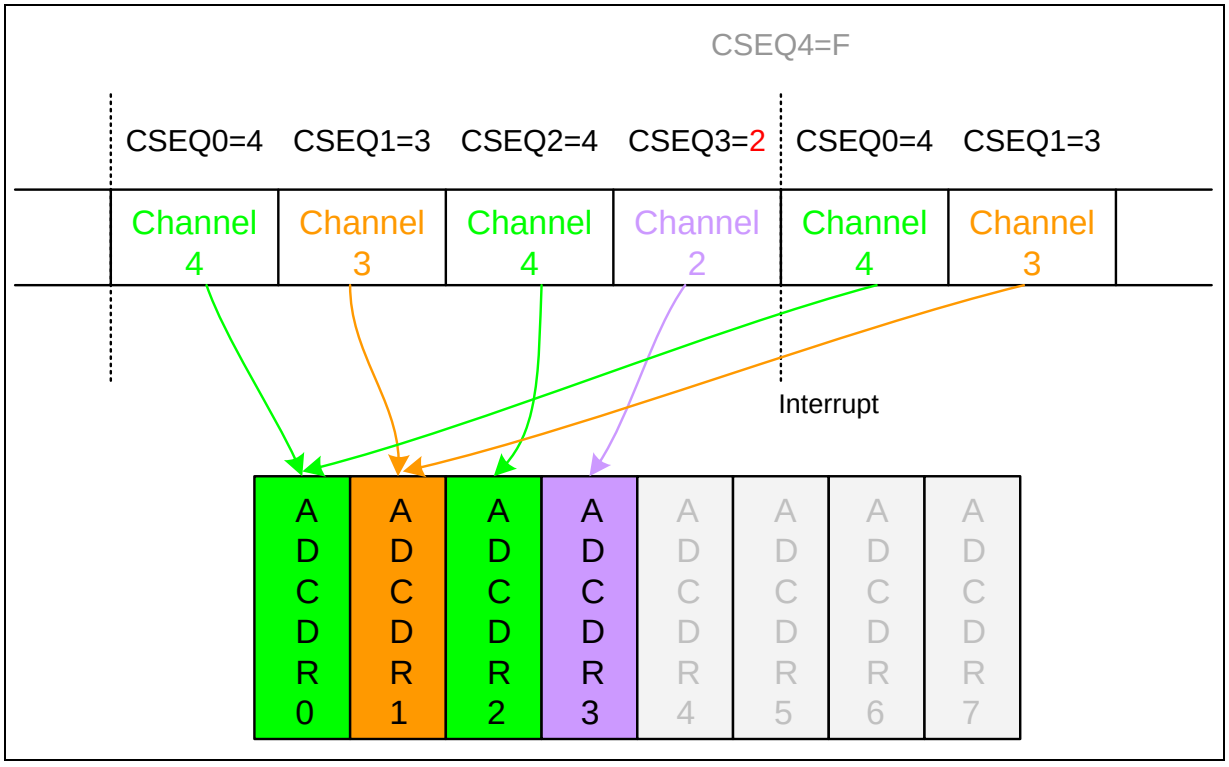


Figure 5.12-3 Continuous Scan on Selected Channels

5.12.5.4 Single-Cycle Scan Mode

In single-cycle scan mode, A/D conversion is to be performed once on the specified channels that are defined by CHSEQx bits in SARADC_CHSEQ0/ SARADC_CHSEQ1 register (14 channels maximum for SARADC). Operations are as follows:

1. When the SWTRG bit in SARADC_CTL is set to 1 by a software, A/D conversion starts on the channel selected by CHSEQ0.
2. When A/D conversion for channel selected by channel sequence registers (CHSEQx) is completed, the result is sequentially transferred to the A/D data register corresponding to channel sequence.
3. When the conversion for all the selected channels is completed, the ADIF bit in SARADC_STATUS0 is set to 1. If the ADCIE bit is 1, an ADINT interrupt is requested after A/D conversion ends.
4. After A/D conversion ends, the SWTRG bit is automatically cleared to 0 and the A/D converter enters the idle state. When the SWTRG bit is cleared to 0 during A/D conversion, A/D conversion stops and the A/D converter enters the idle state.

An example timing diagram for single-cycle scan on 4 channels (CHSEQ0=2, CHSEQ1=4, CHSEQ2=3, CHSEQ3=4 and CHSEQ4=F) is shown as below:

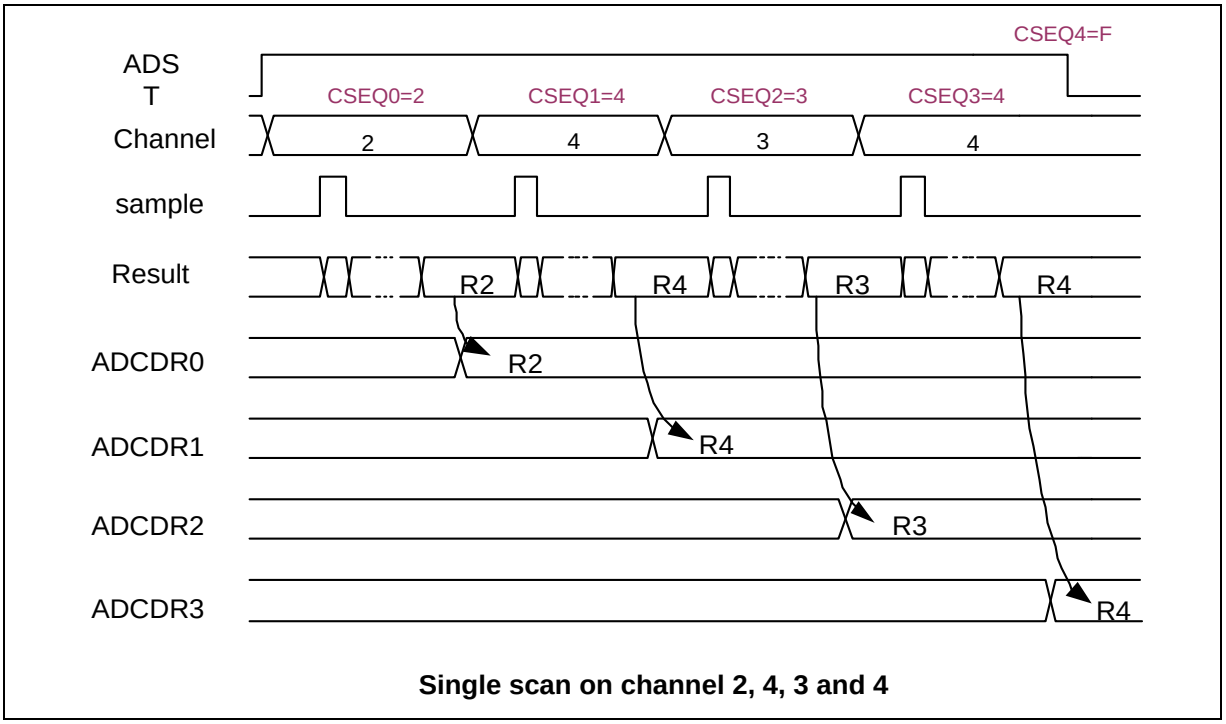


Figure 5.12-4 Single-Cycle Scan on selected Channels

5.12.5.5 Conversion Result Monitor

ISD91500 SARADC controller provides two compare registers ADC_CMP0/1 to monitor specified channel conversion result from A/D conversion module (see figure below). Software can select which channel to be monitored by set CMPCH and CMPCOND bit is used to check conversion result is less than or greater than (equal to) specified value in CMPDAT. When the compared result meets the setting, compare match counter will increase 1. When counter value reaches the setting of CMPMCNT then ADCMPF bit will be set to 1. If ADCMPIE bit is set, then an ADINT interrupt request is generated. The block diagram is shown as below:

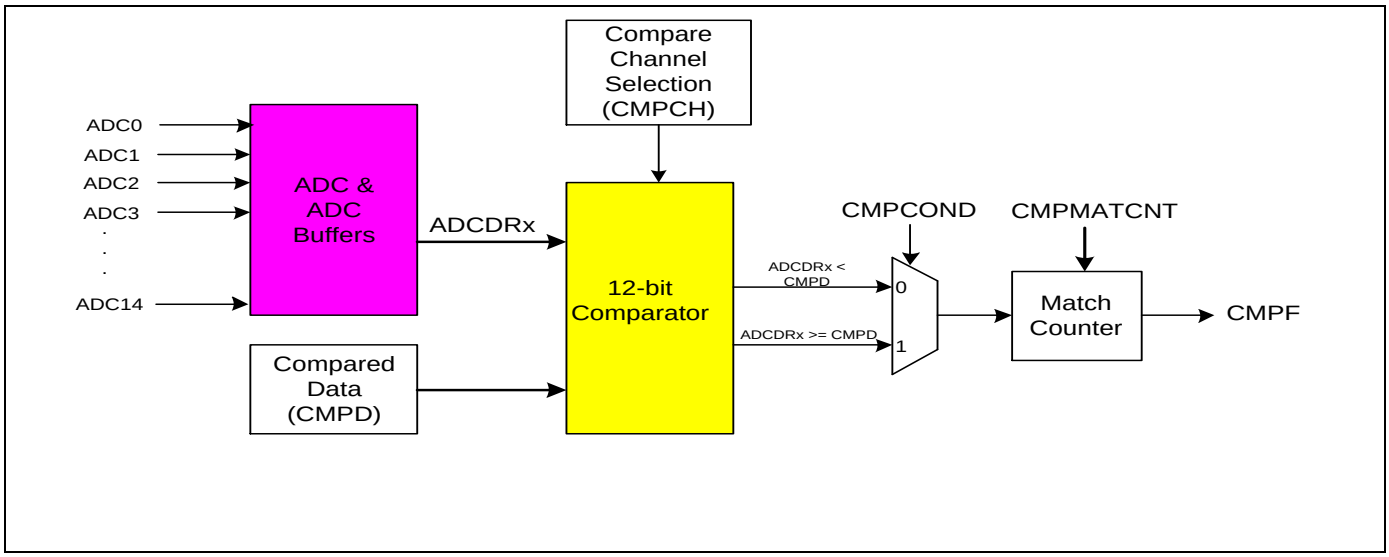


Figure 5.12-5 A/D Conversion Result Comparison

5.12.5.6 Interrupt Sources

The A/D converter generates a conversion end ADIF in SARADC_STATUS0 register upon the end of A/D conversion. If ADCIE bit in SARADC_CTL is set then conversion end interrupt request ADINT is generated. If ADCMPIE bit is enabled, when A/D conversion result meets setting in SARADC_CMPn register, monitor interrupt is generated, ADINT will be set also. The user can set the sample rate by setting CONVN bit in SARADC_HWPARA. When the sample rate set faster than internal conversion time, the OVRF flag will set to 1'b. The sample rate should set lower than maximum 200 KSPS. If OVRIE in SARADC_CTL is set then sample rate over interrupt is generate.

CPU can clear ADCMPF, ADIF and OVRF to stop interrupt request.

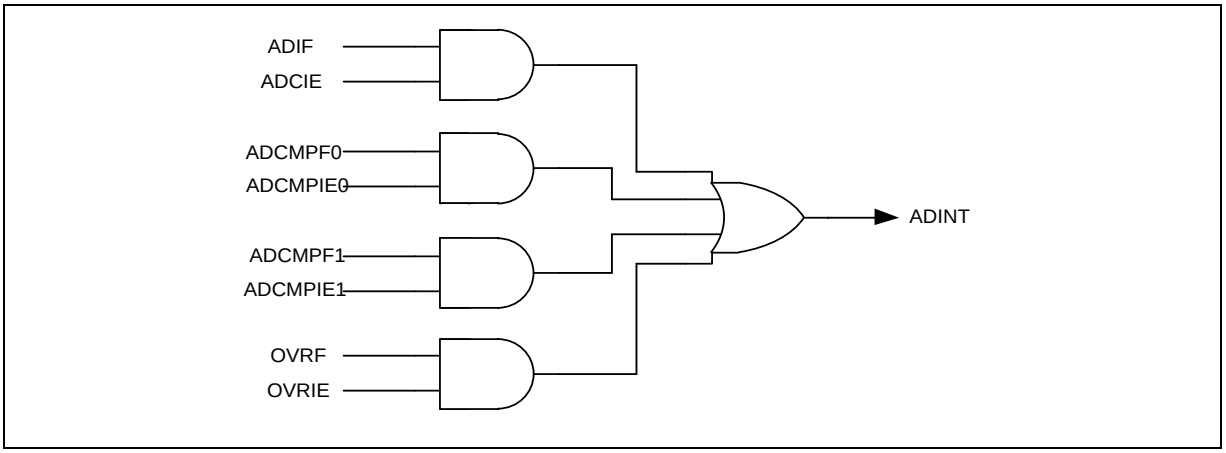


Figure 5.12-6 A/D Controller Interrupt

5.12.6 SARADC Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
SARADC Base Address: **SARADC_BA = 0x400E_0000**				
SARADC_DAT0	SARADC_BA+0x00	R	SARADC Data Register for the channel defined in CHSEQ0	0x0000_0000
SARADC_DAT1	SARADC_BA+0x04	R	SARADC Data Register for the channel defined in CHSEQ1	0x0000_0000
SARADC_DAT2	SARADC_BA+0x08	R	SARADC Data Register for the channel defined in CHSEQ2	0x0000_0000
SARADC_DAT3	SARADC_BA+0x0C	R	SARADC Data Register for the channel defined in CHSEQ3	0x0000_0000
SARADC_DAT4	SARADC_BA+0x10	R	SARADC Data Register for the channel defined in CHSEQ4	0x0000_0000
SARADC_DAT5	SARADC_BA+0x14	R	SARADC Data Register for the channel defined in CHSEQ5	0x0000_0000
SARADC_DAT6	SARADC_BA+0x18	R	SARADC Data Register for the channel defined in CHSEQ6	0x0000_0000
SARADC_DAT7	SARADC_BA+0x1C	R	SARADC Data Register for the channel defined in CHSEQ7	0x0000_0000
SARADC_DAT8	SARADC_BA+0x20	R	SARADC Data Register for the channel defined in CHSEQ8	0x0000_0000
SARADC_DAT9	SARADC_BA+0x24	R	SARADC Data Register for the channel defined in CHSEQ9	0x0000_0000
SARADC_DAT10	SARADC_BA+0x28	R	SARADC Data Register for the channel defined in CHSEQ10	0x0000_0000
SARADC_DAT11	SARADC_BA+0x2C	R	SARADC Data Register for the channel defined in CHSEQ11	0x0000_0000
SARADC_DAT12	SARADC_BA+0x30	R	SARADC Data Register for the channel defined in CHSEQ12	0x0000_0000
SARADC_DAT13	SARADC_BA+0x34	R	SARADC Data Register for the channel defined in CHSEQ13	0x0000_0000
SARADC_CTL	SARADC_BA+0x3C	R/W	SARADC Control Register	0x0000_0040
SARADC_CHSEQ0	SARADC_BA+0x40	R/W	SARADC Channel Sequence Register0	0xFFFF_FFFF
SARADC_CHSEQ1	SARADC_BA+0x44	R/W	SARADC Channel Sequence Register1	0xFFFF_FFFF
SARADC_CMP0	SARADC_BA+0x48	R/W	SARADC Compare Register 0	0x0000_0000
SARADC_CMP1	SARADC_BA+0x4C	R/W	SARADC Compare Register 1	0x0000_0000

SARADC_STATUS0	SARADC_BA+0x50	R/W	SARADC Status Register0	0x0000_00F0
SARADC_STATUS1	SARADC_BA+0x54	R/W	SARADC Status Register1	0x0000_0000
SARADC_PDMADAT	SARADC_BA+0x58	R/W	SARADC PDMA result Register	0x0000_0000
SARADC_HWPARA	SARADC_BA+0x5C	R/W	SARADC H/W Parameter Control Register	0x0000_0B00

5.12.7 SARADC Register Description

SARADC Data Registers (SARADC_DATn)

Register	Offset	R/W	Description	Reset Value
SARADC_DAT0	SARADC_BA+0x00	R	SARADC Data Register for the channel defined in CHSEQ0	0x0000_0000
SARADC_DAT1	SARADC_BA+0x04	R	SARADC Data Register for the channel defined in CHSEQ1	0x0000_0000
SARADC_DAT2	SARADC_BA+0x08	R	SARADC Data Register for the channel defined in CHSEQ2	0x0000_0000
SARADC_DAT3	SARADC_BA+0x0C	R	SARADC Data Register for the channel defined in CHSEQ3	0x0000_0000
SARADC_DAT4	SARADC_BA+0x10	R	SARADC Data Register for the channel defined in CHSEQ4	0x0000_0000
SARADC_DAT5	SARADC_BA+0x14	R	SARADC Data Register for the channel defined in CHSEQ5	0x0000_0000
SARADC_DAT6	SARADC_BA+0x18	R	SARADC Data Register for the channel defined in CHSEQ6	0x0000_0000
SARADC_DAT7	SARADC_BA+0x1C	R	SARADC Data Register for the channel defined in CHSEQ7	0x0000_0000
SARADC_DAT8	SARADC_BA+0x20	R	SARADC Data Register for the channel defined in CHSEQ8	0x0000_0000
SARADC_DAT9	SARADC_BA+0x24	R	SARADC Data Register for the channel defined in CHSEQ9	0x0000_0000
SARADC_DAT10	SARADC_BA+0x28	R	SARADC Data Register for the channel defined in CHSEQ10	0x0000_0000
SARADC_DAT11	SARADC_BA+0x2C	R	SARADC Data Register for the channel defined in CHSEQ11	0x0000_0000
SARADC_DAT12	SARADC_BA+0x30	R	SARADC Data Register for the channel defined in CHSEQ12	0x0000_0000
SARADC_DAT13	SARADC_BA+0x34	R	SARADC Data Register for the channel defined in CHSEQ13	0x0000_0000

The A/D converted results are saved in these SARADC_DATn registers sequentially. The channel scan sequence is defined in SARADC_CHSEQ0/1 register.

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

Bits	Description	
[31:18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17]	**VALID**	**Valid Flag** 0 = Data in RESULT are not valid. 1 = Data in RESULT are valid. This bit is set to 1 when corresponding channel analog input conversion is completed and cleared by hardware after ADC_DAT register is read.
[16]	**OV**	**Over Run Flag** 0 = Data in RESULT are recent conversion result. 1 = Data in RESULT are overwritten. If converted data in RESULT [11:0] have not been read before new conversion result is loaded to this register, OV is set to 1. It is cleared by hardware after ADC_DAT register is read.
[15:12]	**EXTS**	**Extension Bits Of RESULT for Different Data Format** If ADCFM is “0”, EXTS all are read as “0”. If ADCFM is “1”, EXTS all are read as bit RESULT [11].
[11:0]	**RESULT**	**A/D Conversion Result** This field contains the 12-bit conversion result. Its data format is defined by ADCFM bit.

SARADC Control Register (SARADC_CTL)

Register	Offset	R/W	Description	Reset Value
SARADC_CTL	SARADC_BA+0x3C	R/W	SARADC Control Register	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**ADCFM**	**SWTRG**	**OVRIE**	**Reserved**	**MUXEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DLYTRIM		**MUXSW**	**PDMAEN**	**OPMODE**		**ADCIE**	**ADCEN**

Bits	Description	
[31:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12]	**ADCFM**	**Data Format Of ADC Conversion Result** 0 = Unsigned 1 = 2's Complement
[11]	**SWTRG**	**A/D Conversion Start** 0 = Conversion is stopped and A/D converter enters idle state. 1 = Start conversion. **Note1**: SWTRG bit can be reset to 0 by software, or can be cleared to 0 by hardware automatically at the end of single mode and single-cycle scan mode on specified channel. In continuous scan mode, A/D conversion is continuously performed sequentially until software writes 0 to this bit or chip resets. **Note2**: Before trigger SWTRG to start ADC convert , the ADC relative setting should be completed.
[10]	**OVRIE**	**Sample rate over interrupt enable** 0 = Disable 1 = Enable
[9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[8]	**MUXEN**	**Input channel MUX enable control** 0 = Disable(MUX output floating) 1 = Enable **Note: Only works when MUXSW = 1.**
[7:6]	**DLYTRIM**	**Trim bit for SARADC internal conversion time.** When higher level will have longer conversion time. 00 = Level 0 01 = Level 1 10 = Level 2 11 = Level 3 **Note:** Suggest to set Level 3 for best SARADC performance.
[5]	**MUXSW**	**MUXEN software control register** MUXSW is used for MUXEN control. When MUXSW = 0'b, MUX is always enable. 0 : MUX always enable turn on 1 : MUX control by MUXEN
[4]	**PDMAEN**	**PDMA Transfer Enable Bit** When A/D conversion is completed, the converted data is loaded into ADC_DATn (n: 0 ~ 13) register, user can enable this bit to generate a PDMA data transfer request. 0 = PDMA data transfer Disabled. 1 = PDMA data transfer Enabled.
[3:2]	**OPMODE**	**A/D Converter Operation Mode** 00 = Single conversion 01 = Reserved 10 = Single-cycle scan 11 = Continuous scan **Note 1**: When changing the operation mode, software should disable SWTRG bit firstly.
[1]	**ADCIE**	**A/D Interrupt Enable** 0 = Disable A/D interrupt function 1 = Enable A/D interrupt function A/D conversion end interrupt request is generated if ADCIE bit is set to 1.
[0]	**ADCEN**	**A/D Converter Enable** 0 = Disable 1 = Enable Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit power consumption.

SARADC Channel Sequence Register0 (SARADC_CHSEQ0)

Register	Offset	R/W	Description	Reset Value
SARADC_CHSEQ0	SARADC_BA+0x40	R/W	SARADC Channel Sequence Register0	0xFFFF_FFFF

31	30	29	28	27	26	25	24
CHSEQ7				CHSEQ6			
23	22	21	20	19	18	17	16
CHSEQ5				CHSEQ4			
15	14	13	12	11	10	9	8
CHSEQ3				CHSEQ2			
7	6	5	4	3	2	1	0
CHSEQ1				CHSEQ0			

Bits	Description	
[31:28]	**CHSEQ7**	**Select Channel N As The 8th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[27:24]	**CHSEQ6**	**Select Channel N As The 7th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[23:20]	**CHSEQ5**	**Select Channel N As The 6th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[19:16]	**CHSEQ4**	**Select Channel N As The 5th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[15:12]	**CHSEQ3**	**Select Channel N As The 4th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[11:8]	**CHSEQ2**	**Select Channel N As The 3rd Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[7:4]	**CHSEQ1**	**Select Channel N As The 2nd Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.

<table>
<tr>
<td>[3:0]</td>
<td>CHSEQ0</td>
<td>Select Channel N As The 1st Conversion In Scan Sequence
One of the following channel is selected according to CHSEQ0 [3:0].
<table>
<tr><th>CHSEQ0</th><th>Selected channel to ADC input</th></tr>
<tr><td>0000</td><td>Channel 0</td></tr>
<tr><td>0001</td><td>Channel 1</td></tr>
<tr><td>0010</td><td>Channel 2</td></tr>
<tr><td>0011</td><td>Channel 3</td></tr>
<tr><td>0100</td><td>Channel 4</td></tr>
<tr><td>0101</td><td>Channel 5</td></tr>
<tr><td>0110</td><td>Channel 6</td></tr>
<tr><td>0111</td><td>Channel 7</td></tr>
<tr><td>1000</td><td>Channel 8</td></tr>
<tr><td>1000</td><td>Channel 8</td></tr>
<tr><td>1001</td><td>Channel 9</td></tr>
<tr><td>1010</td><td>Channel 10</td></tr>
<tr><td>1011</td><td>Channel 11</td></tr>
<tr><td>Others</td><td>Reserved</td></tr>
</table>
CHSEQ0 = 1111: No channel is selected, scan sequence is stopped.</td>
</tr>
</table>

SARADC Channel Sequence Register1 (SARADC_CHSEQ1)

Register	Offset	R/W	Description	Reset Value
SARADC_CHSEQ1	SARADC_BA+0x44	R/W	SARADC Channel Sequence Register1	0xFFFF_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CHSEQ13				CHSEQ12			
15	14	13	12	11	10	9	8
CHSEQ11				CHSEQ10			
7	6	5	4	3	2	1	0
CHSEQ9				CHSEQ8			

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:20]	**CHSEQ13**	**Select Channel N As The 14th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[19:16]	**CHSEQ12**	**Select Channel N As The 13th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[15:12]	**CHSEQ11**	**Select Channel N As The 12th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[11:8]	**CHSEQ10**	**Select Channel N As The 11th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[7:4]	**CHSEQ9**	**Select Channel N As The 10th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[3:0]	**CHSEQ8**	**Select Channel N As The 9th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.

SARADC Compare Register 0/1 (SARADC_CMPn)

Register	Offset	R/W	Description	Reset Value
SARADC_CMP0	SARADC_BA+0x48	R/W	SARADC Compare Register 0	0x0000_0000
SARADC_CMP1	SARADC_BA+0x4C	R/W	SARADC Compare Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**CMPDAT**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CMPMCNT**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPCH				**Reserved**	**CMPCOND**	**ADCMPIE**	**ADCMPEN**

Bits	Description	
[31:28]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[27:16]	**CMPDAT**	**Compare Data** This field possessing 12-bit compare data, is used to compare with conversion result of specified channel. Software can use it to monitor the external analog input pin voltage transition in scan mode without imposing a load on software. The data format should be consistent with the setting of ADCFM bit.
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**CMPMCNT**	**Compare Match Count** When the specified A/D channel analog conversion result matches the comparing condition, the internal match counter will increase 1. When the internal counter achieves the setting, (CMPMCNT+1) hardware will set the ADCMPF bit.
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[6:3]	**CMPCH**	**Compare Channel Selection** 0000 = Channel 0 conversion result is selected to be compared. 0001 = Channel 1 conversion result is selected to be compared. 0010 = Channel 2 conversion result is selected to be compared. 0011 = Channel 3 conversion result is selected to be compared. 0100 = Channel 4 conversion result is selected to be compared. 0101 = Channel 5 conversion result is selected to be compared. 0110 = Channel 6 conversion result is selected to be compared. 0111 = Channel 7 conversion result is selected to be compared. 1000 = Channel 8 conversion result is selected to be compared. 1001 = Channel 9 conversion result is selected to be compared. 1010 = Channel 10 conversion result is selected to be compared. 1011 = Channel 11 conversion result is selected to be compared. Others = Reserved
[2]	**CMPCOND**	**Compare Condition** 0 = ADCMPFx bit is set if conversion result is less than CMPDAT. 1 = ADCMPFx bit is set if conversion result is greater or equal to CMPDAT.
[1]	**ADCMPIE**	**Compare Interrupt Enable** 0 = Disable 1 = Enable When converted data in RESULT is less (or greater) than the compare data CMPDAT, ADCMPF bit is asserted. If ADCMPIE is set to 1, a compare interrupt request is generated.
[0]	**ADCMPEN**	**Compare Enable** 0 = Disable compare. 1 = Enable compare. Set this bit to 1 to enable the comparison CMPDAT with specified channel conversion result when converted data is loaded into SARADC_DAT register.

SARADC Status Register0 (SARADC_STATUS0)

Register	Offset	R/W	Description	Reset Value
SARADC_STATUS0	SARADC_BA+0x50	R/W	SARADC Status Register0	0x0000_00F0

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							OVRF
7	6	5	4	3	2	1	0
CHANNEL				BUSY	ADCMPF1	ADCMPF0	ADIF

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**OVRF**	**Sampling rate over flag** 0= user setting sample rate not exceed real conversion rate 1= user setting sample rate exceed real conversion rate It is cleared when 1 is written.
[7:4]	**CHANNEL**	**Current Conversion Channel** This filed reflects current conversion channel when BUSY=1. When BUSY=0, it shows the next channel will be converted. It is read only.
[3]	**BUSY**	**BUSY/IDLE** 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion. This bit is mirror of SWTRG bit in SARADC_CTL. It is read only.
[2]	**ADCMPF1**	**Compare Flag1** When the selected channel A/D conversion result meets setting conditions in SARADC_CMP1, then this bit is set to 1. And it is cleared by write 1. 0 = Converted result RESULT in SARADC_DAT does not meet SARADC_CMP1 setting. 1 = Converted result RESULT in SARADC_DAT meets SARADC_CMP1 setting.

[1]	**ADCMPF0**	**Compare Flag0** When the selected channel A/D conversion result meets setting conditions in SARADC_CMP0, then this bit is set to 1. And it is cleared by write 1. 0 = Converted result RESULT in SARADC_DAT does not meet SARADC_CMP0 setting. 1 = Converted result RESULT in SARADC_DAT meets SARADC_CMP0 setting,
[0]	**ADIF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. ADIF is set to 1 under the following two conditions: 1. When A/D conversion ends in single mode, 2. When A/D conversion ends on all channels specified by channel sequence register in scan mode. And it is cleared when 1 is written.

SARADC Status Register1 (SARADC_STATUS1)

Register	Offset	R/W	Description	Reset Value
SARADC_STATUS1	SARADC_BA+0x54	R/W	SARADC Status Register1	0x0000_0000

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

Bits	Description	
[31]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[30:16]	**OV**	**Over Run Flag** It is a mirror to OV bit in SARADC_DATn.
[15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14:0]	**VALID**	**Data Valid Flag** It is a mirror of VALID bit in SARADC_DATn.

SARADC PDMA Result Register (SARADC PDMADAT)

Register	Offset	R/W	Description	Reset Value
SARADC_PDMADAT	SARADC_BA+0x58	R/W	SARADC PDMA result Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**DATA**	**SARADC PDMA transfer data** This register is a shadow register of SARADC_DATn (n=0~14) for PDMA support. This is a read only register.

SARADC H/W Parameter Control Register (SARADC_HWPARA)

Register	Offset	R/W	Description	Reset Value
SARADC_HWPARA	SARADC_BA+0x5C	R/W	SARADC H/W Parameter Control Register	0x0000_0B00

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

Bits	Description	
[31:15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14:8]	**CONVN**	**Specify SARADC conversion clock number** SARADC Conversion clock number = (CONVN + 1). To update this field, programmer can only revise bit [14:8] and keep other bits the same as before. **Note:** CONVN value must bigger than SHCLKN value and should bigger than 2us.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:0]	**SHCLKN**	**Specify the high level of SARADC start signal.** SARADC start signal high level duration time = SARADC_CLK x (SHCLKN + 1). **Note**: SHCLKN must larger than 400ns.

5.13 PDMA Controller

5.13.1 Overview

The ISD91500 incorporates a Peripheral Direct Memory Access (PDMA) controller that transfers data between SRAM and APB devices. The PDMA has eight channels of DMA (CH0~CH7). PDMA transfers are unidirectional and can be Peripheral-to-SRAM, SRAM-to-Peripheral or SRAM-to-SRAM.

The peripherals available for PDMA transfer are UART0/1, SPI0/1, I2S0, SARADC, SDADC and DAC.

PDMA operation is controlled for each channel by configuring a source and destination address and specifying a number of bytes to transfer. Source and destination addresses can be fixed automatically increment or wrap around a circular buffer. When PDMA operation is complete, controller can be configured to provide CPU with an interrupt.

5.13.2 Features

- Supports 8 independently configurable channels
- Supports transfer data width of 8, 16, and 32 bits
- Provides access to UART0/1, SPI0, I2S0, SARADC, SDADC and DAC peripherals.
- AMBA AHB master/slave interface, transfers can occur concurrently with CPU access to flash memory.
- PDMA source and destination addressing modes allow fixed, incrementing, and wrap-around addressing.

5.13.3 Block Diagram

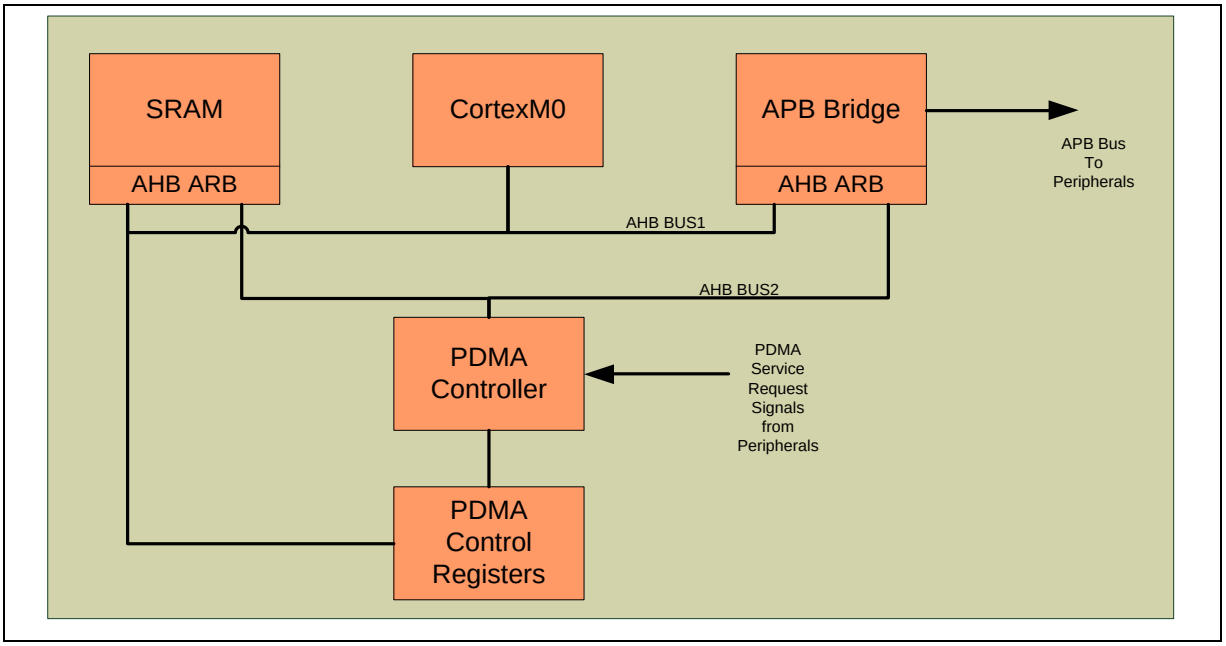


Figure 5.13-1 PDMA Controller Block Diagram

5.13.4 Functional Description

The PDMA controller has eight channels of DMA, each channel can be configured to one of the following transfer types: Peripheral-to-SRAM SRAM-to-Peripheral or SRAM-to-SRAM. The SRAM and the AHB-APB bus bridge each has an AHB bus arbiter that allows AHB bus access to occur either from the CPU or the PDMA controller. The PDMA controller requests bus transfers over the AHB bus from one address into a single word buffer within the PDMA controller then writes this buffer to another address over the AHB bus. Peripherals with PDMA capability generate control signals to the PDMA block requesting service when they need data (Rx request) or have data to transfer (Tx request). The PDMA control registers reside in address space on the AHB bus.

Transfer completion can be determined by polling of status registers or by generation of PDMA interrupt to CPU. A transfer is set up as a specified number of bytes from a source address to a destination address. Both source and destination address can be configured as a fixed address, an incrementing address or a wrap-around buffer address.

The general procedure to operate a DMA channel is as follows:

- Enable PDMA channel *n* clock by setting PDMA->HCLK*n*_EN
- Enable PDMA channel *n* by setting PDMA->channel[*n*].CSR.PDMACEN
- Set source address in PDMA->channel[*n*].SAR
- Set destination address in PDMA->channel[*n*].DAR
- Set the transfer count in PDMA->channel[*n*].BCR
- Set transfer mode and address increment mode in PDMA->channel[*n*].CSR
- Route peripheral PDMA request signal to channel *n* in service selection register.
- Trigger transfer PDMA->channel[*n*].CSR.TRGEN

If the source or destination address is not in wraparound mode, the PDMA will continue the transfer until PDMA->channel[n].CBCR decrements to zero (CBCR is initialized to BCR, in wraparound mode, CBCR will reload and continue until PDMACEN is disabled). If an error occurs during the PDMA operation, the channel stops until software clears the error condition and sets the PDMA->channel[n].CSR.SWRST bit to reset the PDMA channel. After reset the PDMACEN and TRGEN bits would need to be set to start a new operation.

5.13.5 PDMA Controller Register Map

R: read only, **W**: write only, **R/W**: both read and write, **C**: Only value 0 can be written

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMAx_BA = 0x5000_9000 + (0x100 * x)** **x = 0,1,2,3,4,5,6,7** **PDMA_GCR_BA = 0x5000_9F00**				
PDMA_CSR	PDMAx_BA+0x00	R/W	PDMA Channel x Control Register	0x0000_0000
PDMA_SAR	PDMAx_BA+0x04	R/W	PDMA Channel x Source Address Register	0x0000_0000
PDMA_DAR	PDMAx_BA+0x08	R/W	PDMA Channel x Destination Address Register	0x0000_0000
PDMA_BCR	PDMAx_BA+0x0C	R/W	PDMA Channel x Transfer Byte Count Register	0x0000_0000
PDMA_POINT	PDMAx_BA+0x10	R	PDMA Channel x Internal buffer pointer Register	0xXXXX_0000
PDMA_CSAR	PDMAx_BA+0x14	R	PDMA Channel x Current Source Address Register	0x0000_0000
PDMA_CDAR	PDMAx_BA+0x18	R	PDMA Channel x Current Destination Address Register	0x0000_0000
PDMA_CBCR	PDMAx_BA+0x1C	R	PDMA Channel x Current Transfer Byte Count Register	0x0000_0000
PDMA_IER	PDMAx_BA+0x20	R/W	PDMA Channel x Interrupt Enable Register	0x0000_0001
PDMA_ISR	PDMAx_BA+0x24	R/W	PDMA Channel x Interrupt Status Register	0x0X0X_0000
PDMA_GCRCSR	PDMA_GCR_BA+0x00	R/W	PDMA Global Control Register	0x0000_0000
PDMA_PDSSR0	PDMA_GCR_BA+0x04	R/W	PDMA Service Selection Control Register 0	0xFFFF_FFFF
PDMA_PDSSR1	PDMA_GCR_BA+0x08	R/W	PDMA Service Selection Control Register 1	0xFFFF_FFFF
PDMA_GCRISR	PDMA_GCR_BA+0x0C	R	PDMA Global Interrupt Status Register	0x0000_0000

5.13.6 PDMA Controller Register Description

PDMA Channel x Controller Register (PDMA_CSR)

Register	Offset	R/W	Description	Reset Value
PDMA_CSR	PDMAx_BA+0x00	R/W	PDMA Channel x Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TRGEN	**Reserved**		**APBTWS**		**Reserved**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
WAINTSEL				**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DASEL		**SASEL**		**MODESEL**		**SWRST**	**PDMACEN**

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23]	**TRGEN**	**Trigger Enable – Start a PDMA operation.** 0 = Write: no effect. Read: Idle/Finished. 1 = Enable PDMA data read or write transfer. **Note:** When PDMA transfer completed, this bit will be cleared automatically. If a bus error occurs, all PDMA transfer will be stopped. Software must reset PDMA channel, and then trigger again.
[22:21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[20:19]	**APBTWS**	**Peripheral Transfer Width Select.** This parameter determines the data width to be transferred each PDMA transfer operation. 00 = One word (32 bits) is transferred for every PDMA operation. 01 = One byte (8 bits) is transferred for every PDMA operation. 10 = One half-word (16 bits) is transferred for every PDMA operation. 11 = Reserved. **Note:** This field is meaningful only when **MODESEL** is IP to Memory mode (APB-to-Memory) or Memory to IP mode (Memory-to-APB).

[18:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:12]	**WAINTSEL**	**Wrap Interrupt Select** x1xx: If this bit is set, and wraparound mode is in operation a Wrap Interrupt can be generated when half each PDMA transfer is complete. For example if BCR=32 then an interrupt could be generated when 16 bytes were sent. xxx1: If this bit is set, and wraparound mode is in operation a Wrap Interrupt can be generated when each PDMA transfer is wrapped. For example if BCR=32 then an interrupt could be generated when 32 bytes were sent and PDMA wraps around. x1x1: Both half and w interrupts generated.
[11:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:6]	**DASEL**	**Destination Address Select** This parameter determines the behavior of the current destination address register with each PDMA transfer. It can either be fixed, incremented or wrapped. 00 = Transfer Destination Address is incremented. 01 = Reserved. 10 = Transfer Destination Address is fixed (Used when data transferred from multiple addresses to a single destination such as peripheral FIFO input). 11 = Transfer Destination Address is wrapped. When CBCR (Current Byte Count) equals zero, the CDAR (Current Destination Address) and CBCR registers will be reloaded from the DAR (Destination Address) and BCR (Byte Count) registers automatically and PDMA will start another transfer. Cycle continues until software sets PDMA_EN=0. When PDMA_EN is disabled, the PDMA will complete the active transfer but the remaining data in the SBUF will not be transferred to the destination address.
[5:4]	**SASEL**	**Source Address Select** This parameter determines the behavior of the current source address register with each PDMA transfer. It can either be fixed, incremented or wrapped. 00 = Transfer Source address is incremented. 01 = Reserved. 10 = Transfer Source address is fixed 11 = Transfer Source address is wrapped. When CBCR (Current Byte Count) equals zero, the CSAR (Current Source Address) and CBCR registers will be reloaded from the SAR (Source Address) and BCR (Byte Count) registers automatically and PDMA will start another transfer. Cycle continues until software sets PDMACEN=0. When PDMACEN is disabled, the PDMA will complete the active transfer but the remaining data in the SBUF will not be transferred to the destination address.

[3:2]	**MODESEL**	**PDMA Mode Select** This parameter selects to transfer direction of the PDMA channel. Possible values are: 00 = Memory to Memory mode (SRAM-to-SRAM). 01 = IP to Memory mode (APB-to-SRAM). 10 = Memory to IP mode (SRAM-to-APB).
[1]	**SWRST**	**Software Engine Reset** 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset the internal state machine and pointers. The contents of the control register will not be cleared. This bit will auto clear after a few clock cycles.
[0]	**PDMACEN**	**PDMA Channel Enable** Setting this bit to 1 enables PDMA's operation. If this bit is cleared, PDMA will ignore all PDMA request and force Bus Master into IDLE state.

PDMA Channel x Source Address Register (PDMA_SAR)

Register	Offset	R/W	Description	Reset Value
PDMA_SAR	PDMAx_BA+0x04	R/W	PDMA Channel x Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**SAR**	**PDMA Transfer Source Address Register** This register holds the initial Source Address of PDMA transfer. **Note:** The source address must be word aligned.

PDMA Channel x Destination Address Register (PDMA_DAR)

Register	Offset	R/W	Description	Reset Value
PDMA_DAR	PDMAx_BA+0x08	R/W	PDMA Channel x Destination Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**DAR**	**PDMA Transfer Destination Address Register** This register holds the initial Destination Address of PDMA transfer. **Note:** The destination address must be word aligned.

PDMA Channel x Transfer Byte Count Register (PDMA_BCR)

Register	Offset	R/W	Description	Reset Value
PDMA_BCR	PDMAx_BA+0x0C	R/W	PDMA Channel x Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**BCR**	**PDMA Transfer Byte Count Register** This register controls the transfer byte count of PDMA. Maximum value is 0xFFFF. **Note:** The transfer byte count must be word aligned, that is multiples of 4bytes.

PDMA Channel x Internal Buffer Pointer Register (PDMA_POINT)

Register	Offset	R/W	Description	Reset Value
PDMA_POINT	PDMAx_BA+0x10	R	PDMA Channel x Internal buffer pointer Register	0xXXXX_0000

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

Bits	Description	
[31:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3:0]	**POINT**	**PDMA Internal Buffer Pointer Register (Read Only)** A PDMA transaction consists of two stages, a read from the source address and a write to the destination address. Internally this data is buffered in a 32bit register. If transaction width between the read and write transactions are different, this register tracks which byte/half-word of the internal buffer is being processed by the current transaction.

PDMA Channel x Current Source Address Register (PDMA_CSAR)

Register	Offset	R/W	Description	Reset Value
PDMA_CSAR	PDMAx_BA+0x14	R	PDMA Channel x Current Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**CSAR**	**PDMA Current Source Address Register (Read Only)** This register returns the source address from which the PDMA transfer is occurring. This register is loaded from SAR when PDMA is triggered or when a wraparound occurs.

PDMA Channel x Current Destination Address Register (PDMA_CDAR)

Register	Offset	R/W	Description	Reset Value
PDMA_CDAR	PDMAx_BA+0x18	R	PDMA Channel x Current Destination Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**CDAR**	**PDMA Current Destination Address Register (Read Only)** This register returns the destination address to which the PDMA transfer is occurring. This register is loaded from DAR when PDMA is triggered or when a wraparound occurs.

PDMA Channel x Current Transfer Byte Count Register (PDMA_CBCR)

Register	Offset	R/W	Description	Reset Value
PDMA_CBCR	PDMAx_BA+0x1C	R	PDMA Channel x Current Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**CBCR**	**PDMA Current Byte Count Register (Read Only)** This field indicates the current remaining byte count of PDMA transfer. This register is initialized with BCR register when PDMA is triggered or when a wraparound occurs

PDMA Channel x Interrupt Enable Control Register (PDMA_IER)

Register	Offset	R/W	Description	Reset Value
PDMA_IER	PDMAx_BA+0x20	R/W	PDMA Channel x Interrupt Enable Register	0x0000_0001

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**WRAPIEN**	**Wraparound Interrupt Enable** If enabled, and channel source or destination address is in wraparound mode, the PDMA controller will generate a WRAP interrupt to the CPU according to the setting of CSR.WAINTSEL. This can be interrupts when the transaction has finished and has wrapped around and/or when the transaction is half way in progress. This allows the efficient implementation of circular buffers for DMA. 0 = Disable Wraparound PDMA interrupt generation. 1 = Enable Wraparound interrupt generation.
[1]	**TXIEN**	**PDMA Transfer Done Interrupt Enable** If enabled, the PDMA controller will generate and interrupt to the CPU when the requested PDMA transfer is complete. 0 = Disable PDMA transfer done interrupt generation. 1 = Enable PDMA transfer done interrupt generation.
[0]	**ABTIEN**	**PDMA Read/Write Target Abort Interrupt Enable** If enabled, the PDMA controller will generate and interrupt to the CPU whenever a PDMA transaction is aborted due to an error. If a transfer is aborted, PDMA channel must be reset to resume DMA operation. 0 = Disable PDMA transfer target abort interrupt generation. 1 = Enable PDMA transfer target abort interrupt generation.

PDMA Channel x Interrupt Status Register (PDMA_ISR)

Register	Offset	R/W	Description	Reset Value
PDMA_ISR	PDMAx_BA+0x24	R/W	PDMA Channel x Interrupt Status Register	0x0X0X_0000

31	30	29	28	27	26	25	24
INTR	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**WRAPIF**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**TXIF**	**ABTIF**

Bits	Description	
[31]	**INTR**	**Interrupt Pin Status (Read Only)** This bit is the Interrupt pin status of PDMA channel.
[30:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**WRAPIF**	**Wrap around transfer byte count interrupt flag.** These flags are set whenever the conditions for a wraparound interrupt (complete or half complete) are met. They are cleared by writing one to the bits. 0001 = Current transfer finished flag (CBCR==0). 0100 = Current transfer half complete flag (CBCR==BCR/2).
[7:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**TXIF**	**Block Transfer Done Interrupt Flag** This bit indicates that PDMA block transfer complete interrupt has been generated. It is cleared by writing 1 to the bit. 0 = Transfer ongoing or Idle. 1 = Transfer Complete.

[0]	**ABTIF**	**PDMA Read/Write Target Abort Interrupt Flag** This flag indicates a Target Abort interrupt condition has occurred. This condition can happen if attempt is made to read/write from invalid or non-existent memory space. It occurs when PDMA controller receives a bus error from AHB master. Upon occurrence PDMA will stop transfer and go to idle state. To resume, software must reset PDMA channel and initiate transfer again. 0 = No bus ERROR response received. 1 = Bus ERROR response received. **Note:** This bit is cleared by writing 1 to itself.

PDMA Global Control Register (PDMA_GCRCSR)

Register	Offset	R/W	Description	Reset Value
PDMA_GCRCSR	PDMA_GCR_BA+0x00	R/W	PDMA Global Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:8]	**HCLKEN**	**PDMA Controller Channel Clock Enable Control** To enable clock for channel *n* HCLKEN[*n*] must be set. HCLKEN[*n*]=1: Enable Channel *n* clock HCLKEN[*n*]=0: Disable Channel *n* clock
[7:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**RST**	**PDMA Software Reset** 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset the internal state machine and pointers. The contents of control register will be cleared. This bit will auto clear after several clock cycles. **Note:** This bit can reset all channels register(global reset) , but not reset each channel internal state machine.

PDMA Service Selection Control Register 0 (PDMA_PDSSR0)

Register	Offset	R/W	Description	Reset Value
PDMA_PDSSR0	PDMA_GCR_BA+0x04	R/W	PDMA Service Selection Control Register 0	0xFFFF_FFFF

PDMA peripherals have transmit and/or receive request signals to control dataflow during PDMA transfers. These signals must be connected to the PDMA channel assigned by software for use with that peripheral. For instance if PDMA Channel 1 is to be used to transfer data from memory to DAC peripheral, then DACTXSEL should be set to 2. This will route the DAC transmit request signal to PDMA channel 1, whenever DAC has space in FIFO it will request transmission of data from PDMA. When not used the selection should be set to 0x0.

And please noted that one PDMA channel only can support one function at one time.If using multiple functions with PDMA at the same time, different function should select different PDMA channel for application.

31	30	29	28	27	26	25	24
UART0TXSEL				**UART0RXSEL**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
UART1TXSEL				**UART1RXSEL**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
I2S0TXSEL				**I2S0RXSEL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPI0TXSEL				**SPI0RXSEL**			

Bits	Description	
[31:28]	**UART0TXSEL**	**PDMA UART0 Transmit Selection** This field defines which PDMA channel is connected to UART0 peripheral transmit (PDMA destination) request. n = Select channel n.(n= 0~7) Others = Reserved.
[27:24]	**UART0RXSEL**	**PDMA UART0 Receive Selection** This field defines which PDMA channel is connected to UART0 peripheral receive (PDMA source) request. n = Select channel n.(n= 0~7) Others = Reserved.

[23:20]	**UART1TXSEL**	**PDMA UART1 Transmit Selection** This field defines which PDMA channel is connected to UART1 peripheral transmit (PDMA destination) request. n = Select channel n.(n= 0~7) Others = Reserved.
[19:16]	**UART1RXSEL**	**PDMA UART1 Receive Selection** This field defines which PDMA channel is connected to UART1 peripheral receive (PDMA source) request. n = Select channel n.(n= 0~7) Others = Reserved.
[15:12]	**I2S0TXSEL**	**PDMA I2S0 Transmit Selection** This field defines which PDMA channel is connected to I2S0 peripheral transmit (PDMA destination) request. n = Select channel n.(n= 0~7) Others = Reserved.
[11:8]	**I2S0RXSEL**	**PDMA I2S Receive Selection** This field defines which PDMA channel is connected to I2S0 peripheral receive (PDMA source) request. n = Select channel n.(n= 0~7) Others = Reserved.
[7:4]	**SPI0TXSEL**	**PDMA SPI0 Transmit Selection** This field defines which PDMA channel is connected to SPI0 peripheral transmit (PDMA destination) request. n = Select channel n.(n= 0~7) Others = Reserved.
[3:0]	**SPI0RXSEL**	**PDMA SPI0 Receive Selection** This field defines which PDMA channel is connected to SPI0 peripheral receive (PDMA source) request. n = Select channel n.(n= 0~7) Others = Reserved.

PDMA Service Selection Control Register 1 (PDMA_PDSSR1)

Register	Offset	R/W	Description	Reset Value
PDMA_PDSSR1	PDMA_GCR_BA+0x08	R/W	PDMA Service Selection Control Register 1	0xFFFF_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SPI1TXSEL				**SPI1RXSEL**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**SARADCSEL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DACTXSEL				**SDADCSEL**			

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:20]	**SPI1TXSEL**	**PDMA SPI1 Transmit Selection** This field defines which PDMA channel is connected to SPI1 peripheral transmit (PDMA destination) request. n = Select channel n.(n= 0~7) Others = Reserved.
[19:16]	**SPI1RXSEL**	**PDMA SPI1 Receive Selection** This field defines which PDMA channel is connected to SPI1 peripheral receive (PDMA source) request. n = Select channel n.(n= 0~7) Others = Reserved.
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**SARADCSEL**	**PDMA SARADC Receive Selection** This field defines which PDMA channel is connected to SARADC peripheral receive (PDMA source) request. n = Select channel n.(n= 0~7) Others = Reserved.

[7:4]	**DACTXSEL**	**PDMA DAC Transmit Selection** This field defines which PDMA channel is connected to DAC peripheral transmit (PDMA destination) request. n = Select channel n.(n= 0~7) Others = Reserved.
[3:0]	**SDADCSEL**	**PDMA SDADC Receive Selection** This field defines which PDMA channel is connected to SDADC peripheral receive (PDMA source) request. n = Select channel n.(n= 0~7) Others = Reserved.

PDMA Global Interrupt Status Register (PDMA_GCRISR)

Register	Offset	R/W	Description	Reset Value
PDMA_GCRISR	PDMA_GCR_BA+0x0C	R	PDMA Global Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**GCRISR**	**Interrupt Pin Status (Read Only)** GCRISR[*n*] is the interrupt status of PDMA channel *n.*

5.14 UART Interface Controller

5.14.1 Overview

The ISD91500 includes two channels of Universal Asynchronous Receiver/Transmitter (UART). The UART supports normal speed operation and flow control functions.

The Universal Asynchronous Receiver/Transmitter (UART) performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the CPU.The UART channel supports six types of interrupts including transmitter FIFO empty interrupt(THREINT), receiver threshold level interrupt (RDAINT), line status interrupt (overrun error or parity error or framing error or break interrupt) (RLSINT), time out interrupt (RXTOINT), MODEM status interrupt (MODEMINT) and Buffer error interrupt (BUFERRINT) .

5.14.2 Features

- Full duplex, asynchronous communications
- Supports 16 byte FIFO for receive and transmit data payloads.
- Hardware auto-flow control function (/CTS, /RTS) supported.
- Programmable baud-rate generator max up to 4M(additional 1000000/1843200/3250000).
- Programmable receiver buffer trigger level.
- Supports 7-bit RX FIFO time-out detection function
- Supports break error, frame error, parity error and receive/transmit buffer overflow detection function
- Fully programmable serial-interface characteristics:
  - 5-, 6-, 7-, or 8-bit character.
  - Even, odd, or no-parity bit generation and detection.
  - 1-, 1&1/2, or 2-stop bit generation.
  - Baud rate generation.
  - False start bit detection.
- Support PDMA transfer function

5.14.3 Block Diagram

The UART clock control and block diagram are shown as following.

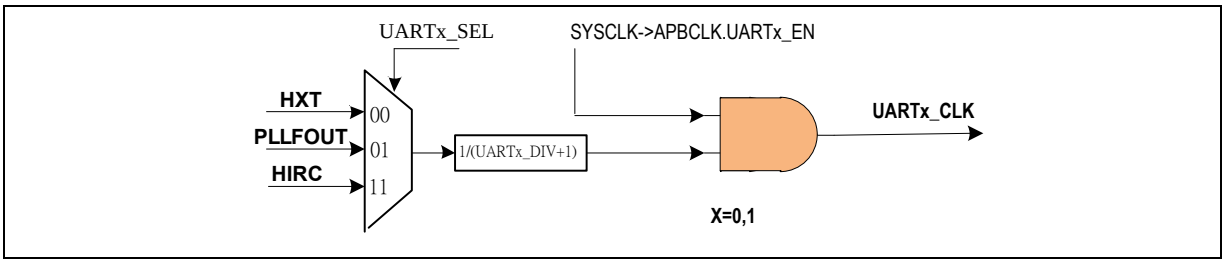


Figure 5.14-1 UART Clock Control Diagram

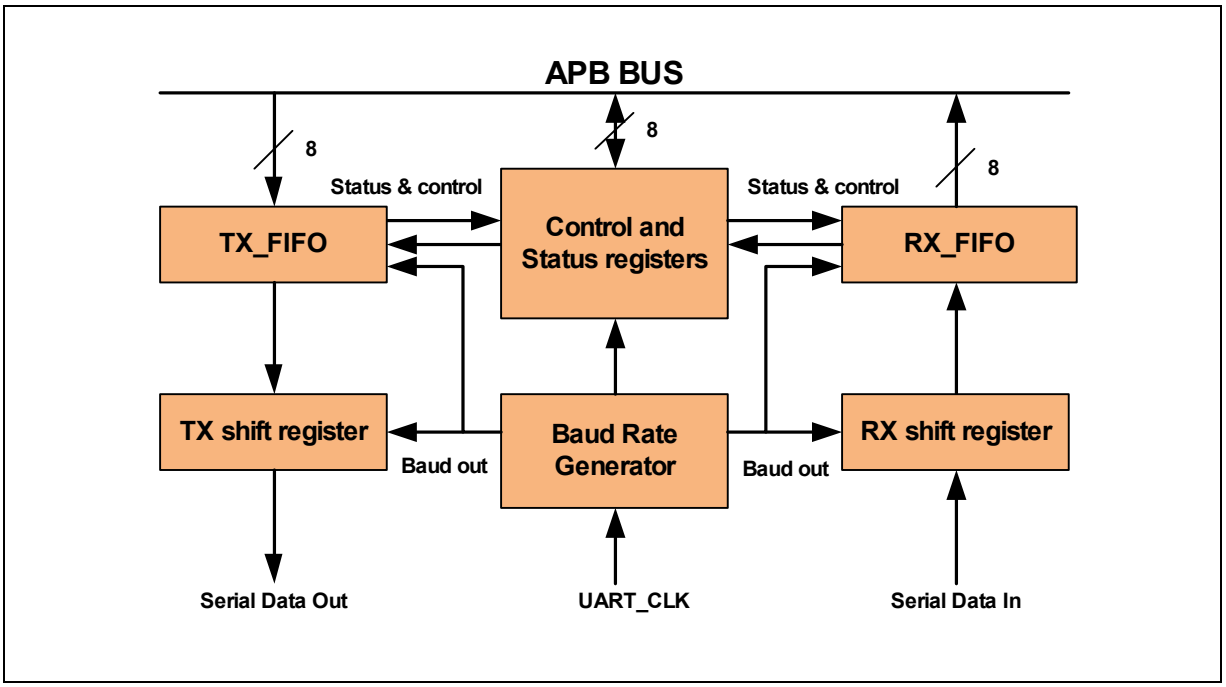


Figure 5.14-2 UART Block Diagram

5.14.4 Functional Description

The UART has a 16-byte transmit FIFO (TX_FIFO) and a 16-byte receive FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU. The CPU can read the status of the UART at any time during the operation. The reported status information includes the type and condition of the transfer operations being performed by the UART, as well as 4 error conditions (parity error, overrun error, framing error and break interrupt) that can occur while receiving data. The UART includes a programmable baud rate generator that is capable of dividing master clock input by divisors to produce the baud rate clock. The baud rate equation is Baud Rate = UART_CLK / M * [BRD + 2], where M and BRD are defined in Baud Rate Divider Register (UARTn->BAUD). Table 5.14-1 lists the equations under various conditions.

The UART controller supports auto-flow control function that uses two active-low signals,/CTS (clear-to-send) and /RTS (request-to-send), to control the flow of data transfer between the UART and external devices (e.g. Modem). When auto-flow is enabled, the UART will not receive data until the UART asserts /RTS to external device. When the number of bytes in the Rx FIFO equals the value of UART->FIFO.RTSTRGLV, the /RTS is de-asserted. The UART sends data out when UART controller detects /CTS is asserted from external device. If /CTS is not detected the UART controller will not send data out.

Table 5.14-1 UART Baud Rate Equation

Mode	BAUDM1	BAUDM0	EDIVM1[3:0]	BRD[15:0]	Baud rate equation
0	0	0	Don't care	A	UART_CLK / [16 * (A+2)]
1	1	0	B	A	UART_CLK / [(B+1) * (A+2)] , B ≥ 8
2	1	1	Don't care	A	UART_CLK / (A+2), A ≥ 10

Table 5.14-2 UART Baud Rate Setting Table

System clock = 49.152MHz						
Baud rate	Mode0	%err	Mode1	%err	Mode2	%err
3250000	x	-	x	-	A=13	0.8
1843200	x	-	x	-	A=25	-1.2
1000000	x	-	x	-	A=47	0.3
921600	x	-	A=4,B=8	1.2	A=51	-0.6
460800	x	-	A=10,B=8	1.2	A=104	0.3
230400	x	-	A=22,B=8 A=7,B=11	1.2 1.2	A=211	-0.2
115200	A=25	1.2	A=37,B=10 A=31,B=12	0.5 0.5	A=425	0.1
57600	A=51	-0.6	A=59,B=13 A=93,B=8	0.1 0.2	A=851	0.0
38400	A=78	0.0	A=126,B=9 A=78,B=15	0.0 0.0	A=1278	0.0

19200	A=158	0.0	A=254,B=9 A=158,B=15	0.0 0.0	A=2558	0.0
9600	A=318	0.0	A=510,B=9 A=318,B=15	0.0 0.0	A=5118	0.0
4800	A=638	0.0	A=1022,B=9 A=638,B=15	0.0 0.0	A=10238	0.0

Note: For UART maximum baud rate 4M, it is suggested to use specific clock frequency(48M Hz) for UART engine clock.

TX_FIFO
The transmitter is buffered with a 16 byte FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO
The receiver is buffered with a 16 byte FIFO (plus three error bits per byte) to reduce the number of interrupts presented to the CPU.

TX shift Register
Shifts the transmit data out serially

RX shift Register
Shifts the receive data in serially

Modem Control Register
This register controls the interface to the MODEM or data set (or a peripheral device emulating a MODEM).

Baud Rate Generator
Divides the UART_CLK clock by the divisor to get the desired baud rate clock. Refer to Table 5.14-1 for the baud rate equation.

Control and Status Register

This is a register set, including the FIFO control registers (FIFO), FIFO status registers (FIFOSTS), and line control register (LINE) for transmitter and receiver. The time out register (TOUT) identifies the condition of time out interrupt. This register set also includes the interrupt enable register (INTEN) and interrupt status register (INTSTS) to enable or disable the responding interrupt and to identify the occurrence of the responding interrupt. There are six types of interrupts, transmitter FIFO empty interrupt(THREINT), receiver threshold level reaching interrupt (RDAINT), line status interrupt (overrun error or parity error or framing error or break interrupt) (RLSINT) , time out interrupt (RXTOINT), MODEM status interrupt (MODEMINT) and Buffer error interrupt (BUFERRINT).

Figure 5.14-3 demonstrates the auto-flow control block diagram.

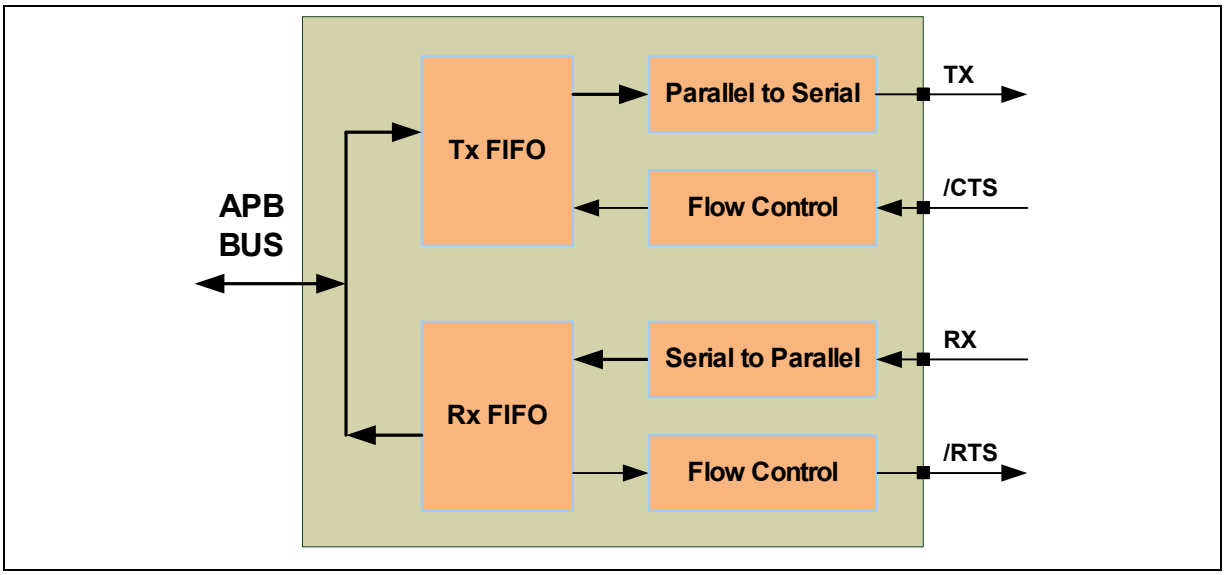


Figure 5.14-3 Auto Flow Control Block Diagram

5.14.5 UART Interface Control Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
UART Base Address: **UARTx_BA= 0x4006_0000 + (0x1000 * x)** **x=0,1**				
UART_DAT	UARTx_BA+0x00	R/W	UART Receive/Transmit FIFO Register.	0x0000_0000
UART_INTEN	UARTx_BA+0x04	R/W	UART Interrupt Enable Register.	0x0000_0000
UART_FIFO	UARTx_BA+0x08	R/W	UART FIFO Control Register.	0x0000_0000
UART_LINE	UARTx_BA+0x0C	R/W	UART Line Control Register.	0x0000_0000
UART_MODEM	UARTx_BA+0x10	R/W	UART Modem Control Register.	0x0000_0000
UART_MODEMSTS	UARTx_BA+0x14	R/W	UART Modem Status Register.	0x0000_0000
UART_FIFOSTS	UARTx_BA+0x18	R/W	UART FIFO Status Register.	0x1040_4000
UART_INTSTS	UARTx_BA+0x1C	R/W	UART Interrupt Status Register.	0x0000_0002
UART_TOUT	UARTx_BA+0x20	R/W	UART Time Out Register	0x0000_0000
UART_BAUD	UARTx_BA+0x24	R/W	UART Baud Rate Divisor Register	0x0F00_0000

5.14.6 UART Interface Control Register Description

UART Receive/Transmit FIFO Data Register (UART_DAT)

Register	Offset	R/W	Description	Reset Value
UART_DAT	UARTx_BA+0x00	R/W	UART Receive/Transmit FIFO Register.	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**DAT**	**Receive/Transmit FIFO Register** Reading this register will return data from the receive data FIFO. By reading this register, the UART will return the 8-bit data received from Rx pin (LSB first). By writing to this register, transmit data will be pushed onto the transmit FIFO. The UART will send out an 8-bit data through the Tx pin (LSB first).

UART Interrupt Enable Register (UART_INTEN)

Register	Offset	R/W	Description	Reset Value
UART_INTEN	UARTx_BA+0x04	R/W	UART Interrupt Enable Register.	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DMARXEN	**DMATXEN**	**ATOCTSEN**	**ATORTSEN**	**TOCNTEN**	**Reserved**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**BUFERRIEN**	**RXTOIEN**	**MODEMIEN**	**RLSIEN**	**THREIEN**	**RDAIEN**

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**DMARXEN**	**Receive DMA Enable** If enabled, the UART will request DMA service when data is available in receive FIFO.
[14]	**DMATXEN**	**Transmit DMA Enable** If enabled, the UART will request DMA service when space is available in transmit FIFO.
[13]	**ATOCTSEN**	**CTS Auto Flow Control Enable** 0 = Disable CTS auto flow control. 1 = Enable CTS auto flow control. When CTS auto-flow is enabled, the UART will send data to external device when CTS input is asserted (UART will not send data to device until CTS is de-asserted).
[12]	**ATORTSEN**	**RTS Auto Flow Control Enable** 0 = Disable RTS auto flow control. 1 = Enable RTS auto flow control. When RTS auto-flow is enabled, if the number of bytes in the Rx FIFO equals FCR.RTS_TRIG_LEVEL, the UART will de-assert the RTS signal.

[11]	**TOCNTEN**	**Time-Out Counter Enable** 0 = Disable Time-out counter. 1 = Enable Time-out counter.
[10:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**BUFERRIEN**	**Buffer Error Interrupt Enable** 0 = Mask off BUF_ERR_INT 1 = Enable IBUF_ERR_INT
[4]	**RXTOIEN**	**Receive Time out Interrupt Enable** 0 = Mask off TOUT_INT 1 = Enable TOUT_INT
[3]	**MODEMIEN**	**Modem Status Interrupt Enable** 0 = Mask off MODEM_INT 1 = Enable MODEM_INT
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable** 0 = Mask off RLS_INT 1 = EnableRLS_INT
[1]	**THREIEN**	**Transmit FIFO Register Empty Interrupt Enable** 0 = Mask off THRE_INT 1 = Enable THRE_INT
[0]	**RDAIEN**	**Receive Data Available Interrupt Enable**. 0 = Mask off RDA_INT 1 = Enable RDA_INT

UART FIFO Control Register (UART_FIFO)

Register	Offset	R/W	Description	Reset Value
UART_FIFO	UARTx_BA+0x08	R/W	UART FIFO Control Register.	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**RTSTRGLV**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RFITL				**Reserved**	**TXRST**	**RXRST**	**Reserved**

Bits	Description	
[31:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[19:16]	**RTSTRGLV**	**RTS Trigger Level for Auto-flow Control** Sets the FIFO trigger level when auto-flow control will de-assert RTS (request-to-send). **RTS_Tri_Lev** / **Trigger Level (Bytes)** 0000 / 1 Byte 0001 / 4 Bytes 0010 / 8 Bytes 0011 / 14 Bytes
[15:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

<table>
<tr><td>[7:4]</td><td>RFITL</td><td>Receive FIFO Interrupt (RDA_INT) Trigger Level
When the number of bytes in the receive FIFO equals the RFITL then the RDA_IF will be set and, if enabled, an RDA_INT interrupt will generated.
<table><tr><th>RFITL</th><th>INTR_RDA Trigger Level (Bytes)</th></tr><tr><td>0000</td><td>1 Byte</td></tr><tr><td>0001</td><td>4 Bytes</td></tr><tr><td>0010</td><td>8 Bytes</td></tr><tr><td>0011</td><td>14 Bytes</td></tr></table></td></tr>
<tr><td>[3]</td><td>Reserved</td><td>Reserved. Any values read should be ignored. When writing to this field always write with reset value.</td></tr>
<tr><td>[2]</td><td>TXRST</td><td>Transmit FIFO Reset
When TFR is set, all the bytes in the transmit FIFO are cleared and transmit internal state machine is reset.
0 = Writing 0 to this bit has no effect.
1 = Writing 1 to this bit will reset the transmit internal state machine and pointers.
Note: This bit will auto-clear after 3 UART engine clock cycles.</td></tr>
<tr><td>[1]</td><td>RXRST</td><td>Receive FIFO Reset
When RFR is set, all the bytes in the receive FIFO are cleared and receive internal state machine is reset.
0 = Writing 0 to this bit has no effect.
1 = Writing 1 to this bit will reset the receive internal state machine and pointers.
Note: This bit will auto-clear after 3 UART engine clock cycles.</td></tr>
<tr><td>[0]</td><td>Reserved</td><td>Reserved. Any values read should be ignored. When writing to this field always write with reset value.</td></tr>
</table>

UART Line Control Register (UART_LINE)

Register	Offset	R/W	Description	Reset Value
UART_LINE	UARTx_BA+0x0C	R/W	UART Line Control Register.	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**BCB**	**SPE**	**EPE**	**PBE**	**NSB**	**WLS**	

Bits	Description	
[31:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**BCB**	**Break Control Bit** When this bit is set to logic 1, the serial data output (Tx) is forced to the 'Space' state (logic 0). Normal condition is serial data output is 'Mark' state. This bit acts only on Tx and has no effect on the transmitter logic.
[5]	**SPE**	**Stick Parity Enable** 0 = Disable stick parity 1 = When bits PBE and SPE are set 'Stick Parity' is enabled. If EPE=0 the parity bit is transmitted and checked as always set, if EPE=1, the parity bit is transmitted and checked as always cleared.

<table>
<tr><td>[4]</td><td>EPE</td><td>Even Parity Enable
0 = Odd number of logic 1’s are transmitted or checked in the data word and parity bits.
1 = Even number of logic 1’s are transmitted or checked in the data word and parity bits.
This bit has effect only when PBE (parity bit enable) is set.</td></tr>
<tr><td>[3]</td><td>PBE</td><td>Parity Bit Enable
0 = Parity bit is not generated (transmit data) or checked (receive data) during transfer.
1 = Parity bit is generated or checked between the "last data word bit" and "stop bit" of the serial data.</td></tr>
<tr><td>[2]</td><td>NSB</td><td>Number of STOP bits
0= One “STOP bit” is generated after the transmitted data
1= Two “STOP bits” are generated when 6-, 7- and 8-bit word length is selected; One and a half “STOP bits” are generated in the transmitted data when 5-bit word length is selected;</td></tr>
<tr><td>[1:0]</td><td>WLS</td><td>Word Length Select
<table>
<tr><th>WLS[1:0]</th><th>Character length</th></tr>
<tr><td>00</td><td>5 bits</td></tr>
<tr><td>01</td><td>6 bits</td></tr>
<tr><td>10</td><td>7 bits</td></tr>
<tr><td>11</td><td>8 bits</td></tr>
</table></td></tr>
</table>

UART MODEM Control Register (UART_MODEM)

Register	Offset	R/W	Description	Reset Value
UART_MODEM	UARTx_BA+0x10	R/W	UART Modem Control Register.	0x0000_0000

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

Bits	Description	
[31:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**RTSSTS**	**RTS Pin State(read only)** This bit is the pin status of RTS.
[12:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**RTSACTLV**	**Request-to-Send (RTS)Active Trigger Level** This bit can change the RTS trigger level. 1=RTS is active low level. 0=RTS is active high level
[8:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**RTS**	**RTS (Request-To-Send) Signal** If IER.ATORTSEN=0, this bit controls whether RTS pin is active or not. 1: Drive RTS inactive (=~RTSACTLV). 0: Drive RTS active (=RTSACTLV).
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

UART Modem Status Register (UART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UART_MODEMSTS	UARTx_BA+0x14	R/W	UART Modem Status Register.	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**CTSACTLV**	**Clear-to-Send (CTS)Active Trigger Level** This bit can change the CTS trigger level. 1= CTS is active low level. 0= CTS is active high level
[7:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**CTSSTS**	**CTS Pin Status (read only)** This bit is the pin status of CTS.
[3:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**CTSDETF**	**Detect CTS State Change Flag** This bit is set whenever CTS input has state change. It will generate Modem interrupt to CPU when IER.MS_IEN=1 **Note:** This bit is cleared by writing 1 to itself.

UART FIFO Status Register (UART_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
UART_FIFOSTS	UARTx_BA+0x18	R/W	UART FIFO Status Register.	0x1040_4000

31	30	29	28	27	26	25	24
Reserved			TXEMPTYF	Reserved			TXOVIF
23	22	21	20	19	18	17	16
TXFULL	TXEMPTY	TXPTR					
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	RXPTR					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	Reserved			RXOVIF

Bits	Description	
[31:29]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[28]	**TXEMPTYF**	**Transmitter Empty (Read Only)** Bit is set by hardware when Tx FIFO is empty and the STOP bit of the last byte has been transmitted. Bit is cleared automatically when Tx FIFO is not empty or the last byte transmission has not completed. **Note:** This bit is read only.
[27:25]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**TXOVIF**	**Tx Overflow Error Interrupt Flag** If the Tx FIFO (UART->DATA) is full, an additional write to UART->DATA will cause an overflow condition and set this bit to logic 1. It will also generate a BUFERRIF event and interrupt if enabled. **Note:** This bit is cleared by writing 1 to itself.
[23]	**TXFULL**	**Transmit FIFO Full(Read Only)** This bit indicates whether the Tx FIFO is full or not. This bit is set when TxFIFO is full; otherwise it is cleared by hardware. TXFULL=0 indicates there is room to write more data to Tx FIFO.

[22]	**TXEMPTY**	**Transmit FIFO Empty(Read Only)** This bit indicates whether the Tx FIFO is empty or not. When the last byte of Tx FIFO has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared after writing data to FIFO (Tx FIFO not empty).
[21:16]	**TXPTR**	**Tx FIFO Pointer (Read Only)** This field returns the Tx FIFO buffer pointer. When CPU writes a byte into the TxFIFO, TXPTR is incremented. When a byte from Tx FIFO is transferred to the Transmit Shift Register, TXPTR is decremented.
[15]	**RXFULL**	**Receive FIFO Full(Read Only)** This bit indicates whether the Rx FIFO is full or not. This bit is set when RxFIFO is full; otherwise it is cleared by hardware.
[14]	**RXEMPTY**	**Receive FIFO Empty(Read Only)** This bit indicates whether the Rx FIFO is empty or not. When the last byte of Rx FIFO has been read by CPU, hardware sets this bit high. It will be cleared when UART receives any new data.
[13:8]	**RXPTR**	**Rx FIFO pointer (Read Only)** This field returns the Rx FIFO buffer pointer. It is the number of bytes available for read in the Rx FIFO. When UART receives one byte from external device, RXPTR is incremented. When one byte of Rx FIFO is read by CPU, RXPTR is decremented.
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**BIF**	**Break Interrupt Flag** This bit is set to a logic 1 whenever the receive data input(Rx) is held in the "space" state (logic 0) for longer than a full word transmission time (that is, the total time of start bit + data bits + parity + stop bits). It is reset whenever the CPU writes 1 to this bit.
[5]	**FEF**	**Framing Error Flag** This bit is set to logic 1 whenever the received character does not have a valid "stop bit" (that is, the stop bit following the last data bit or parity bit is detected as a logic 0),and is reset whenever the CPU writes 1 to this bit.
[4]	**PEF**	**Parity Error Flag** This bit is set to logic 1 whenever the received character does not have a valid "parity bit", and is reset whenever the CPU writes 1 to this bit.
[3:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[0]	**RXOVIF**	**Rx Overflow Error Interrupt Flag** If the Rx FIFO (UART->DATA) is full, and an additional byte is received by the UART, an overflow condition will occur and set this bit to logic 1. It will also generate a BUFERRIF event and interrupt if enabled. **Note:** This bit is cleared by writing 1 to itself.

UART Interrupt Status Register (UART_INTSTS)

Register	Offset	R/W	Description	Reset Value
UART_INTSTS	UARTx_BA+0x1C	R/W	UART Interrupt Status Register.	0x0000_0002

31	30	29	28	27	26	25	24
Reserved		**DBERRINT**	**DRXTOINT**	**DMODINT**	**DRLSINT**	**Reserved**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**DBERRIF**	**DRXTOIF**	**DMODIF**	**DRLSIF**	**Reserved**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**BUFERRINT**	**RXTOINT**	**MODEMINT**	**RLSINT**	**THREINT**	**RDAINT**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**BUFERRIF**	**RXTOIF**	**MODEMIF**	**RLSIF**	**THREIF**	**RDAIF**

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29]	**DBERRINT**	**DMA MODE Buffer Error Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DBERRIF.
[28]	**DRXTOINT**	**DMA MODE Time Out Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DRXTOIF.
[27]	**DMODINT**	**DMA MODE MODEM Status Interrupt Indicator to Interrupt** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DMODENIF.
[26]	**DRLSINT**	**DMA MODE Receive Line Status Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DRLSIF.
[25:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[21]	**DBERRIF**	**DMA MODE Buffer Error Interrupt Flag (Read Only)** This bit is set when either the Tx or Rx FIFO overflows (UART_FIFOSTS.TXOVIF or UART_FIFOSTS.RXOVIF is set). When BUFERRIF is set, the serial transfer may be corrupted. If UART_INTEN.BUFERRIEN is enabled a CPU interrupt request will be generated. NOTE: This bit is cleared when both UART_FIFOSTS.TXOVIF and UART_FIFOSTS.RXOVIF are cleared.
[20]	**DRXTOIF**	**DMA MODE Time Out Interrupt Flag (Read Only)** This bit is set when the Rx FIFO is not empty and no activity occurs in the Rx FIFO and the time out counter equal to TOIC. If UART_INTEN.RXTOIEN is enabled a CPU interrupt request will be generated. NOTE: This bit is read only and user can read FIFO to clear it.
[19]	**DMODIF**	**DMA MODE MODEM Interrupt Flag (Read Only)** This bit is set when the CTS pin has changed state (UART_MODEMSTS.DCTSF =1). If UART_INTEN.MODEMIEN is enabled, a CPU interrupt request will be generated. NOTE: This bit is read only and reset when bit UART_MODEMSTS.DCTSF is cleared by a write 1.
[18]	**DRLSIF**	**DMA MODE Receive Line Status Interrupt Flag (Read Only)** This bit is set when the Rx receive data has a parity, framing or break error (at least one of, UART_FIFOSTS.BIF, UART_FIFOSTS.FEF and UART_FIFOSTS.PEF, is set). If UART_INTEN.RLSIEN is enabled, the RLS interrupt will be generated. NOTE: This bit is read only and reset to 0 when all bits of BIF, FEF and PEF are cleared.
[17:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**BUFERRINT**	**Buffer Error Interrupt Indicator to Interrupt Controller** Logical AND of IER.BUFERRIEN and BUFERRIF
[12]	**RXTOINT**	**Time Out Interrupt Indicator to Interrupt Controller** Logical AND of IER.RXTOIEN and RXTOIF
[11]	**MODEMINT**	**MODEM Status Interrupt Indicator to Interrupt** Logical AND of IER.MSIEN and MODEMIF
[10]	**RLSINT**	**Receive Line Status Interrupt Indicator to Interrupt Controller** Logical AND of IER.RLSIEN and RLSIF
[9]	**THREINT**	**Transmit Holding Register Empty Interrupt Indicator to Interrupt Controller** Logical AND of IER.THREIEN and THREIF

[8]	**RDAINT**	**Receive Data Available Interrupt Indicator to Interrupt Controller** Logical AND of IER.RDAIEN and RDAIF
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**BUFERRIF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when either the Tx or Rx FIFO overflows (FSR.TXOVIF or FSR.RXOVIF is set). When BUFERRIF is set, the serial transfer may be corrupted. If IER.BUFERRIEN is enabled a CPU interrupt request will be generated. **Note:** This bit is cleared when both FSR.TXOVIF and FSR.RXOVIF are cleared.
[4]	**RXTOIF**	**Time Out Interrupt Flag (Read Only)** This bit is set when the Rx FIFO is not empty and no activity occurs in the Rx FIFO and the time out counter equal to TOIC. If IER.RXTOIEN is enabled a CPU interrupt request will be generated. **Note:** This bit is read only and user can read FIFO to clear it.
[3]	**MODEMIF**	**MODEM Interrupt Flag (Read Only)** This bit is set when the CTS pin has changed state (MSR.DCTSF=1). If IER.MS_IEN is enabled, a CPU interrupt request will be generated. **Note:** This bit is read only and reset when bit MSR.DCTSF is cleared by a write 1.
[2]	**RLSIF**	**Receive Line Status Interrupt Flag (Read Only).** This bit is set when the Rx receive data has a parity, framing or break error (at least one of, FSR.BIF, FSR.FEF and FSR.PEF, is set). If IER.RLS_IEN is enabled, the RLS interrupt will be generated. **Note:** This bit is read only and reset to 0 when all bits of BIF, FEF and PEF are cleared.
[1]	**THREIF**	**Transmit Holding Register Empty Interrupt Flag (Read Only).** This bit is set when the last data of Tx FIFO is transferred to Transmitter Shift Register. If IER.THRE_IEN is enabled, the THRE interrupt will be generated. **Note:** This bit is read only and it will be cleared when writing data into the Tx FIFO.
[0]	**RDAIF**	**Receive Data Available Interrupt Flag (Read Only).** When the number of bytes in the Rx FIFO equals FCR.RFITL then the RDA_IF will be set. If IER.RDA_IEN is enabled, the RDA interrupt will be generated. **Note:** This bit is read only and it will be cleared when the number of unread bytes of Rx FIFO drops below the threshold level (RFITL).

Table 5.14-3 UART Interrupt Sources and Flags Table In Software Mode

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator to Interrupt Controller	Interrupt Flag	Flag Cleared by
Buffer Error Interrupt BUFERRINT	BUFERRIEN	BUFERRINT	BUFERRIF = (TXOVIF or RXOVIF)	Write '1' to TXOVIF/RXOVIF
Rx Timeout Interrupt RXTOINT	RXTOIEN	RXTOINT	TOUT_IF	Read data FIFO
Modem Status Interrupt MODEMINT	MSIEN	MODEMINT	MODEMIF = (DCTSF)	Write '1' to DCTSF
Receive Line Status Interrupt RLSINT	RLSIEN	RLSINT	RLSIF = (BIF or FEF or PEF)	Write '1' to BIF/FEF/PEF
Transmit Holding Register Empty Interrupt THREINT	THREIEN	THREINT	THREIF	Write data FIFO
Receive Data Available Interrupt RDAINT	RDAIEN	RDAINT	RDAIF	Read data FIFO

Table 5.14-4 UART Interrupt Sources and Flags Table In DMA Mode

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator To Interrupt Controller	Interrupt Flag	Flag Cleared By
Buffer Error Interrupt BUFERRINT	BUFERRIEN	DBERRINT	DBERRIF = (TXOVIF or RXOVIF)	Write '1' to TXOVIF/ RXOVIF
Rx Timeout Interrupt RXTOINT	RXTOIEN	DRXTOINT	DRXTOIF	Read data FIFO
Modem Status Interrupt MODEMINT	MODEMIEN	DMODINT	DMODIF = (DCTSF)	Write '1' to DCTSF
Receive Line Status Interrupt RLSINT	RLSIEN	DRLSINT	DRLSIF = (BIF or FEF or PEF)	Write '1' to BIF/FEF/PEF

UART Time Out Register (UART_TOUT)

Register	Offset	R/W	Description	Reset Value
UART_TOUT	UARTx_BA+0x20	R/W	UART Time Out Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6:0]	**TOIC**	**Time Out Interrupt Comparator** The time out counter resets and starts counting whenever the Rx FIFO receives a new data word. Once the content of time out counter (TOIC) is equal to that of time out interrupt comparator (TOIC), a receiver time out interrupt (RXTOINT) is generated if IER.RXTOIEN is set. A new incoming data word or RX FIFO empty clears RXTOIF. The period of the time out counter is the baud rate.

UART Baud Rate Divider Register (UART_BAUD)

Register	Offset	R/W	Description	Reset Value
UART_BAUD	UARTx_BA+0x24	R/W	UART Baud Rate Divisor Register	0x0F00_0000

The baud rate generator takes the UART master clock UART_CLK and divides it to produce the baud rate (bit rate) clock. The divider has two division stages controlled by BRD and EDIVM1 fields. These are configured in three modes depending on the selections of BAUDM1 and BAUDM0. These modes and the baud rate equations for them are described in Table 5.14-5

31	30	29	28	27	26	25	24
Reserved		**BAUDM1**	**BAUDM0**	**EDIVM1**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BRD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BRD							

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29]	**BAUDM1**	**BAUD Rate Mode Selection Bit 1** The baud rate equation is: **Baud Rate = UART_CLK / [M * (BRD + 2)]** ; The default value of M is 16. 0 = Disable divider X (M = 16) 1 = Enable divider X (M = **EDIVM1**+1, with **EDIVM1** ≥8). Refer to Table 5.14-5 for more information.
[28]	**BAUDM0**	**BAUD Rate Mode Selection Bit 0** 0:M = **EDIVM1**+1, with restriction **EDIVM1** ≥8. 1: M = 1, with restriction BRD[15:0]≥ 3. Refer to Table 5.14-5 for more information.
[27:24]	**EDIVM1**	**Extra Divider for BAUD Rate Mode 1** The baud rate divider M = **EDIVM1**+1.
[23:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[15:0]	**BRD**	**Baud Rate Divider.** Refer to Table 5.14-5 for more information.

Table 5.14-5 Baud Rate Equations

Mode	BAUDM1	BAUDM0	EDIVM1[3:0]	BRD[15:0]	Baud rate equation
0	0	0	Don't care	A	UART_CLK / [16 * (A+2)]
1	1	0	B	A	UART_CLK / [(B+1) * (A+2)] , requires B≥ 8
2	1	1	Don't care	A	UART_CLK / (A+2), requires A ≥ 10

5.15 Flash Memory Controller (FMC)

5.15.1 Overview

The ISD91500 series is available with 64kbytes of on-chip embedded Flash EEPROM for application program and data flash memory. The memory can be updated through procedures for In-Circuit Programming (ICP) through the ARM Serial-Wire Debug (SWD) port or via In-System Programming (ISP) functions under software control. In-System Programming (ISP) functions let user to update program memory when chip is soldered onto PCB.

Main flash memory is divided into two partitions: Application Program ROM (APROM) and Data flash (DATAF). In addition there are two other partitions, a 6K Byte Boot Loader ROM (LDROM) and Configuration ROM (CONFIG).

Upon chip power-on, the Cortex-M0 CPU fetches code from APROM or LDROM determined by a boot select configuration in CONFIG.

The boundary between APROM and user DATA Flash can be configured to any sector address boundary. Erasable sector size is 512 Byte. This boundary is also specified in the CONFIG memory.

5.15.2 Features

- AHB interface compatible
- Running up to 49 MHz with one wait state and 24.5 MHz without wait state for discontinuous address read access.
- Mini-cache(8 x 32 bit) to reduce flash access and power consumption.
- 64KB application program memory (APROM)
- 6KB in system programming (ISP) boot loader program memory (LDROM)
- Configurable data flash with 512 Bytes sector erase unit
- Programmable data flash start address.
- In System Program (ISP) capability to update on chip Flash EEPROM

5.15.3 Flash Memory Controller Block Diagram

The flash memory controller consist of AHB slave interface, ISP control logic, writer interface and flash macro interface timing control logic. The block diagram of flash memory controller is shown as following:

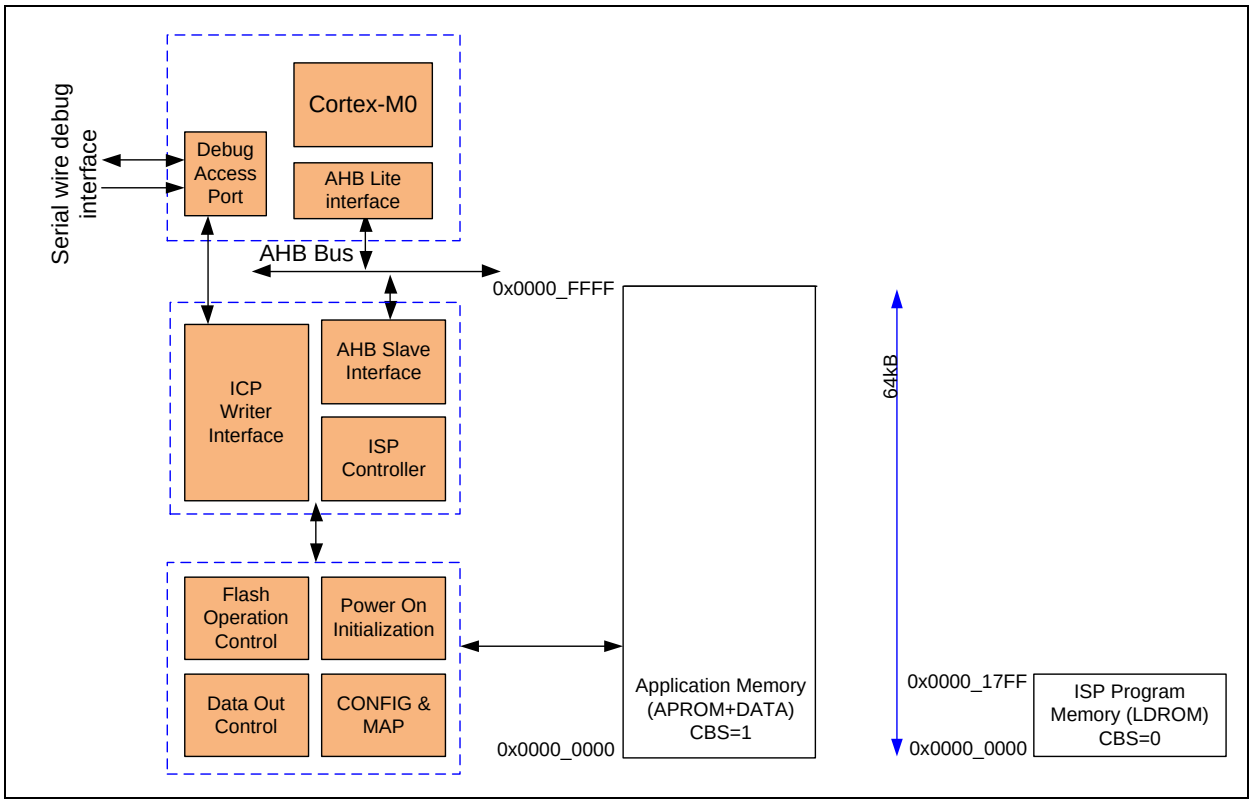


Figure 5.15-1 Flash Memory Control Block Diagram

5.15.4 Flash Memory Organization

The ISD91500 flash memory consists of Application Program (APROM) memory (64KB), data flash (DATAF), ISP boot loader (LDROM) program memory (6KB), user configuration (CONFIG). User configuration block provides 2 words that control system configuration, like flash security lock, boot select, brown out voltage level and data flash base address. An additional 504Bytes are available in CONFIG memory for the user to store custom configuration data. The first two CONFIG words are loaded from CONFIG memory at power-on into device control registers to initialize certain chip functions. The data flash start address (DFBADR) is defined in CONFIG memory and determines the relative size of the APROM and DATAF partitions.

Table 5.15-1 Memory Address Map

Block Name	Size	Start Address	End Address
APROM	64KB	0x0000_0000	0x0000_FFFF (64KB) OR DFBADR-1 if DFEN=0
DATAF	User Configurable	DFBADR	0x0000_FFFF (64KB)
LDROM	6 kB	0x0010_0000	0x0010_17FF
CONFIG	512B	0x0030_0000	0x0030_01FF

The Flash memory organization is shown as below:

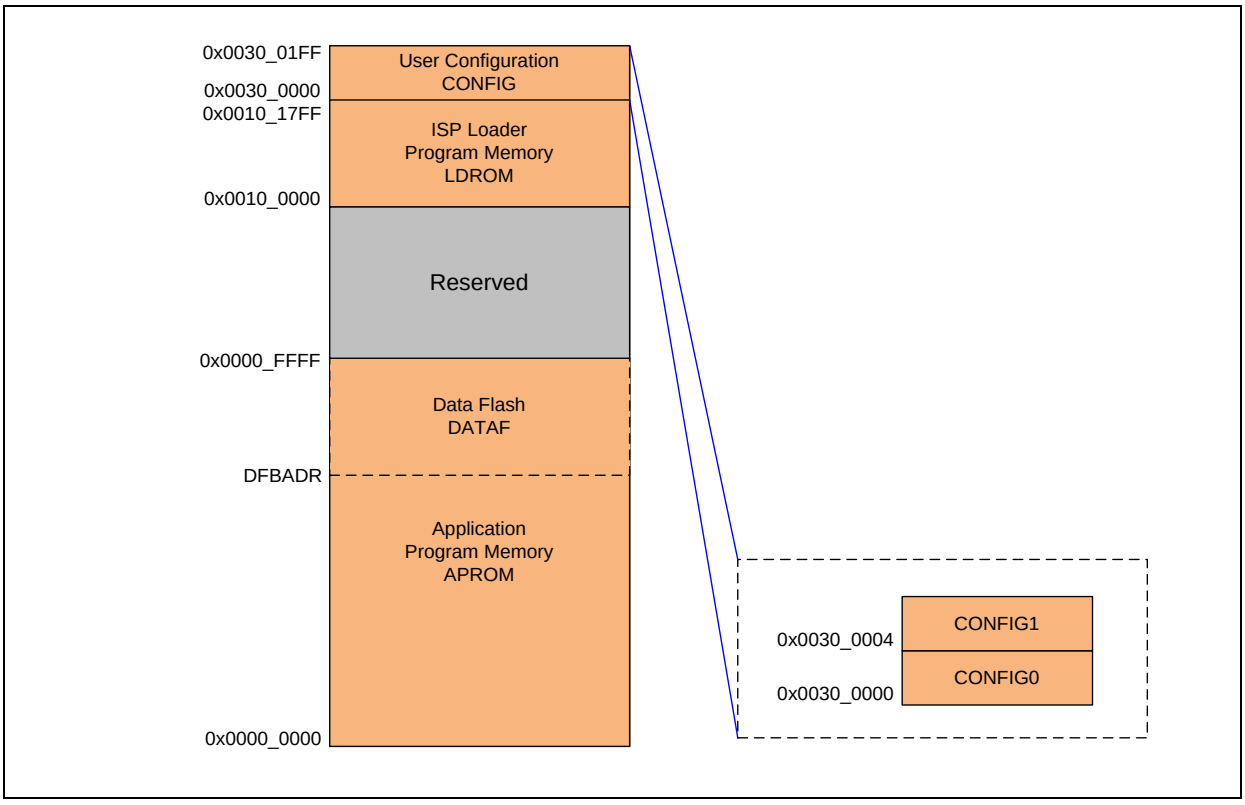


Figure 5.15-2 Flash Memory Organization

5.15.5 Boot Selection

The ISD91500 provides an in-system programming (ISP) feature to enable user to update the application program memory when the chip is mounted on a PCB. A dedicated 6KB boot loader program memory is used to store ISP firmware. The user customizes this firmware to implement a protocol specific to their system to download updated application code. This firmware could utilize device peripherals such as UART, SPI to fetch new application code. The memory area from which the ISD91500 boots is controlled by the CBS bit in Config0 register.

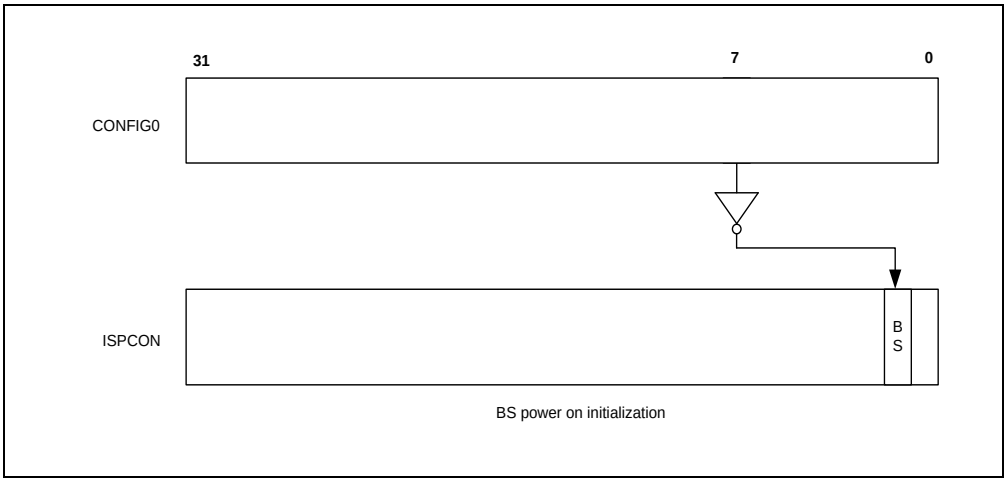


Figure 5.15-3 Boot Select (BS) for Power-on Action

Since CBS in CONFIG0 is used to control system memory map after booting. When set CBS = 1 to boot from APROM, the application in APROM will not be able to access LDROM by CPU read. In other words, when CBS = 0 are set to boot from LDROM, the software executed in LDROM will not be able to access APROM by memory read. The following figure shows the memory map when booting from APROM and LDROM.

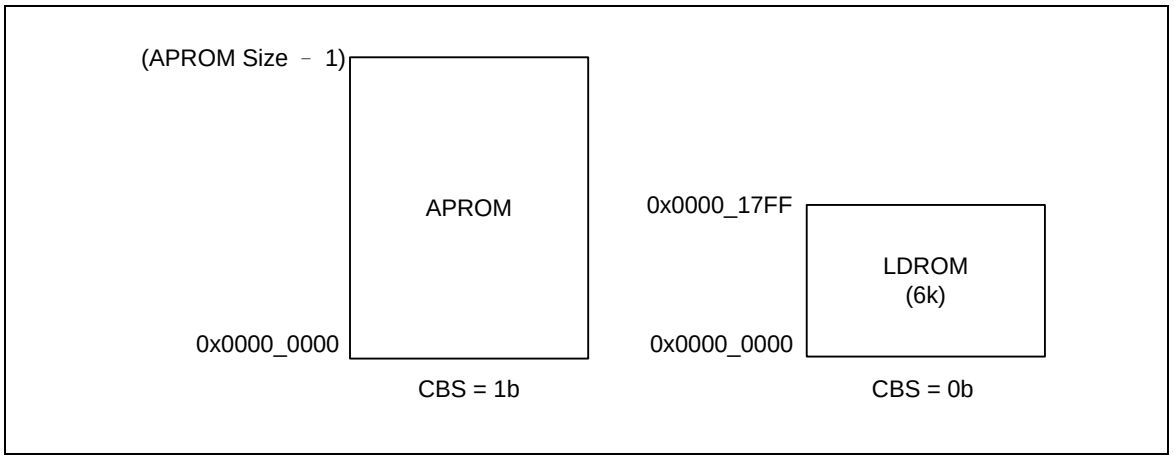


Figure 5.15-4 Program Executing Range for boot from APROM and boot from LDROM

5.15.6 Data Flash (DATAF)

The ISD91500 provides Data Flash for user to store data which is read/write thru ISP registers. The erase unit is 512 bytes. When a word will be changed, all 128 words need to be copied to another page or SRAM in advance. The data flash base address is defined by DFBA if DFEN bit in Config0 is enabled. For example for 4k/2k/1k/0kB data flash, the DFBA setting value is listed in the following table.

Table 5.15-2 Data Flash Table

Data Flash / APROM	4kB (DFEN=0)	2kB (DFEN=0)	1kB (DFEN=0)	0kB (DFEN=1)
64k Flash	DFBA=0x0000_F000	DFBA=0x0000_F800	DFBA=0x0000_FC00	DFEN=1

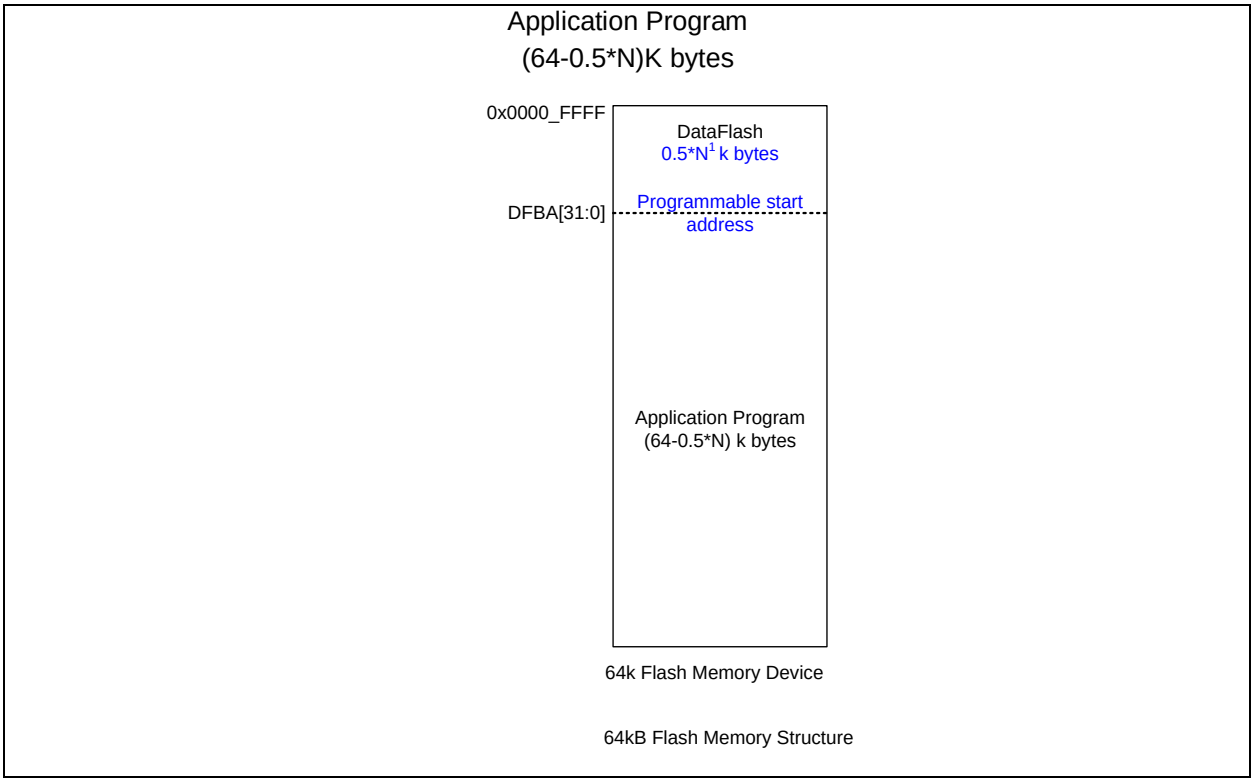


Figure 5.15-5 Flash Memory Structure

5.15.7 User Configuration (CONFIG)

Config0 (Address = 0x0030_0000)

31	30	29	28	27	26	25	24
Reserved				CLVR	Reserved	CBOV	
23	22	21	20	19	18	17	16
CBOV		CBORST	CBODEN	Reserved			
15	14	13	12	11	10	9	8
Reserved						RST_DEB	
7	6	5	4	3	2	1	0
CBS	Reserved					LOCK	DFEN

Config0	Address = 0x0030_0000	
Bits	**Description**	
[31:28]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[27]	**CLVR**	**Config Low Voltage Reset Enable.** 0: LVR Reset enabled after configuration 1: LVR Reset disabled after configuration.
[26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Config0	Address = 0x0030_0000	
Bits	**Description**	
[25:22]	**CBOV**	**Config BOD Voltage** Sets the BOD detect voltage after configuration. CBOV[3:0] =1111: 3.4V CBOV[3:0] =1110: 3.4V CBOV[3:0] =1101: 3.4V CBOV[3:0] =1100: 3.4V CBOV[3:0] =1011: 3.4V CBOV[3:0] =1010: 3.4V CBOV[3:0] =1001: 3.1V CBOV[3:0] =1000: 3.0V CBOV[3:0] =0111: 2.8V CBOV[3:0] =0110: 2.6V CBOV[3:0] =0101: 2.4V CBOV[3:0] =0100: 2.2V CBOV[3:0] =0011: 2.1V CBOV[3:0] =0010: 2.0V CBOV[3:0] =0001: 1.9V CBOV[3:0] =0000: 1.8V
[21]	**CBORST**	**Config BOD Reset** 0: BOD reset enabled after configuration 1: BOD Reset disabled after configuration.
[20]	**CBODEN**	**Config BOD Enable.** 0: BOD function enabled after configuration 1: BOD disabled after configuration.
[19:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9:8]	**RST_DEB**	**Reset debounce time select** RST_DEB [1:0]=00: 7.5 us RST_DEB [1:0]=01: 15 us RST_DEB [1:0]=10: 61us RST_DEB [1:0]=11: 122 us

Config0	Address = 0x0030_0000	
Bits	**Description**	
[7]	**CBS**	**Chip Boot Selection** 0 = LDROM 1 = APROM.
[6:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**LOCK**	**Security Lock** 0 = Flash data locked. 1 = Flash data unlocked. When flash data is locked, only device ID, user configuration can be read by writer and ICP through serial debug interface. Others data is locked as 0xFFFFFFFF. ISP can read data anywhere regardless of LOCK bit value. Only method to unlock chip is a CHIP_ERASE function from ICP which will erase APROM, LDROM and CONFIG.
[0]	**DFEN**	**Data Flash Enabled** 0 = Data Flash Enabled. 1 = Data Flash Disabled.

Note: The reserved bits of user configuration should be kept as '1'.

Config1 (Address = 0x0030_0004)

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

Config1	Address = 0x0030_0004	
Bits	**Descriptions**	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**DFBADR**	**Data Flash Base Address** The Data Flash base address is defined by user. Since on chip flash erase unit is 512 bytes, it is mandatory to keep bit 8-0 as 0.

5.15.8 In-System Programming (ISP)

The program and data flash memory support both in hardware In-Circuit Programming (ICP) and firmware based In-System programming (ISP). Hardware ICP programming mode uses the Serial-Wire Debug (SWD) port to program chip. Dedicated ICE Debug hardware or ICP gang-writers are available to reduce programming and manufacturing costs. For firmware updates in the field, the ISD91500 provides an ISP mode allowing a device to be reprogrammed under software control.

ISP is performed without removing the device from the system. Various interfaces enable LDROM firmware to fetch new program code from an external source. A common method to perform ISP would be via a UART controlled by firmware in LDROM. In this scenario, a PC could transfer new APROM code through a serial port. The LDROM firmware receives it and re-programs APROM through ISP commands. An alternative might be to fetch new firmware from an attached SD-Card via the SPI interface.

5.15.9 ISP Procedure

The ISD91500 will boot from APROM or LDROM from a power-on reset as defined by user configuration bit CBS. If user desires to update application program in APROM, the ISPCON.BS can be set to '1' and a CPU reset(CPURST) issued. This will cause the chip to boot from LDROM. An example flow diagram of the ISP sequence is shown in Figure 5.15-7.

The ISPCON register is a protected register, user must first follow the unlock sequence (see Register Lock Control Register (SYS_REGLCTL)) to gain access. This procedure is to protect the flash memory from unintentional access.

To enable ISP functionality software must first ensure the ISP clock (AHBCLK.ISP_EN) is present then set the ISPCON.ISPEN bit.

Several error conditions (include BOD event) are checked after software writes the ISPTRG register. If an error condition occurs, ISP operation is not started and the ISP fail flag (ISPCON.ISPFF) will be set instead. The ISPFF flag will remain set until it is cleared by software. Subsequent ISP procedure can be started even if ISPFF is set. It is recommended that software check ISPFF bit and clear it after each ISP operation if set.

When ISPTRG register is set, the CoretxM0 CPU will wait for ISP operation to finish, during this period; peripherals operate as usual. If any interrupt requests occur, CPU will not service them until ISP operation finishes. As the ISP functions affect the operation of the flash memory M0 instruction pipeline should be flushed with an ISB (Instruction Synchronization Barrier) instruction after the ISP is triggered.

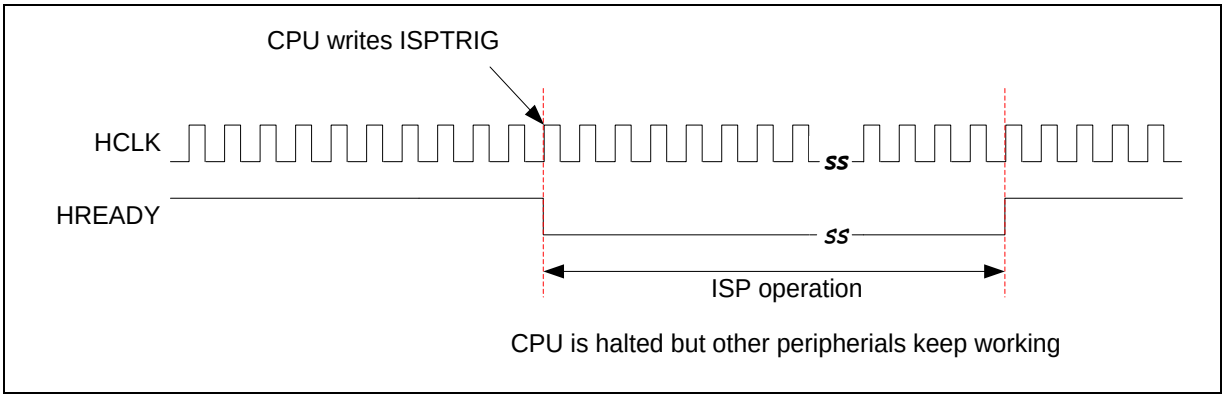


Figure 5.15-6 ISP Operation Timing

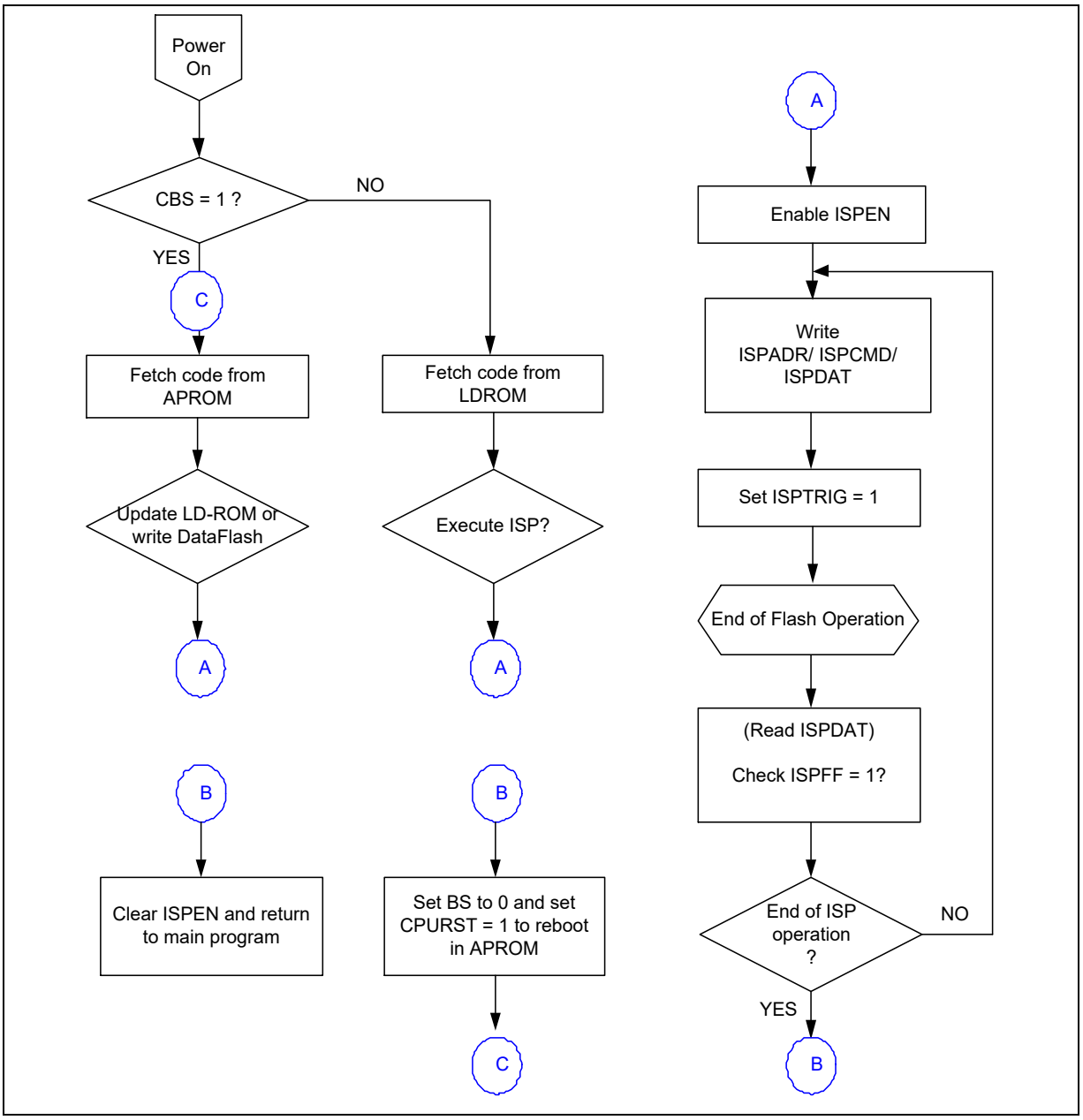


Figure 5.15-7 Boot Sequence and ISP Procedure

The ISP command set is shown in Table 5.15-3. Three registers determine the action of a command: ISPCMD is the command register and accepts commands for reading ID registers and read/write/erase of flash memory. The ISPADR is the address register where the flash memory address for access is written. ISPDAT is the data register that input data is written to and return data read from. An ISP command is executed by setting ISPCMD, ISPDAT and ISPADR then writing to the trigger register ISPTRG.

There is an ISP command to read the device ID register. This register returns a code that reports the memory configuration of the ISD91500 series part as given in Figure 5.15-7.

Table 5.15-3 ISP Command Set

ISP Mode	ISPCMD	ISPADR			ISPDAT
	ISPCMD[5:0]	A21	A20	A[19:0]	D[31:0]
Read Company ID	0x0B	x	x	x	Returns 0x0000_00DA
Read Device ID	0x0C	x	x	0x00000	
FLASH Page Erase	0x22	0	A[20]	A[19:0]	x
FLASH Program	0x21	0	A[20]	A[19:0]	Data input
FLASH Read	0x00	0	A[20]	A[19:0]	Data output
CONFIG Page Erase	0x22	1	1	A[19:0]	x
CONFIG Program	0x21	1	1	A[19:0]	Data input
CONFIG Read	0x00	1	1	A[19:0]	Data output

5.15.10 Flash Control Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
FMC Base Address **FMC_BA = 0x5000_C000**				
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0002_0000
FMC_ISPADR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000
FMC_DFBADR	FMC_BA+0x14	R	Data Flash Base Address	0xXXXX_XXXX

5.15.11 Flash Control Register Description

ISP Control Register (FMC_ISPCTL)

The ISPCON register is a protected register, user must first follow the unlock sequence (see Register Lock Control Register (SYS_REGLCTL)) to gain access.

Register	Offset	R/W	Description	Reset Value
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0002_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		CACHE_DIS	Reserved		WAIT_CFG		
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	ISPFF	LDUEN	CFGUEN	APUWEN	Reserved	BS	ISPEN

Bits	Description	
[31:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**CACHE_DIS**	**Cache Disable** When set to 1, caching of flash memory reads is disabled. **Note:** Cache possibly becomes wrong if there is FMC erase/write command. 8 different address reading on known code area (not erase/write address) can make sure cache correct again.
[20:19]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[18:16]	**WAIT_CFG**	**Flash Access Wait State Configuration** 0x3= Zero wait states. HCLK < 24 MHz 0x2= One wait states. HCLK <= 50 MHz 0x1= Two wait state. 0x0= Three wait state. Before changing WAIT_CFG, ensure HCLK speed is < 25 MHz.
[15:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[6]	**ISPFF**	**ISP Fail Flag** This bit is set by hardware when a triggered ISP meets any of the following conditions: (1) APROM writes to itself. (2) LDROM writes to itself. (3) Destination address is illegal, such as over an available range. (4) BOD event happen Write 1 to clear.
[5]	**LDUEN**	**LDROM Update Enable** LDROM update enable bit. 0 = LDROM cannot be updated 1 = LDROM can be updated when the MCU runs in APROM.
[4]	**CFGUEN**	**CONFIG Update Enable** 0 = Disable 1 = Enable When enabled, ISP functions can access the CONFIG address space and modify device configuration area.
[3]	**APUWEN**	**APU Write Enable** 0 = APROM can't write itself. ISPFF with "1" 1 = APROM write to itself.
[2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**BS**	**Boot Select** 0: APROM 1: LDROM This bit functions as MCU boot status flag, which can be used to check where MCU booted from. **This bit is initialized after power-on reset with the inverse of CBS in Config0; It is not reset for any other reset event.**
[0]	**ISPEN**	**ISP Enable** 0 = Disable ISP function 1 = Enable ISP function

ISP Address Register (FMC_ISPADR)

Register	Offset	R/W	Description	Reset Value
FMC_ISPADR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**ISPADR**	**ISP Address Register** This is the memory address register that a subsequent ISP command will access. ISP operation are carried out on 32bit words only, consequently ISPARD [1:0] must be 00b for correct ISP operation.

ISP Data Register (FMC_ISPDAT)

Register	Offset	R/W	Description	Reset Value
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000

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

Bits	Description	
[31:0]	**ISPDAT**	**ISP Data Register** Write data to this register before an ISP program operation. Read data from this register after an ISP read operation

ISP Command (FMC_ISPCMD)

Register	Offset	R/W	Description	Reset Value
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:0]	**ISPCMD**	**ISP Command** Operation Mode / ISPCMD: Read: 0x00 Program: 0x21 Page Erase: 0x22 Read CID: 0x0B Read DID: 0x0C Read UID: 0x04

ISP Trigger Control Register (FMC_ISPTRG)

The ISPTRG register is a protected register, user must first follow the unlock sequence (see Register Lock Control Register (SYS_REGLCTL)) to gain access.

Register	Offset	R/W	Description	Reset Value
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**ISPGO**	**ISP Start Trigger** Write 1 to start ISP operation. This will be cleared to 0 by hardware automatically when ISP operation is finished. 0 = ISP operation is finished 1 = ISP is on going After triggering an ISP function M0 instruction pipeline should be flushed with a ISB instruction to guarantee data integrity. This is a protected register, user must first follow the unlock sequence see Register Lock Control Register (SYS_REGLCTL)) to gain access.

Data Flash Base Address Register (FMC_DFBADR)

Register	Offset	R/W	Description	Reset Value
FMC_DFBADR	FMC_BA+0x14	R	Data Flash Base Address	0xXXXX_XXXX

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

Bits	Description	
[31:0]	DFBA	**Data Flash Base Address** This register reports the data flash starting address. It is a read only register. Data flash size is defined by user configuration, register content is loaded from Config1 when chip is reset.

5.16 USB 1.1 Device Controller (USBD)

5.16.1 Overview

There is one set of USB 2.0 full-speed device controller and transceiver in this device. It is compliant with USB 2.0 full-speed device specification and supports control/bulk/interrupt/isochronous transfer types.

In this device controller, there are two main interfaces: the APB bus and USB bus which comes from the USB PHY transceiver. For the APB bus, the CPU can program control registers through it. There are 1Kbytes internal SRAM as data buffer in this controller. For IN or OUT transfer, it is necessary to write data to SRAM or read data from SRAM through the APB interface or SIE. User needs to set the effective starting address of SRAM for each endpoint buffer through buffer segmentation register (USBD_BUFSEGx).

There are 12 endpoints in this controller. Each of the endpoint can be configured as IN or OUT endpoint. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. The block of “Endpoint Control” is also used to manage the data sequential synchronization, endpoint states, current start address, transaction status, and data buffer status for each endpoint.

There are four different interrupt events in this controller. They are the no-event-wake-up, device plug-in or plug-out event, USB events, like IN ACK, OUT ACK etc, and BUS events, like suspend and resume, etc. Any event will cause an interrupt, and users just need to check the related event flags in interrupt event status register (USBD_INTSTS) to acknowledge what kind of interrupt occurring, and then check the related USB Endpoint Status Register (USBD_EPSTS0 and USBD_EPSTS1) to acknowledge what kind of event occurring in this endpoint.

A software-disconnect function is also supported for this USB controller. It is used to simulate the disconnection of this device from the host. If user enables SE0 bit (USBD_SE0), the USB controller will force the output of USB_D+ and USB_D- to level low and its function is disabled. After disable the SE0 bit, host will enumerate the USB device again.

For more information on the Universal Serial Bus, please refer to *Universal Serial Bus Specification Revision 1.1*.

5.16.2 Features

- Compliant with USB 2.0 Full-Speed specification
- Provides 1 interrupt vector with 4 different interrupt events (NEVWK, VBUSDET, USB and BUS)
- Supports Control/Bulk/Interrupt/Isochronous transfer type
- Supports suspend function when no bus activity existing for 3 ms
- Supports 12 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 1Kbytes buffer size
- Provides remote wake-up capability

5.16.3 Block Diagram

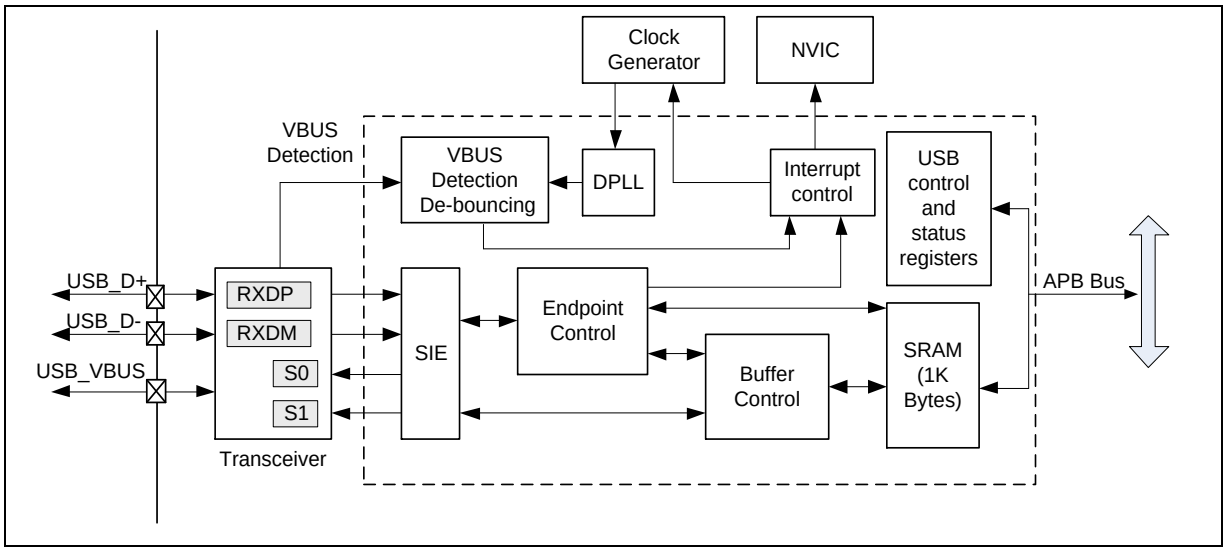


Figure 5.16-1 USB Block Diagram

5.16.4 Basic Configuration

The internal USB 3.3V LDO can be power on by PHYPD (USBD_ATTR[9]). The USBD clock source is derived from PLL or HIRC. User has to set the PLL related configurations before USB device controller is enabled. Set the USBEN (CLK_APBCLK[24]) bit to enable USBD clock and 4-bit pre-scaler USBDIV (CLK_CLKDIV0[15:12]) to generate the proper USBD clock rate.

Please be noted that even I91500 HIRC has SOF auto trim function, but the HIRC source still hard to reach USB frequency +/-0.25% spec. Suggest user to select USB clock source from PLL which source from stable external crystal(HXT) to reach the USB frequency spec.

- Clock source configuration
  - Enable HIRC (CLK_PWRCTL[2]) or set PLL controller (CLK_PLLCTL)
  - Set USBDSEL(CLK_CLKSEL2[0]) to select USB clock source from HIRC or PLL
  - Select the clock divider number of USBD peripheral clock on USBDIV (CLK_CLKDIV0[15:12]).
  - Enable USBD peripheral clock in USBEN (CLK_APBCLK[24]).
- Reset configuration
  - Reset USBD controller in USBRST (SYS_IPRST1[24]).

5.16.5 Functional Description

5.16.5.1 Serial Interface Engine (SIE)

The SIE is the front-end of the device controller and handles most of the USB packet protocol. The SIE typically comprehends signaling up to the transaction level. The functions that it handles could include:

- Packet recognition and transaction sequencing
- SOP, EOP, RESET, RESUME signal detection/generation
- Clock/Data separation
- NRZI Data encoding/decoding and bit-stuffing
- CRC generation and checking (for Token and Data)
- Packet ID (PID) generation and checking/decoding
- Serial-Parallel/Parallel-Serial conversion

5.16.5.2 Endpoint Control

This controller supports 12 endpoints. Each of the endpoint can be configured as Control, Bulk, Interrupt, or Isochronous transfer type. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. It is also used to manage the data sequential synchronization, endpoint state control, current endpoint start address, current transaction status, and data buffer status in each endpoint.

5.16.5.3 Digital Phase Lock Loop (DPLL)

The bit rate of USB data is 12 MHz. The DPLL uses the 48 MHz which comes from the clock controller to lock the input data RXDP and RXDM. The 12 MHz bit rate clock is also converted from DPLL.

5.16.5.4 VBUS Detection De-bouncing

A USB device may be plugged-in or plugged-out from the USB host. To monitor the state of a USB device when it is detached from the USB host, the device controller provides hardware de-bouncing for USB VBUS detection interrupt to avoid bounce problems on USB plug-in or unplug. VBUS detection interrupt appears about 10 ms later than USB plug-in or plug-out. User can acknowledge USB plug-in/plug-out by reading USBD_VBUSDET register. The VBUSDET flag represents the current state on the bus without de-bouncing. If VBUSDET is 1, it means the USB cable is plugged-in. If user polls the flag to check USB state, software de-bouncing must be added if needed.

5.16.5.5 Interrupt control

This USB provides 1 interrupt vector with 4 interrupt events (NEVWK, VBUSDET, USB and BUS). The NEVWK event occurs after waking up the system from Power-down mode (The power mode function is defined in system power-down control register, CLK_PWRCTL). The VBUSDET event is used for USB plug-in or unplug. The USB event notifies users of some USB requests, such as IN ACK, OUT ACK., and the BUS event notifies users of some bus events, such as suspend and, resume. The related bits must be set in the interrupt enable register (USBD_INTEN) of USB Device Controller to enable USB interrupts.

NEVWK interrupt is only presented when no the other USB interrupt events happened more than 20ms after the chip is waked up from Power-down mode. After the chip enters Power-down mode, any change on USB_VBUS, USB_D+ and USB_D- can wake up this chip if USB wake-up function is enabled. If this change is not intentionally, no interrupt but NEVWK interrupt will occur. After waking up by USB, this interrupt will occur when no the other USB interrupt events are presented for more than 20ms. The Figure 5.16-2 is the control flow of wake-up interrupt.

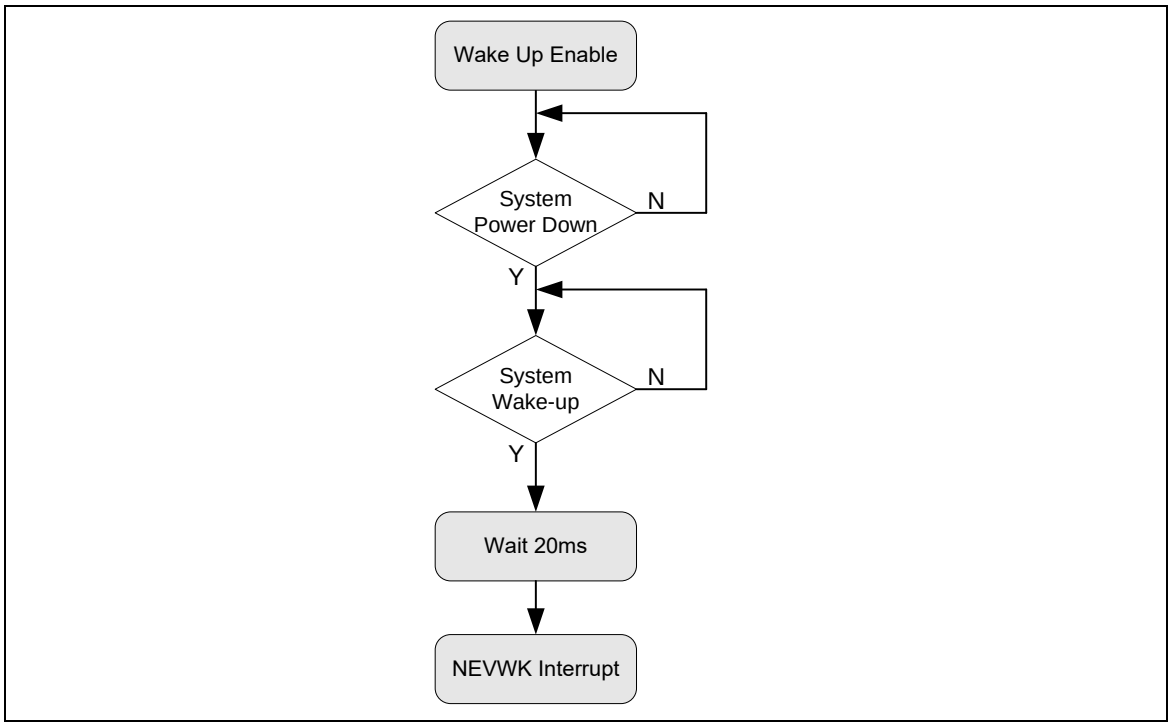


Figure 5.16-2 NEVWK Interrupt Operation Flow

The USB interrupt is used to notify users of any USB event on the bus, and user can read EPSTS (USBD_EPSTS0 and USBD_EPSTS1) and EPEVT11~0 (USBD_INTSTS[27:16]) to take necessary responses.

Same as USB interrupt, BUS interrupt notifies users of some bus events, like USB reset, suspend, time-out, and resume. User can read USBD_ATTR to acknowledge bus events.

5.16.5.6 Power Saving

User can write 0 to USBD_ATTR[4] to disable PHY under special circumstances, like suspend, to conserve power.

5.16.5.7 Buffer Control

There is 1Kbytes SRAM in the controller and the 12 endpoints share this buffer. User shall configure each endpoint's effective starting address in the buffer segmentation register before the USB function active. The "Buffer Control" block is used to control each endpoint's effective starting address and its SRAM size is defined in the USBD_MXPLDx register.

Figure 5.16-3 depicts the starting address for each endpoint according the content of USBD_BUFSEGx and USBD_MXPLDx registers. If the USBD_BUFSEG0 is programmed as 0x08h and USBD_MXPLD0 is set as 0x40h, the SRAM size of endpoint 0 is start from USBD_BA+0x108h and end in USBD_BA+0x148h. (Note: The USBD SRAM base is USBD_BA+0x100h).

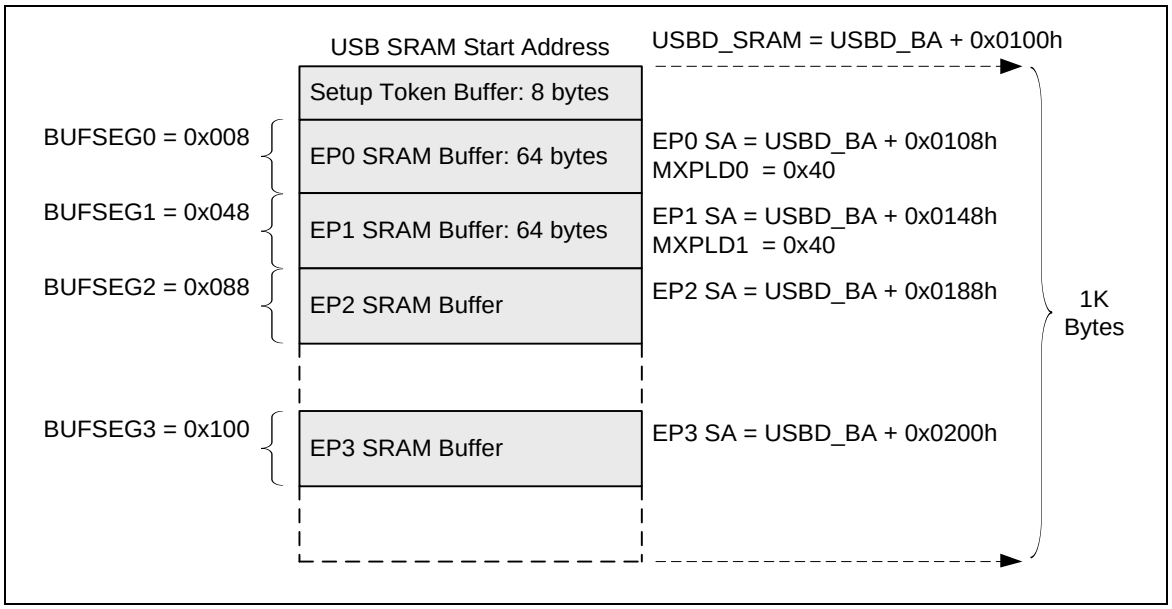


Figure 5.16-3 Endpoint SRAM Structure

5.16.5.8 Handling Transactions with USB Device Peripheral

User can use interrupt or polling USBD_INTSTS to monitor the USB transactions. When transactions occur, USBD_INTSTS will be set by hardware and send an interrupt request to CPU (if related interrupt enabled), or user can polling USBD_INTSTS to get these events without interrupt. The following is the control flow with interrupt enabled.

When USB host has requested data from a device controller, user needs to prepare related data in the specified endpoint buffer in advance. After buffering the required data, user needs to write the actual data length in the specified USBD_MXPLDx register. Once this register is written, the internal signal “In_Rdy” will be asserted and the buffering data will be transmitted immediately after receiving associated IN token from Host. Note that after transferring the specified data, the signal “In_Rdy” will de-assert automatically by hardware.

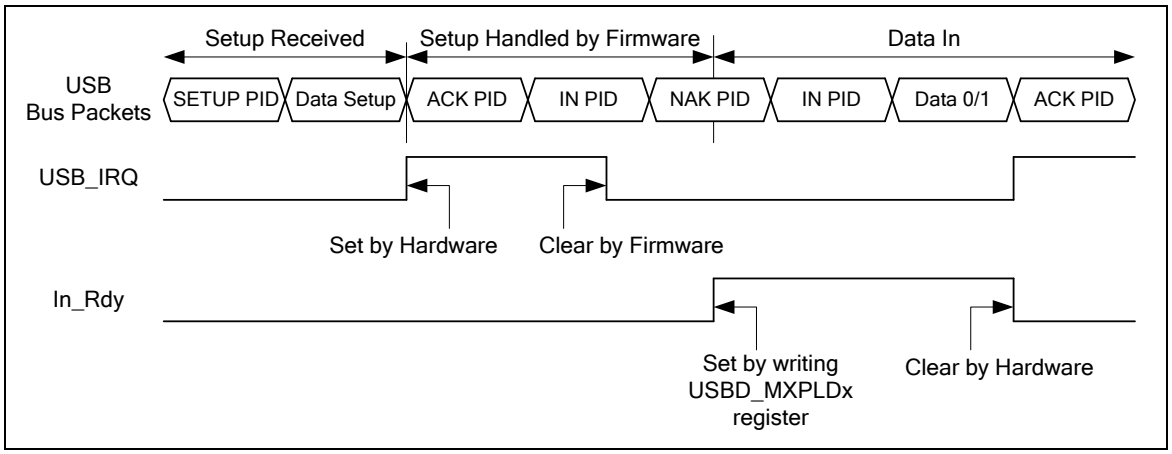


Figure 5.16-4 Setup Transaction Followed by Data IN Transaction

Alternatively, when USB host wants to transmit data to the OUT endpoint in the device controller, hardware will buffer these data to the specified endpoint buffer. After this transaction is completed, hardware will record the

data length in specified USBD_MXPLDx register and de-assert the internal signal “Out_Rdy”. This will avoid hardware accepting next transaction until user moves out the current data in the related endpoint buffer. Once users have processed this transaction, the specified USBD_MXPLDx register needs to be written by firmware to assert the signal “Out_Rdy” again to accept the next transaction.

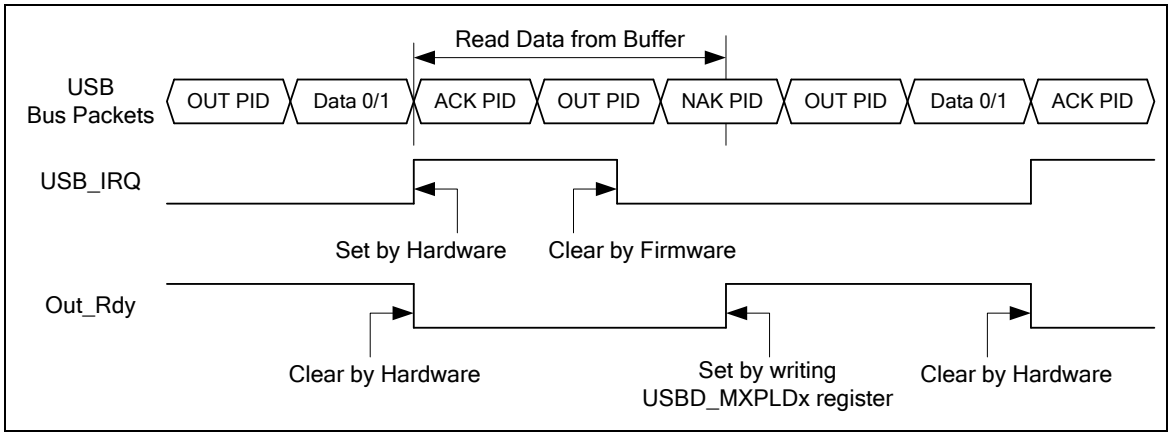


Figure 5.16-5 Data Out Transfer

5.16.6 USB Control Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
USBD Base Address: **USBD_BA = 0x400C_0000**				
USBD_INTEN	USBD_BA+0x000	R/W	USB Device Interrupt Enable Register	0x0000_0000
USBD_INTSTS	USBD_BA+0x004	R/W	USB Device Interrupt Event Status Register	0x0000_0000
USBD_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000
USBD_EPSTS	USBD_BA+0x00C	R	USB Device Endpoint Status Register	0x0000_0000
USBD_ATTR	USBD_BA+0x010	R/W	USB Device Bus Status and Attribution Register	0x0000_0040
USBD_VBUSDET	USBD_BA+0x014	R	USB Device VBUS Detection Register	0x0000_0000
USBD_STBUFSEG	USBD_BA+0x018	R/W	SETUP Token Buffer Segmentation Register	0x0000_0000
USBD_EPSTS0	USBD_BA+0x020	R	USB Device Endpoint Status Register 0	0x0000_0000
USBD_EPSTS1	USBD_BA+0x024	R	USB Device Endpoint Status Register 1	0x0000_0000
USBD_FN	USBD_BA+0x08C	R	USB Frame number Register	0x0000_0XXX
USBD_SE0	USBD_BA+0x090	R/W	USB Device Drive SE0 Control Register	0x0000_0001
USBD_VDDIS	USBD_BA+0x0A8	R/W	USB Device VBUS Detection De-bounce Control Register	0x0000_0000
USBD_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USBD_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USBD_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
USBD Base Address: **USBD_BA = 0x400C_0000**				
USBD_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USBD_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG7	USBD_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD7	USBD_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000
USBD_CFG7	USBD_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000
USBD_CFGP7	USBD_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG8	USBD_BA+0x580	R/W	Endpoint 8 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD8	USBD_BA+0x584	R/W	Endpoint 8 Maximal Payload Register	0x0000_0000
USBD_CFG8	USBD_BA+0x588	R/W	Endpoint 8 Configuration Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
USBD Base Address: **USBD_BA = 0x400C_0000**				
USBD_CFGP8	USBD_BA+0x58C	R/W	Endpoint 8 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG9	USBD_BA+0x590	R/W	Endpoint 9 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD9	USBD_BA+0x594	R/W	Endpoint 9 Maximal Payload Register	0x0000_0000
USBD_CFG9	USBD_BA+0x598	R/W	Endpoint 9 Configuration Register	0x0000_0000
USBD_CFGP9	USBD_BA+0x59C	R/W	Endpoint 9 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG10	USBD_BA+0x5A0	R/W	Endpoint 10 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD10	USBD_BA+0x5A4	R/W	Endpoint 10 Maximal Payload Register	0x0000_0000
USBD_CFG10	USBD_BA+0x5A8	R/W	Endpoint 10 Configuration Register	0x0000_0000
USBD_CFGP10	USBD_BA+0x5AC	R/W	Endpoint 10 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG11	USBD_BA+0x5B0	R/W	Endpoint 11 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD11	USBD_BA+0x5B4	R/W	Endpoint 11 Maximal Payload Register	0x0000_0000
USBD_CFG11	USBD_BA+0x5B8	R/W	Endpoint 11 Configuration Register	0x0000_0000
USBD_CFGP11	USBD_BA+0x5BC	R/W	Endpoint 11 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

Memory Type	Address	Size	Description
USBD_BA = 0x400C_0000			
USBD_SRAM	USBD_BA+0x100 ~ USBD_BA+0x4FF	1024 Bytes	The SRAM is used for the entire endpoints buffer. Refer to section 5.16.5.7 for the endpoint SRAM structure and its description.

5.16.7 USB Control Register Description

USB Interrupt Enable Register (USBD_INTEN)

Register	Offset	R/W	Description	Reset Value
USBD_INTEN	USBD_BA+0x000	R/W	USB Device Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
INNAKEN	Reserved						WKEN
7	6	5	4	3	2	1	0
Reserved			SOFIEN	NEVWKIEN	VBDETIEN	USBIEN	BUSIEN

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**INNAKEN**	**Active NAK Function and Its Status in IN Token** 0 = When device responds NAK after receiving IN token, IN NAK status will not be updated to USBD_EPSTS0 and USBD_EPSTS1, so that the USB interrupt event will not be asserted. 1 = IN NAK status will be updated to USBD_EPSTS0 and USBD_EPSTS1 and the USB interrupt event will be asserted, when the device responds NAK after receiving IN token.
[14:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**WKEN**	**Wake-up Function Enable Bit** 0 = USB wake-up function Disabled. 1 = USB wake-up function Enabled.
[7:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**SOFIEN**	**Start of Frame Interrupt Enable Bit** 0 = SOF Interrupt Disabled. 1 = SOF Interrupt Enabled.

[3]	**NEVWKIEN**	**USB No-event-wake-up Interrupt Enable Bit** 0 = No-event-wake-up Interrupt Disabled. 1 = No-event-wake-up Interrupt Enabled.
[2]	**VBDETIEN**	**VBUS Detection Interrupt Enable Bit** 0 = VBUS detection Interrupt Disabled. 1 = VBUS detection Interrupt Enabled.
[1]	**USBIEN**	**USB Event Interrupt Enable Bit** 0 = USB event interrupt Disabled. 1 = USB event interrupt Enabled.
[0]	**BUSIEN**	**Bus Event Interrupt Enable Bit** 0 = BUS event interrupt Disabled. 1 = BUS event interrupt Enabled.

USB Interrupt Event Status Register (USBD_INTSTS)

Register	Offset	R/W	Description	Reset Value
USBD_INTSTS	USBD_BA+0x004	R/W	USB Device Interrupt Event Status Register	0x0000_0000

31	30	29	28	27	26	25	24
SETUP	Reserved			EPEVT11	EPEVT10	EPEVT9	EPEVT8
23	22	21	20	19	18	17	16
EPEVT7	EPEVT6	EPEVT5	EPEVT4	EPEVT3	EPEVT2	EPEVT1	EPEVT0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			SOFIF	NEVWKIF	VBDETIF	USBIF	BUSIF

Bits	Description	
[31]	**SETUP**	**Setup Event Status** 0 = No Setup event. 1 = Setup event occurred, cleared by writing 1 to USBD_INTSTS[31].
[30:28]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[27]	**EPEVT11**	**Endpoint 11's USB Event Status** 0 = No event occurred in endpoint 11. 1 = USB event occurred on Endpoint 11, check USBD_EPSTS1[15:12] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[27] or USBD_INTSTS[1].
[26]	**EPEVT10**	**Endpoint 10's USB Event Status** 0 = No event occurred in endpoint 10. 1 = USB event occurred on Endpoint 10, check USBD_EPSTS1[11 :8] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[26] or USBD_INTSTS[1].
[25]	**EPEVT9**	**Endpoint 9's USB Event Status** 0 = No event occurred in endpoint 9. 1 = USB event occurred on Endpoint 9, check USBD_EPSTS1[7 :4] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[25] or USBD_INTSTS[1].

[24]	**EPEVT8**	**Endpoint 8's USB Event Status** 0 = No event occurred in endpoint 8. 1 = USB event occurred on Endpoint 8, check USBD_EPSTS1[3 :0] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[24] or USBD_INTSTS[1].
[23]	**EPEVT7**	**Endpoint 7's USB Event Status** 0 = No event occurred in endpoint 7. 1 = USB event occurred on Endpoint 7, check USBD_EPSTS0[31:28] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[23] or USBD_INTSTS[1].
[22]	**EPEVT6**	**Endpoint 6's USB Event Status** 0 = No event occurred in endpoint 6. 1 = USB event occurred on Endpoint 6, check USBD_EPSTS0[27:24] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[22] or USBD_INTSTS[1].
[21]	**EPEVT5**	**Endpoint 5's USB Event Status** 0 = No event occurred in endpoint 5. 1 = USB event occurred on Endpoint 5, check USBD_EPSTS0[23:20] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[21] or USBD_INTSTS[1].
[20]	**EPEVT4**	**Endpoint 4's USB Event Status** 0 = No event occurred in endpoint 4. 1 = USB event occurred on Endpoint 4, check USBD_EPSTS0[19:16] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[20] or USBD_INTSTS[1].
[19]	**EPEVT3**	**Endpoint 3's USB Event Status** 0 = No event occurred in endpoint 3. 1 = USB event occurred on Endpoint 3, check USBD_EPSTS0[15:12] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[19] or USBD_INTSTS[1].
[18]	**EPEVT2**	**Endpoint 2's USB Event Status** 0 = No event occurred in endpoint 2. 1 = USB event occurred on Endpoint 2, check USBD_EPSTS0[11:8] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[18] or USBD_INTSTS[1].
[17]	**EPEVT1**	**Endpoint 1's USB Event Status** 0 = No event occurred in endpoint 1. 1 = USB event occurred on Endpoint 1, check USBD_EPSTS0[7:4] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[17] or USBD_INTSTS[1].

[16]	**EPEVT0**	**Endpoint 0's USB Event Status** 0 = No event occurred in endpoint 0. 1 = USB event occurred on Endpoint 0, check USBD_EPSTS0[3:0] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[16] or USBD_INTSTS[1].
[15:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**SOFIF**	**Start of Frame Interrupt Status** 0 = SOF event does not occur. 1 = SOF event occurred, cleared by write 1 to USBD_INTSTS[4].
[3]	**NEVWKIF**	**No-event-wake-up Interrupt Status** 0 = NEVWK event does not occur. 1 = No-event-wake-up event occurred, cleared by writing 1 to USBD_INTSTS[3].
[2]	**VBDETIF**	**VBUS Detection Interrupt Status** 0 = There is not attached/detached event in the USB. 1 = There is attached/detached event in the USB bus and it is cleared by writing 1 to USBD_INTSTS[2].
[1]	**USBIF**	**USB Event Interrupt Status** The USB event includes the SETUP Token, IN Token, OUT ACK, ISO IN, or ISO OUT events in the bus. 0 = No USB event occurred. 1 = USB event occurred, check EPSTS (USBD_EPSTS0 and USBD_EPSTS1) to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[1] or EPEVT11~0 (USBD_INTSTS[27:16] and SETUP (USBD_INTSTS[31]).
[0]	**BUSIF**	**BUS Interrupt Status** The BUS event means that there is one of the suspend or the resume function in the bus. 0 = No BUS event occurred. 1 = Bus event occurred; check USBD_ATTR[3:0] to know which kind of bus event was occurred, cleared by writing 1 to USBD_INTSTS[0].

USB Device Function Address Register (USBD_FADDR)

A 7-bit value is used as the address of a device on the USB BUS.

Register	Offset	R/W	Description	Reset Value
USBD_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000

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

Bits	Description	
[31:7]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6:0]	FADDR	USB Device Function Address

USB Endpoint Status Register (USBD_EPSTS)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS	USBD_BA+0x00C	R	USB Device Endpoint Status Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**OV**	**Overrun** It indicates that the received data is over the maximum payload number or not. 0 = No overrun. 1 = Out Data is more than the Max Payload in MXPLD register or the Setup Data is more than 8 Bytes.
[6:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

USB Bus Status and Attribution Register (USBD_ATTR)

Register	Offset	R/W	Description	Reset Value
USBD_ATTR	USBD_BA+0x010	R/W	USB Device Bus Status and Attribution Register	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BYTEM	PHYPD	DPPUEN
7	6	5	4	3	2	1	0
USBEN	Reserved	RWAKEUP	PHYEN	TOUT	RESUME	SUSPEND	USBRST

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**BYTEM**	**CPU Access USB SRAM Size Mode Selection** 0 = Word mode: The size of the transfer from CPU to USB SRAM can be Word only. 1 = Byte mode: The size of the transfer from CPU to USB SRAM can be Byte only.
[9]	**PHYPD**	**PHY Power Down Control** 0 = Power Down 1 = Power On
[8]	**DPPUEN**	**Pull-up Resistor on USB_DP Enable Bit** 0 = Pull-up resistor in USB_D+ bus Disabled. 1 = Pull-up resistor in USB_D+ bus Active.
[7]	**USBEN**	**USB Controller Enable Bit** 0 = USB Controller Disabled. 1 = USB Controller Enabled.
[6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**RWAKEUP**	**Remote Wake-up** 0 = Release the USB bus from K state. 1 = Force USB bus to K (USB_D+ low, USB_D-: high) state, used for remote

		wake-up.
[4]	**PHYEN**	**PHY Transceiver Function Enable Bit** 0 = PHY transceiver function Disabled. 1 = PHY transceiver function Enabled.
[3]	**TOUT**	**Time-out Status** 0 = No time-out. 1 = No Bus response more than 18 bits time. **Note:** This bit is read only.
[2]	**RESUME**	**Resume Status** 0 = No bus resume. 1 = Resume from suspend. **Note:** This bit is read only.
[1]	**SUSPEND**	**Suspend Status** 0 = Bus no suspend. 1 = Bus idle more than 3ms, either cable is plugged off or host is sleeping. **Note:** This bit is read only.
[0]	**USBRST**	**USB Reset Status** 0 = Bus no reset. 1 = Bus reset when SE0 (single-ended 0) more than 2.5us. **Note:** This bit is read only.

USB Device VBUS Detection Register (USBD_VBUSDET)

Register	Offset	R/W	Description	Reset Value
USBD_VBUSDET	USBD_BA+0x014	R	USB Device VBUS Detection Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**PULLD**	**VBUS pull down resistor enable** 0: Pull down disable(open) 1: Pull down enable(short)
[7:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**VBUSDET**	**Device VBUS Detection** 0 = Controller is not attached to the USB host. 1 = Controller is attached to the USB host.

USB SETUP Token Buffer Segmentation Register (USBD_STBUFSEG)

Register	Offset	R/W	Description	Reset Value
USBD_STBUFSEG	USBD_BA+0x018	R/W	SETUP Token Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:3]	**STBUFSEG**	**SETUP Token Buffer Segmentation** It is used to indicate the offset address for the SETUP token with the USB Device SRAM starting address The effective starting address is USBD_SRAM address + {STBUFSEG, 3'b000} Where the USBD_SRAM address = USBD_BA+0x100h. **Note:** It is used for SETUP token only.
[2:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

USB Endpoint Status Register 0 (USBD_EPSTS0)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS0	USBD_BA+0x020	R	USB Device Endpoint Status Register 0	0x0000_0000

31	30	29	28	27	26	25	24
EPSTS7				EPSTS6			
23	22	21	20	19	18	17	16
EPSTS5				EPSTS4			
15	14	13	12	11	10	9	8
EPSTS3				EPSTS2			
7	6	5	4	3	2	1	0
EPSTS1				EPSTS0			

Bits	Description	
[31:28]	**EPSTS7**	**Endpoint 7 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[27:24]	**EPSTS6**	**Endpoint 6 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

[23:20]	**EPSTS5**	**Endpoint 5 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[19:16]	**EPSTS4**	**Endpoint 4 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[15:12]	**EPSTS3**	**Endpoint 3 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[11:8]	**EPSTS2**	**Endpoint 2 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[7:4]	**EPSTS1**	**Endpoint 1 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

[3:0]	**EPSTS0**	**Endpoint 0 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

USB Endpoint Status Register 1 (USBD_EPSTS1)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS1	USBD_BA+0x024	R	USB Device Endpoint Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
EPSTS11				EPSTS10			
7	6	5	4	3	2	1	0
EPSTS9				EPSTS8			

Bits	Description	
[31:16]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:12]	EPSTS11	**Endpoint 11 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[11:8]	EPSTS10	**Endpoint 10 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

[7:4]	**EPSTS9**	**Endpoint 9 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[3:0]	**EPSTS8**	**Endpoint 8 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

USB Frame Number Register (USBD_FN)

Register	Offset	R/W	Description	Reset Value
USBD_FN	USBD_BA+0x08C	R	USB Frame number Register	0x0000_0XXX

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

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:0]	**FN**	**Frame Number** These bits contain the 11-bits frame number in the last received SOF packet.

USB Drive SE0 Register (USBD_SE0)

Register	Offset	R/W	Description	Reset Value
USBD_SE0	USBD_BA+0x090	R/W	USB Device Drive SE0 Control Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SE0**

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**SE0**	**Drive Single Ended Zero in USB Bus** The Single Ended Zero (SE0) is when both lines (USB_D+ and USB_D-) are being pulled low. 0 = Normal operation. 1 = Force USB PHY transceiver to drive SE0.

USB Device VBUS Detection De-bounce Control Register (USBD_VDDIS)

Register	Offset	R/W	Description	Reset Value
USBD_VDDIS	USBD_BA+0x0A8	R/W	USB Device VBUS Detection De-bounce Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**VDDIS**	**VBUS Detection De-bounce Disable Bit** 0 = VBUS debounce control enabled. 1 = VBUS debounce control disabled. Note1: When this bit is set as 1, it must be set after USB interrupt (USBD_INTEN) configured. Note2: Before the chip enters power-down mode, setting this bit to 1 to support any level change on VBUS, D- and D- wake up function.

USB Buffer Segmentation Register (USBD_BUFSEGx)

Register	Offset	R/W	Description	Reset Value
USBD_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG7	USBD_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG8	USBD_BA+0x580	R/W	Endpoint 8 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG9	USBD_BA+0x590	R/W	Endpoint 9 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG10	USBD_BA+0x5A0	R/W	Endpoint 10 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG11	USBD_BA+0x5B0	R/W	Endpoint 11 Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:3]	**BUFSEG**	**Endpoint Buffer Segmentation** It is used to indicate the offset address for each endpoint with the USB SRAM starting address The effective starting address of the endpoint is USBD_SRAM address + { BUFSEG, 3'b000} Where the USBD_SRAM address = USBD_BA+0x100h. Refer to the section 5.16.5.7 for the endpoint SRAM structure and its

		description.
[2:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

USB Maximal Payload Register (USBD_MXPLDx)

Register	Offset	R/W	Description	Reset Value
USBD_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USBD_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USBD_MXPLD7	USBD_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000
USBD_MXPLD8	USBD_BA+0x584	R/W	Endpoint 8 Maximal Payload Register	0x0000_0000
USBD_MXPLD9	USBD_BA+0x594	R/W	Endpoint 9 Maximal Payload Register	0x0000_0000
USBD_MXPLD10	USBD_BA+0x5A4	R/W	Endpoint 10 Maximal Payload Register	0x0000_0000
USBD_MXPLD11	USBD_BA+0x5B4	R/W	Endpoint 11 Maximal Payload Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[8:0]	**MXPLD**	**Maximal Payload** Define the data length which is transmitted to host (IN token) or the actual data length which is received from the host (OUT token). It also used to indicate that the endpoint is ready to be transmitted in IN token or received in OUT token. i. When the register is written by CPU, For IN token, the value of MXPLD is used to define the data length to be transmitted and indicate the data buffer is ready. For OUT token, it means that the controller is ready to receive data from the host and the value of MXPLD is the maximal data length comes from host. ii. When the register is read by CPU, For IN token, the value of MXPLD is indicated by the data length be transmitted to host For OUT token, the value of MXPLD is indicated the actual data length receiving from host. **Note:** Once MXPLD is written, the data packets will be transmitted/received immediately after IN/OUT token arrived.

USB Configuration Register (USBD_CFGx)

Register	Offset	R/W	Description	Reset Value
USBD_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USBD_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USBD_CFG7	USBD_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000
USBD_CFG8	USBD_BA+0x588	R/W	Endpoint 8 Configuration Register	0x0000_0000
USBD_CFG9	USBD_BA+0x598	R/W	Endpoint 9 Configuration Register	0x0000_0000
USBD_CFG10	USBD_BA+0x5A8	R/W	Endpoint 10 Configuration Register	0x0000_0000
USBD_CFG11	USBD_BA+0x5B8	R/W	Endpoint 11 Configuration Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**CSTALL**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DSQSYNC	**STATE**		**ISOCH**	**EPNUM**			

Bits	Description	
[31:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**CSTALL**	**Clear STALL Response** 0 = Disable the device to clear the STALL handshake in setup stage. 1 = Clear the device to response STALL handshake in setup stage.
[8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[7]	**DSQSYNC**	**Data Sequence Synchronization** 0 = DATA0 PID. 1 = DATA1 PID. **Note:** It is used to specify the DATA0 or DATA1 PID in the following IN token transaction. Hardware will toggle automatically in IN token base on the bit.
[6:5]	**STATE**	**Endpoint STATE** 00 = Endpoint is Disabled. 01 = Out endpoint. 10 = IN endpoint. 11 = Undefined.
[4]	**ISOCH**	**Isochronous Endpoint** This bit is used to set the endpoint as Isochronous endpoint, no handshake. 0 = No Isochronous endpoint. 1 = Isochronous endpoint.
[3:0]	**EPNUM**	**Endpoint Number** These bits are used to define the endpoint number of the current endpoint

USB Extra Configuration Register (USBD_CFGPx)

Register	Offset	R/W	Description	Reset Value
USBD_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP7	USBD_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP8	USBD_BA+0x58C	R/W	Endpoint 8 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP9	USBD_BA+0x59C	R/W	Endpoint 9 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP10	USBD_BA+0x5AC	R/W	Endpoint 10 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP11	USBD_BA+0x5BC	R/W	Endpoint 11 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**SSTALL**	**Set STALL** 0 = Disable the device to response STALL. 1 = Set the device to respond STALL automatically.
[0]	**CLRRDY**	**Clear Ready** When the USBD_MXPLDx register is set by user, it means that the endpoint is ready to transmit or receive data. If the user wants to disable this transaction before the transaction start, users can set this bit to 1 to disable it and it is auto clear to 0. For IN token, write '1' to clear the IN token had ready to transmit the data to USB. For OUT token, write '1' to clear the OUT token had ready to receive the data from USB. This bit is write 1 only and is always 0 when it is read back.

5.17 Companding

5.17.1 Overview

The companding is used in digital communication systems to optimize signal-to-noise ratio with reduced data bit rates using non-linear algorithms. The ISD91500 supports telecommunications companding: A-law, u-law (G.711) and ADPCM (G.726). Both Encoder and Decoder has input and output FIFO. Encoder circuit has an 8-level 16 bits FIFO at encoder input, and an 8-level 8 bits output FIFO at encoder output. Decoder circuit has an 8-level 8 bits FIFO at decoder input, and an 8-level 16 bits output FIFO at decoder output. Each FIFO supports threshold interrupt and empty/full flag.

5.17.2 Features

- Compatible with CCITT G.726 and G.726 AnnexA
- Support 8-bit u-law/A-law, or 16-bit linear interfacing
- ADPCM rates : 40 kbps(5-bit ADPCM)
  32 kbps(4-bit ADPCM)
  24 kbps(3-bit ADPCM)
  16 kbps(2-bit ADPCM)
- Support full-duplex ,can encode and decode at same time with same bit rate
- 8-level FIFO with interrupt

5.17.3 Functional Description

5.17.3.1 ADPCM , A-law and u-law format

For ISD91500 companding encoder and decoder, user can set **CPD_CTRL.MODE** to select the companding algorithm as ADPCM or A/u-law. If ADPCM is set, 4 kinds of ADPCM bit rate is selectable in register **CPD_CTRL.BITRATE** by user. When A/u-law is selected, user can further choose A-law or u-law by **CPD_CTRL.LAW**. After the companding algorithm is set, the encoder input and decoder output can set as **CPD_CTRL.TYPE** to decide the data type asPCM or A/u-law.

5.17.3.2 FIFO pointer and flag

Companding FIFO pointer range is from 0~8.When FIFO pointer reach maximum value 8, it means the FIFO full and the FIFO full flag will issue as 1'b. When FIFO pointer decrease to minimum value 0, it means the FIFO empty and the FIFO empty flag will issue as 1'b. The companding FIFO also has overflow flag. When FIFO is full, an additional data is filled to FIFO. It will cause the overflow flag issue.

5.17.3.3 Interrupt Sources

The companding can be configured to generate an interrupt when the data level in the FIFO exceeds/less than a defined threshold. The interrupt condition is only cleared by disabling the interrupt or reading/writing values from/to the FIFO.

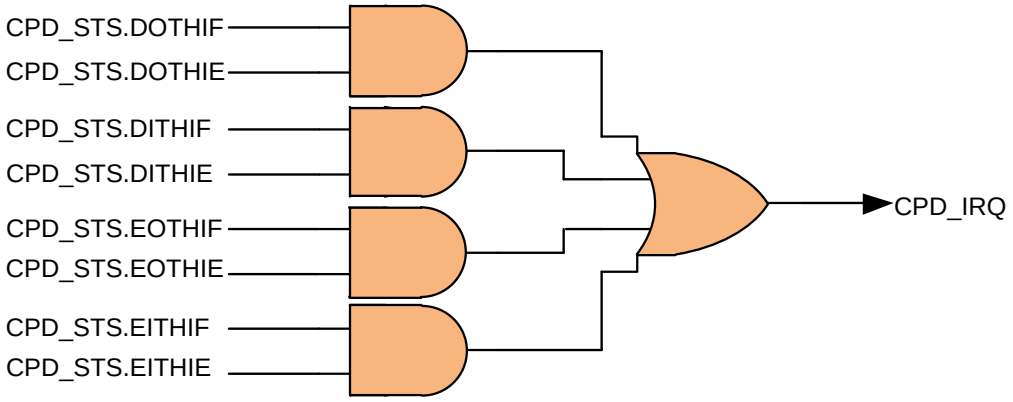


Figure 5.17-1 CPD controller interrupt

5.17.3.4 Configuring Companding setting

To operate the companding setting the following configuration is recommended:

- Enable Companding clock source by register CPDCKEN (CLK_AHBCLK[1]).
- Reset CPD IP block by register CPDRST (SYS_IPRST1[6]).
- Select CPD algorithm format and input/output data type.
- Enable CPD by using register CPD_CTRL.EN (CPD_CTRL[0]).
- Enable encoder/decoder threshold interrupt
- Start to write data to encoder/decoder input FIFO, and then receive the data from output FIFO

5.17.4 Companding Control Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
CPD Base Address: **CPD_BA = 0x5000_E000**				
CPD_CTRL	CPD_BA+0x00	R/W	CPD Control Register	0x0000_0000
CPD_STS	CPD_BA+0x04	R/W	CPD FIFO Status Register	0x0202_0202
CPD_ENCIN	CPD_BA+0x08	W	CPD Encoder Input FIFO	0x0000_0000
CPD_ENCOUT	CPD_BA+0x0C	R	CPD Encoder Output FIFO	0x0000_0000
CPD_DECIN	CPD_BA+0x10	W	CPD Decoder Input FIFO	0x0000_0000
CPD_DECOUT	CPD_BA+0x14	R	CPD Decoder Output FIFO	0x0000_0000

5.17.5 Companding Control Register Description

CPD Control Register (CPD_CTRL)

Register	Offset	R/W	Description	Reset Value
CPD_CTRL	CPD_BA+0x00	R/W	CPD Control Register	0x0000_0000

31	30	29	28	27	26	25	24
DOTHIE	**DOTH**			**DITHIE**	**DITH**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
EOTHIE	**EOTH**			**EITHIE**	**EITH**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**DECRST**	**ENCRST**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**BITRATE**		**LAW**	**TYPE**	**MODE**	**EN**

Bits	Description	
[31]	**DOTHIE**	**Decoder output FIFO Threshold Interrupt** 0 =Decoder output FIFO threshold interrupt Disabled 1 =Decoder output FIFO threshold interrupt Enabled.
[30:28]	**DOTH**	**Decoder output FIFO Threshold Level** If the valid data count of the FIFO data buffer is larger than or equal to DOTH (CPD_CTRL[30:28]) setting, the DOTHIF (CPD_STS[27]) will set to 1, else the DOTHIF (CPD_STS[27]) will be cleared to 0.
[27]	**DITHIE**	**Decoder input FIFO Threshold Interrupt** 0 =Decoder input FIFO threshold interrupt Disabled 1 =Decoder input FIFO threshold interrupt Enabled.
[26:24]	**DITH**	**Decoder input FIFO Threshold Level** If the valid data count of the FIFO data buffer is less than or equal to DITH (CPD_CTRL[26:24]) setting, the DITHIF (CPD_STS[19]) will set to 1, else the DITHIF (CPD_STS[19]) will be cleared to 0.
[23]	**EOTHIE**	**Encoder output FIFO Threshold Interrupt** 0 =Encoder output FIFO threshold interrupt Disabled 1 =Encoder output FIFO threshold interrupt Enabled.
[22:20]	**EOTH**	**Encoder output FIFO Threshold Level** If the valid data count of the FIFO data buffer is larger than or equal to EOTH (CPD_CTRL[22:20]) setting, the EOTHIF (CPD_STS[11]) will set to 1, else the EOTHIF (CPD_STS[11]) will be cleared to 0.

[19]	**EITHIE**	**Encoder input FIFO Threshold Interrupt** 0 =Encoder input FIFO threshold interrupt Disabled 1 =Encoder input FIFO threshold interrupt Enabled.
[18:16]	**EITH**	**Encoder input FIFO Threshold Level** If the valid data count of the FIFO data buffer is less than or equal to EITH (CPD_CTRL[18:16]) setting, the EITHIF (CPD_STS[3]) will set to 1, else the DITHIF (CPD_STS[3]) will be cleared to 0.
[15:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**DECRST**	**Decoder reset register** Write 1 to this bit will reset decoder state machine and clear input/output FIFO. This bit will auto change to 0 after reset done.
[8]	**ENCRST**	**Encoder reset register** Write 1 to this bit will reset encoder state machine and clear input/output FIFO. This bit will auto change to 0 after reset done.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:4]	**BITRATE**	**CPD ADPCM bitrate select** 0 = 16K bit/s (2 bits per sample) 1 = 24K bit/s (3 bits per sample) 2 = 32K bit/s (4 bits per sample) 3 = 40K bit/s (5 bits per sample)
[3]	**LAW**	**CPD A-law / u-law select** 0 = u-law 1 = A-law
[2]	**TYPE**	**CPD encoder input and decoder output type select under ADPCM algorithm** 0 = A-law / u-law 1 = PCM **Note:** This bit only works when **MODE** set as 0'b (ADPCM)
[1]	**MODE**	**CPD encode/decode algorithm select** 0 = ADPCM (G.726) 1 = A-law / u-law (G.711)
[0]	**EN**	**CPD enable control** 0 = CPD disable 1 = CPD enable

CPD FIFO Status Register (CPD_STS)

Register	Offset	R/W	Description	Reset Value
CPD_STS	CPD_BA+0x04	R/W	CPD FIFO Status Register	0x0202_0202

31	30	29	28	27	26	25	24
DOFPTR				**DOTHIF**		**DOE**	**DOF**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DIFPTR				**DITHIF**	**DIOV**	**DIE**	**DIF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EOFPTR				**EOTHIF**		**EOE**	**EOF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EIFPTR				**EITHIF**	**EIOV**	**EIE**	**EIF**

Bits	Description	
[31:28]	**DOFPTR**	**CPD decoder output FIFO Pointer (Read Only)** The FULL (CPD_STS[24]) and FIFOPTR (CPD_STS[31:28]) indicates the field that the valid data count within the Decoder output FIFO buffer. The maximum value shown in FIFOPTR is 8. When the using level of decoder output FIFO buffer equal to 8, The FULL (CPD_STS[24]) is set to 1.
[27]	**DOTHIF**	**CPD decoder output FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the FIFO data buffer is less than the setting value of DOTH (CPD_CTL[30:28]). 1 = The valid data count within the FIFO data buffer is more than or equal to the setting value of DOTH (CPD_CTL[30:28]). **Note:** This bit is cleared by writing 1 to itself.
[26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25]	**DOE**	**CPD decoder output FIFO empty flag** 0 = CPD decoder output FIFO is NOT empty 1 = CPD decoder output FIFO is empty
[24]	**DOF**	**CPD decoder output FIFO full flag** 0 = CPD decoder output FIFO is NOT full 1 = CPD decoder output FIFO is full

[23:20]	**DIFPTR**	**CPD decoder input FIFO Pointer (Read Only)** The FULL (CPD_STS[16]) and FIFOPTR (CPD_STS[23:20]) indicates the field that the valid data count within the decoder input FIFO buffer. The maximum value shown in FIFOPTR is 8. When the using level of decoder input FIFO buffer equal to 8, The FULL (CPD_STS[16]) is set to 1.
[19]	**DITHIF**	**CPD decoder input FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the FIFO data buffer is more than the setting value of DITH (CPD_CTL[26:24]). 1 = The valid data count within the FIFO data buffer is less than or equal to the setting value of DITH (CPD_CTL[26:24]). **Note:** This bit is cleared by writing 1 to itself.
[18]	**DIOV**	**CPD decoder input FIFO overflow flag** If decoder input FIFO (CPD->CPD_DEC_IN) is full, and an additional data is written to the FIFO, an overflow condition will occur and set this bit to logic 1. **Note:** This bit is cleared by writing 1 to itself.
[17]	**DIE**	**CPD decoder input FIFO empty flag** 0 = CPD decoder input FIFO is NOT empty 1 = CPD decoder input FIFO is empty
[16]	**DIF**	**CPD decoder input FIFO full flag** 0 = CPD decoder input FIFO is NOT full 1 = CPD decoder input FIFO is full
[15:12]	**EOFPTR**	**CPD encoder output FIFO Pointer (Read Only)** The FULL (CPD_STS[8]) and FIFOPTR (CPD_STS[15:12]) indicates the field that the valid data count within the encoder output FIFO buffer. The maximum value shown in FIFOPTR is 8. When the using level of encoder output FIFO buffer equal to 8, The FULL (CPD_STS[8]) is set to 1.
[11]	**EOTHIF**	**CPD encoder output FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the FIFO data buffer is less than the setting value of EOTH (CPD_CTL[22:20]). 1 = The valid data count within the FIFO data buffer is more than or equal to the setting value of EOTH (CPD_CTL[22:20]). **Note:** This bit is cleared by writing 1 to itself.
[10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**EOE**	**CPD encoder output FIFO empty flag** 0 = CPD encoder output FIFO is NOT empty 1 = CPD encoder output FIFO is empty

[8]	**EOF**	**CPD encoder output FIFO full flag** 0 = CPD encoder output FIFO is NOT full 1 = CPD encoder output FIFO is full
[7:4]	**EIFPTR**	**CPD encoder input FIFO Pointer (Read Only)** The FULL (CPD_STS[0]) and FIFOPTR (CPD_STS[7:4]) indicates the field that the valid data count within the encoder input FIFO buffer. The maximum value shown in FIFOPTR is 8. When the using level of encoder input FIFO buffer equal to 8, The FULL (CPD_STS[0]) is set to 1.
[3]	**EITHIF**	**CPD encoder input FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the FIFO data buffer is more than the setting value of EITH (CPD_CTL[18:16]). 1 = The valid data count within the FIFO data buffer is less than or equal to the setting value of EITH (CPD_CTL[18:16]). **Note:** This bit is cleared by writing 1 to itself.
[2]	**EIOV**	**CPD encoder input FIFO overflow flag** If encoder input FIFO (CPD->CPD_ENC_IN) is full, and an additional data is written to the FIFO, an overflow condition will occur and set this bit to logic 1. **Note:** This bit is cleared by writing 1 to itself.
[1]	**EIE**	**CPD encoder input FIFO empty flag** 0 = CPD encoder input FIFO is NOT empty 1 = CPD encoder input FIFO is empty
[0]	**EIF**	**CPD encoder input FIFO full flag** 0 = CPD encoder input FIFO is NOT full 1 = CPD encoder input FIFO is full

CPD Encoder Input FIFO (CPD_ENCIN)

Register	Offset	R/W	Description	Reset Value
CPD_ENCIN	CPD_BA+0x08	W	CPD Encoder Input FIFO	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**ENCIN**	**CPD encoder input FIFO** By writing to this register, encoder input data will be pushed onto the transmit FIFO. CPD will start encoding if this FIFO is not empty.

CPD Encoder Output FIFO (CPD_ENCOUT)

Register	Offset	R/W	Description	Reset Value
CPD_ENCOUT	CPD_BA+0x0C	R	CPD Encoder Output FIFO	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**ENCOUT**	**CPD encoder output FIFO** Reading this register will return data from encoder output data FIFO.

CPD Decoder Input FIFO (CPD_DECIN)

Register	Offset	R/W	Description	Reset Value
CPD_DECIN	CPD_BA+0x10	W	CPD Decoder Input FIFO	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**DECIN**	**CPD decoder input FIFO** By writing to this register, decoder input data will be pushed onto the transmit FIFO. CPD will start encoding if this FIFO is not empty.

CPD Decoder Output FIFO (CPD_DECOUT)

Register	Offset	R/W	Description	Reset Value
CPD_DECOUT	CPD_BA+0x14	R	CPD Decoder Output FIFO	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**DECOUT**	**CPD decoder output FIFO** Reading this register will return data from decoder output data FIFO.

5.18 Digital-to-Analog Converter(DAC) with Headphone Driver Output

5.18.1 Overview

The ISD91500 series includes stereo 24-bits Audio Digital -to- Analog converter(DAC). The DAC provides high quality audio playback suitable for all portable audio applications. There are also digital volume from -80dB to 6dB with 0.5dB/step in front of DAC. And mute function is also supported in the digital volume.

The DAC output is equipped with high quality classAB drivers intended for driving low impedance loads such as headphone. Analog headphone volume are also supported for better analog noise attenuate. The analog volume are range from -57dB to 6dB with 1dB/step.

Different oversampling rate ratio of DAC are supported from 32 to 256, and a special oversampling ratio 50 is also supported.

5.18.2 Features

- Stereo 24-bits Digital-to-Analog Convertor
- Supports programmable digital volume control from -80dB to 6dB in 0.5dB per step
- Support zero cross function
- Support soft volume and soft mute
- Support sample rate from 8KHz~48KHz
- Over sampling rate 32/50/64/128/256 Fs
- 16-level 2-word FIFO data buffer with stereo 32/24/16/8 bits data width
- Support FIFO threshold setting for interrupt
- Support ClassAB HPO driver with -75dB Total Harmonic Distortion (THD+N) performance
- With typical 90dB Signal-to-Noise (SNR) performance
- 20mW per channel drive capability into 32Ω load @3.3V
- Supports programmable analog headphone output volume control from -57dB to 6dB in 1dB steps
- DMA support for minimal CPU intervention

5.18.3 Block diagram

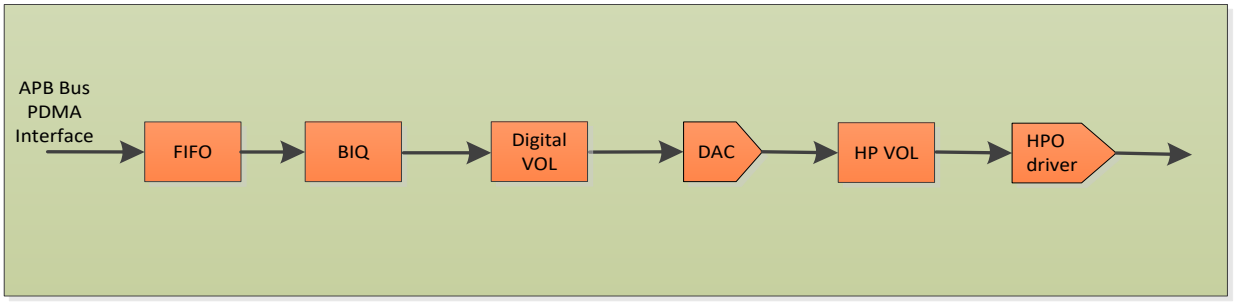


Figure 5.18-1 DAC Function Block Diagram

5.18.4 Functional Description

The DAC block receives audio data by writing PCM audio to the FIFO. FIFO is accessed through PDMA for ease of streaming. The audio stream is sampled by a zero-order hold and fed to an upsmaple filter. The signal then sent to the DAC driver stage. Master clock rate of the Delta-Sigma modulator is controlled by DAC_CLK.

5.18.4.1 DAC Clock Generation

The DAC module has six clock sources selected by register DACSEL (CLK_CLKSEL2[14:12]).

F_DAC_CLK is the frequency of DAC_CLK, F_DAC_CLK_SRC is the source clock of DAC.

F_DAC_CLK = F_DAC_CLK_SRC / (DACDIV+1)

DACDIV is given by CLK_CLKDIV1[7:0]

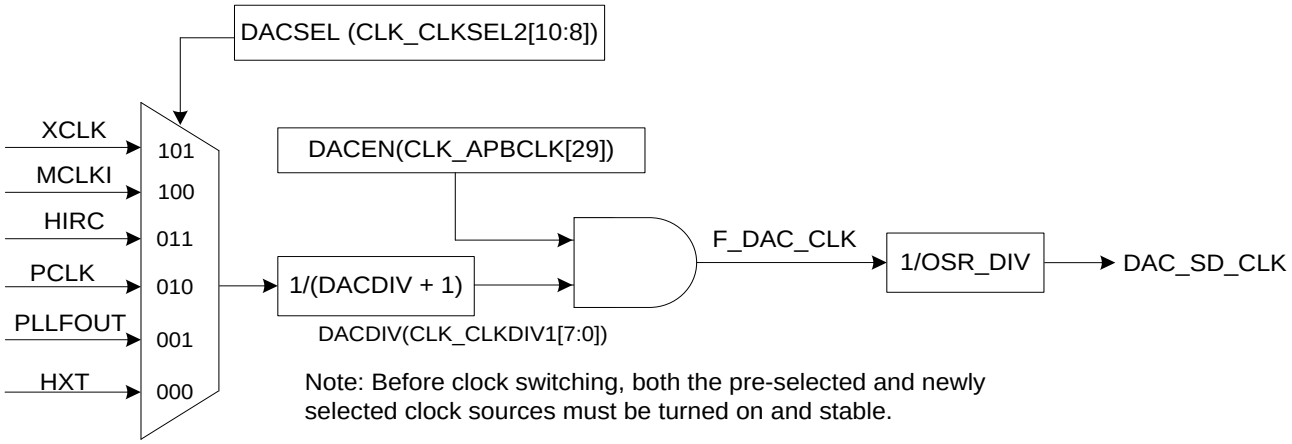


Figure 5.18-2 DAC Clock Control Diagram

5.18.4.2 Determining Sample Rate

The sample rate at which the DAC block consumes audio data is given by:

Fs = F_DAC_CLK / K

K = 250, if CLKSET (DAC_CTL0[31]) is 1.

K = 256, if CLKSET (DAC_CTL0[31]) is 0.

DAC_SD_CLK is the clock of the Sigma-Delta Converter, maximum is 6.144MHz.

DAC_SD_CLK = F_DAC_CLK / OSR_DIV

OSR_DIV is defined by DAC_CTL1[15:13], as shown in Table 5.18-1

Effective oversampling rate is defined as DAC_SD_CLK/Fs

So, effective OSR = K / OSR_DIV

Table 5.18-1 Effective OSR for different OSR_DIV setting

DAC_CTL1[15:13]	OSR_DIV	K	DAC_SD_CLK	Effective OSR
b'000	4	256	F_DAC_CLK/4	64
b'001	8	256	F_DAC_CLK/8	32
b'010	2	256	F_DAC_CLK/2	128

b'100	1	256	F_DAC_CLK/1	256

For CLKSET (DAC_CTL0[31]) = 1, OSR100 (DAC_CTL1[11]) and MIPS500 (DAC_CTL1[12]) must be set. The OSR_DIV is automatically set to 5. OSRSEL is invalid in this case. The effective OSR is as shown in Table 5.18-2.

Table 5.18-2 Effective OSR for special oversampling ratio

DAC_CTL1[12:11]	OSR_DIV	K	DAC_SD_CLK	Effective OSR
b'11	5	250	F_DAC_CLK/5	50

Table 5.18-3 Sample Rates for CLKSET (DAC_CTL0[31]) = 0

Fs	K	F_DAC_CLK	OSR	DAC_SD_CLK	OSR_DIV	DAC_CTL1[15:11]
48KHz	256	12.288MHz	128	6.144MHz	2	b'01000
			64	3.072MHz	4	b'00000
			32	1.536MHz	8	b'00100
32KHz	256	8.192MHz	128	4.096MHz	2	b'01000
			64	2.048MHz	4	b'00000
			32	1.024MHz	8	b'00100
24KHz	256	6.144MHz	256	6.144MHz	1	b'10000
			128	3.072MHz	2	b'01000
			64	1.536MHz	4	b'00000
			32	768KHz	8	b'00100
16KHz	256	4.096MHz	256	4.096MHz	1	b'10000
			128	2.048MHz	2	b'01000
			64	1.024MHz	4	b'00000
			32	512KHz	8	b'00100
8KHz	256	2.048MHz	256	2.048MHz	1	b'10000
			128	1.024MHz	2	b'01000
			64	512KHz	4	b'00000
			32	256KHz	8	b'00100

Table 5.18-4 Sample Rates for CLKSET (DAC_CTL0[31]) = 1

Fs	K	F_DAC_CLK	OSR	DAC_SD_CLK	OSR_DIV	DAC_CTL1[15:11]
48KHz	250	12MHz	50	2.4MHz	5	b'00011
32KHz	250	8MHz	50	1.6MHz	5	b'00011
24KHz	250	6MHz	50	1.2MHz	5	b'00011
16KHz	250	4MHz	50	800KHz	5	b'00011
8KHz	250	2MHz	50	400KHz	5	b'00011

5.18.4.3 **FIFO Data Operation**

FIFO bits is 32 bits. The channel position is as below. The bit-width of audio data in a channel block can be 8, 16 or 24bits. The memory arrangements of audio data for various settings are shown in Figure 5.18-3 to Figure 5.18-6. Due to the audio data in FIFO must be word alignment, user should implement the data to follow the word alignment format which shown in Figure 5.18-3 to Figure 5.18-6 first. And then write the audio data to FIFO register by word at one time.

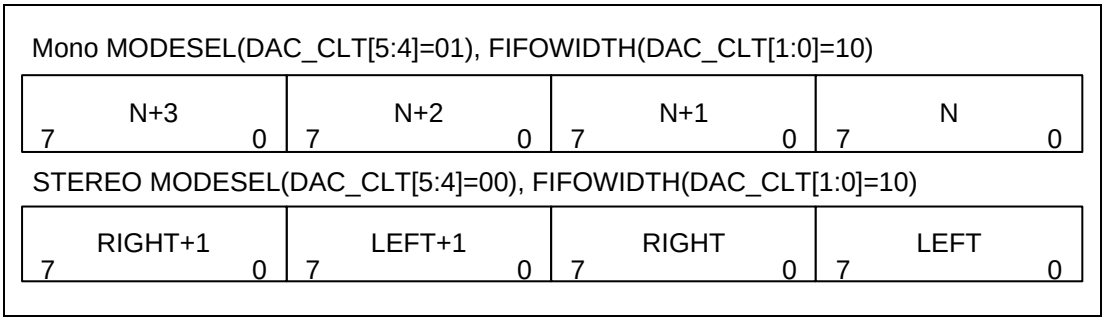


Figure 5.18-3 Audio DAC FIFO Contents for 8bits

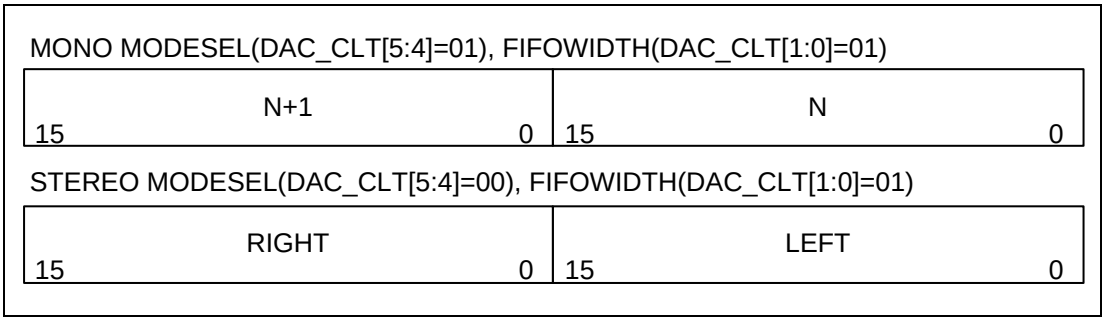


Figure 5.18-4 Audio DAC FIFO Contents for 16bits

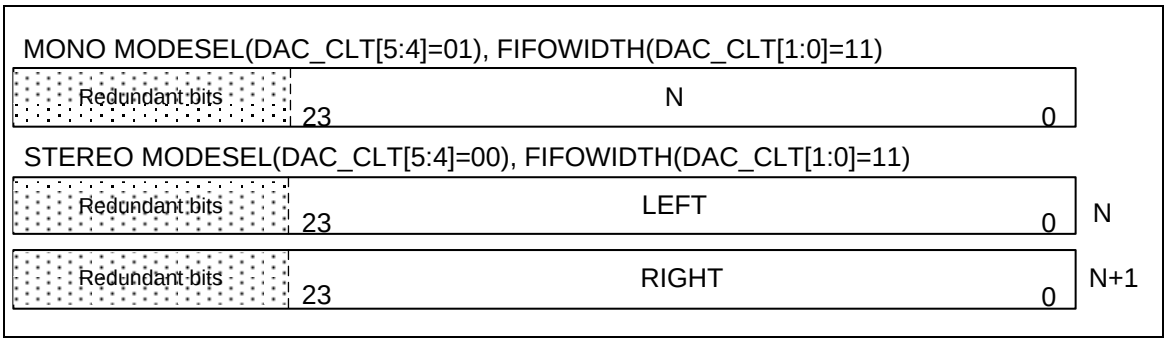


Figure 5.18-5 Audio DAC FIFO Contents for 24bits

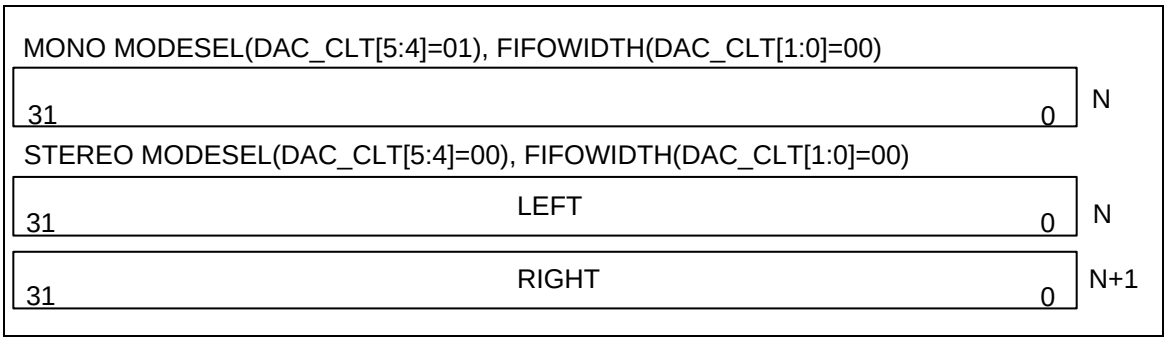


Figure 5.18-6 Audio DAC FIFO Contents for 32bits

5.18.4.4 **FIFO pointer and flag**

The DAC has FIFO pointer for user to indicate the valid data in FIFO.The FIFO block diagram is shown in Figure 5.18-7. When one word(32-bit) valid data was insert in DAC FIFO, the FIFO pointer number will plus one. When one word(32-bit) valid data was output from DAC FIFO, the FIFO pointer number will minus one.

The FIFO pointer range is from 0~32.When FIFO pointer reach maximum value 32, it means the FIFO full and the FIFO full flag (DAC_FIFOSTS[0]) will issue as 1'b. When FIFO pointer decrease to minimum value 0, it means the FIFO empty and the FIFO empty flag (DAC_FIFOSTS[1]) will issue as 1'b.

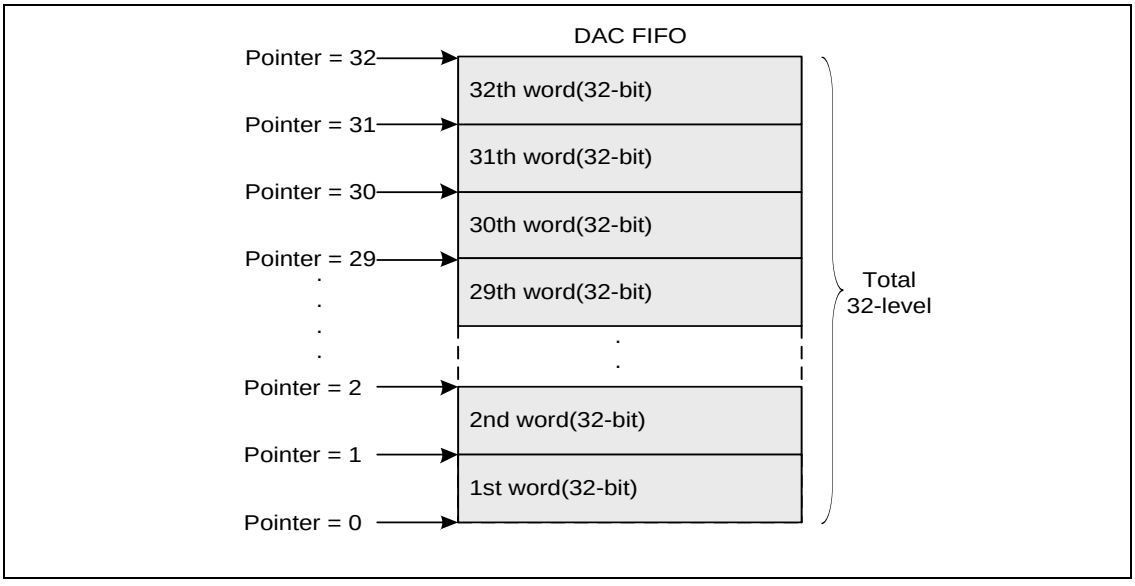


Figure 5.18-7 Audio DAC FIFO pointer block

5.18.4.5 Peripheral DMA Request

Normal use of the audio DACis with PDMA. In this mode DAC requests PDMA service whenever there is space in FIFO. PDMA channel will copy data from a streaming buffer to the DAC FIFO and alert the CPU when buffer is empty. In this way an entire buffer of data can be sent to DAC without any CPU intervention. When using PDMA for DAC, the PDMA transfer bit width must set as 32 bit.

5.18.4.6 DAC Soft Mute Function

The Soft Mute function ramps down the DAC digital volume to zero when it is enabled by SMCTL (DAC_CTL3[0]). When the soft mute function is disabled, the volme increases to the register-specified volume level for each channel if the DAC path has been enabled by setting DACENR/L (DAC_CTL1[1:0]). This function is beneficial for using the DAC without introducing pop sounds. In addition, UNMUTECTL (DAC_CTL3[2]) controls the ramp rate for each volume step when soft mute function is disabled.

5.18.4.7 Unused Headphone Outputs

When headphone output pins are being used, the voltage of the headphone outputs will be very close to ½ of the VCCA voltage that is present on the VMIDH.

In all cases, the DC blocking capacitors of headphone outputs will become charged to the operating voltage. The goal to reduce pops and clicks is to insure that the charge voltage on these capacitors does not change suddenly at any time.

When headphons outputs are in a not-used operating condition, it is desirable to keep the voltage on the headphone outputs at the same voltage level as the voltage of the used operating condition. This is accomplished using VMIDH that are at the required voltage. This helps to prevent pop noise when the headphone output is re-enabled. This type of connection is known as a “clamp” condition. Figure 5.18-8 summarises the clamp options for the headphone outputs.

Setting PDIBGEN, PDVBUF2/1 and PRECHG in register DAC_ANA1, the DC blocking capacitors on HPOL/R are started be charged to the operating voltage through the resistor between VMID buffer and headphone outputs. When headphone outputs are in a not-used operating condition, setting CLPVMID (DAC_ANA1[24]) will connect the headphone outpus to internal DC voltage source to keep the voltage of headphone output capacitors the same as the headphone outputs in operating condition.

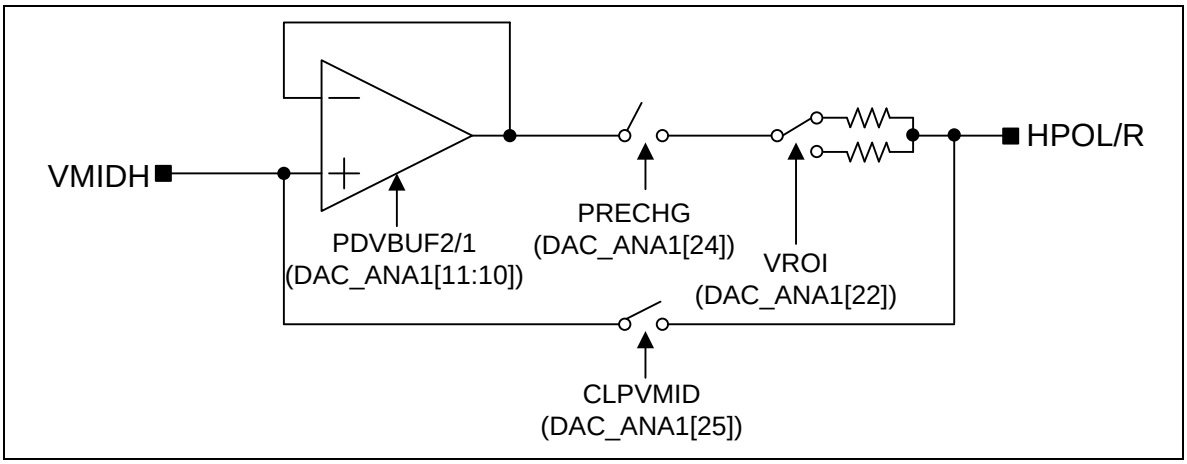


Figure 5.18-8 Headphone Output Clamp Options

5.18.4.8 Recommended Power up/down Sequence (RevC)

In order to reduce headphone outputs pop noise, it is recommended to use the specific sequence to configure the DAC path. Table 5.18-5 summaries the value of register DAC_ANA1 in each phase.

- Power-up Phase
  1. Turn on external power supplies. Wait for supply voltage to settle.
  2. Set ANAEN (CLK_APBCLK[31]) = 1, and set VMIDHRH, VMIDHRL and VMIDHPD to required value in register ANA_VMID. Wait for the VMIDH supply to settle. Refer note 1.
- Pre-charge Phase
  1. Select clock source in register DACSEL (CLK_CLKSEL2[10:8]), set DACEN (CLK_APBCLK[29]) = 1, and reset DAC block to default state by DACRST (SYS_IPRST1[29]).
  2. Set PRECHG, VOLEN2/1, PDBDAC2/1, ENDAC2/1, ENCLK2/1 = 1, and set PDIBGEN, PDVBUF2/1, PDFLITSM2/1 = 0 in register DAC_ANA1. Wait for the HPOL/R supply to settle. Refer note 2.
- Operational Phase
  1. Set ENHP2/1 = 3, VOLEN2/1, PDBDAC2/1, ENDAC2/1, ENCLK2/1 = 1, and set PRECHG, CLPVMID, PDIBGEN, PDVBUF2/1,PDFLITSM2/1 = 0 in register DAC_ANA1.
  2. According to the desired sample rate and data width, set related functions in register DAC_CTL0 and DAC_CTL1.
  3. If need, set the PDMA related registers to transmit audio data to the DAC.
- Clamp Phase
  1. Set PRECHG, ENHP2, ENHP1, VOLEN2, VOLEN1, PDBDAC2 and PDBDAC1 = 0, and set CLPVMID, PDIBGEN, PDVBUF2, PDVBUF1, PDFLITSM2 and PDFLITSM1 = 1 in register DAC_ANA1.
- Power-down Phase
  1. Disable all functions.
  2. Turn off external power supplies.

Note 1:

Select the value of VMIDHRH and VMIDHRL bit based on the VMIDH voltage settle time (VMIDHRL = 0 for fastest startup, VMIDHRH = 0 for slowest startup). The VMIDH voltage settle time is defined by the VMIDHRH/L bit and the external decoupling capacitor on VMIDH.

Note 2:

In the per-charge phase, the VMID buffer charges the DC blocking capacitors on HPOL/R through the resistor between the VMID buffer to HPOL/R. The charge time is based on the capacitance of the DC blocking capacitor, the headphone loading, and the resistance of resistor between the VMID buffer to HPOL/R. The resistance between the VMID buffer and the HPOL/R can be controlled using the VROI bit.

Table 5.18-5 DAC_ANA1 Register Setting in Each Phase

DAC_ANA1 Register	Pre-charge Phase	Operational Phase	Clamp Phase
CLPVMID	0	0	1
PRECHG	1	0	0
VOLEN2/1	1	1	0
PDBDAC2/1	1	1	0
PDIBGEN	0	0	1
PDVBUF2/1	0	0	1
PDFLITSM2/1	0	0	1
ENHP2/1	0	3	0
ENDAC2/1	1	1	0
ENCLK2/1	1	1	0

5.18.5 DAC Control Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
DAC Base Address: **DAC_BA = 0x4007_0000**				
DAC_DAT	DAC_BA+0x00	W	DAC FIFO Data Write Register	0xXXXX_XXXX
DAC_CTL0	DAC_BA+0x04	R/W	DAC Control Register 0	0x0000_0000
DAC_DVOL	DAC_BA+0x08	R/W	DAC Digital Volume Control Register	0x0000_CFCF
DAC_FIFOSTS	DAC_BA+0x0C	R/W	DAC FIFO Status Register	0x0000_0002
DAC_PDMACTL	DAC_BA+0x10	R/W	DAC PDMA Control Register	0x0000_0000
DAC_CTL1	DAC_BA+0x200	R/W	DAC Control Register 1	0x0003_8000
DAC_CTL3	DAC_BA+0x20C	R/W	DAC Control Register 3	0x0000_0000
DAC_ANA0	DAC_BA+0x300	R/W	DAC Analog Block Control Register 0	0x0000_40B9
DAC_ANA1	DAC_BA+0x304	R/W	DAC Analog Block Control Register 1	0x0001_FF00

5.18.6 DAC Control Register Description

DAC FIFO Data Register(DAC_DAT)

Register	Offset	R/W	Description	Reset Value
DAC_DAT	DAC_BA+0x00	W	DAC FIFO Data Write Register	0xXXXX_XXXX

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

Bits	Description	
[31:0]	**FIFO**	**FIFO Data Input Register** DAC contains 32 words (32x32 bit) data buffer for data transmit. A write to this register pushes data onto the FIFO data buffer and increments the write pointer. This is the address that CPU/PDMA writes audio data to. The remaining word number is indicated by FIFOPTR (DAC_FIFOSTS[9:4]).

DAC Control Register 0 (DAC_CTL0)

Register	Offset	R/W	Description	Reset Value
DAC_CTL0	DAC_BA+0x04	R/W	DAC Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
CLKSET	SWRST	FCLR		Reserved			
23	22	21	20	19	18	17	16
Reserved							TH
15	14	13	12	11	10	9	8
TH				THIE	Reserved		
7	6	5	4	3	2	1	0
FIFOEN		MODESEL		Reserved		FIFOWIDTH	

Bits	Description	
[31]	**CLKSET**	**Working Clock Selection** 0 = The sampling rate is DACCLK/256 1 = The sampling rate is DACCLK/250
[30]	**SWRST**	**State Machine Software Reset** 0 = State Machine normal operation 1 = State Machine Reset
[29:28]	**FCLR**	**FIFO Clear** 11 = Clear the FIFO. Others = Reserved. Do not use. **Note 1:** To clear the FIFO, need to write FCLR (DAC_CTL[29:28]) to 11b, and can read the EMPTY (DAC_FIFOSTS[1]) bit to make sure that the FIFO has been cleared. **Note 2:** This field is auto cleared by hardware.
[27:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16:12]	**TH**	**FIFO Threshold Level** If the valid data count of the FIFO data buffer is less than or equal to TH (DAC_CTL[16:12]) setting, the THIF (DAC_FIFOSTS[2]) will set to 1, else the THIF (DAC_FIFOSTS[2]) will be cleared to 0.
[11]	**THIE**	**FIFO Threshold Interrupt** 0 = FIFO threshold interrupt Disabled 1 = FIFO threshold interrupt Enabled.

[10:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:6]	**FIFOEN**	**DAC FIFO enable control** 00 = FIFO disable 11 = FIFO enable Others = Reserved. Do not use.
[5:4]	**MODESEL**	**Data Control in FIFO** 00 = Data is stereo format 01 = Data is monaural format Others = Reserved. Do not use. **Note :** Only left channel will output audio stream in mono mode.
[3:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**FIFOWIDTH**	**FIFO Data Width** This bit field is used to define the bit-width of data word and valid bits in register DAC_DAT. 00 = The bit-width of data word is 32-bit, valid bits is DAC_DAT[31:0]. 01 = The bit-width of data word is 16-bit, valid bits is DAC_DAT[15:0]. 10 = The bit-width of data word is 8-bit, valid bits is DAC_DAT[7:0]. 11 = The bit-width of data word is 24-bit, valid bits is DAC_DAT[23:0].

DAC Digital Volume Control Register (DAC_DVOL)

Register	Offset	R/W	Description	Reset Value
DAC_DVOL	DAC_BA+0x08	R/W	DAC Digital Volume Control Register	0x0000_CFCF

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:8]	**DACRVOL**	**DACR Digital Volume Control Register** 0xff = +6dB 0xfe = +5.5dB ▼ 0xf3 = 0dB 0xf2= -0.5dBdB ▼ 0x53= -80dB 0x52= Reserved ▼ 0x01= Reserved 0x00 = Mute **Note:** Volume per step 0.5dB
[7:0]	**DACLVOL**	**DACL Digital Volume Control Register** 0xff = +6dB 0xfe = +5.5dB ▼ 0xf3 = 0dB 0xf2= -0.5dBdB ▼ 0x53= -80dB

		0x52= Reserved ▼ 0x01= Reserved 0x00 = Mute **Note:** Volume per step 0.5dB

DAC FIFO Status Register (DAC_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
DAC_FIFOSTS	DAC_BA+0x0C	R/W	DAC FIFO Status Register	0x0000_0002

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

Bits	Description	
[31:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9:4]	**FIFOPTR**	**FIFO Pointer (Read Only)** The FULL (DAC_FIFOSTS[0]) and FIFOPTR (DAC_FIFOSTS[9:4]) indicates the field that the valid data count within the DAC FIFO buffer. The maximum value shown in FIFOPTR is 32. When the using level of DAC FIFO buffer equal to 32, The FULL (DAC_FIFOSTS[0]) is set to 1. The minimum value shown in FIFOPTR is 0. When the using level of DAC FIFO buffer equal to 0, The EMPTY (DAC_FIFOSTS[1]) is set to 1.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**THIF**	**FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the FIFO data buffer is more than the setting value of TH (DAC_CTL[24:20]). 1 = The valid data count within the FIFO data buffer is less than or equal to the setting value of TH (DAC_CTL[24:20]).
[1]	**EMPTY**	**FIFO Empty (Read Only)** 0 = FIFO is not empty. 1 = FIFO is empty.
[0]	**FULL**	**FIFO Full (Read Only)** 0 = FIFO is not full. 1 = FIFO is full.

DAC PDMA Control Register (DAC_PDMACTL)

Register	Offset	R/W	Description	Reset Value
DAC_PDMACTL	DAC_BA+0x10	R/W	DAC PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**PDMAEN**	**PDMA Transfer Enable Bit** 0 = PDMA data transfer Disabled. 1 = PDMA data transfer Enabled.

DAC Control Register 1 (DAC_CTL1)

Register	Offset	R/W	Description	Reset Value
DAC_CTL1	DAC_BA+0x200	R/W	DAC control register 1	0x0003_8000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						DISDEM	Reserved
15	14	13	12	11	10	9	8
OSRDIV			MISP500	OSR100	DEMDITHER		
7	6	5	4	3	2	1	0
DEMDITHER	SDDITHER					DACENR	DACENL

Bits	Description	
[31:18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17]	**DISDEM**	**Disable DEM (dynamic element matching)** 0 = Enable 1 = Disable
[16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:13]	**OSRDIV**	**DAC Oversample Rate Data Clock Divider Selection** 000 = OSR_DIV 4 001 = OSR_DIV 8 010 = OSR_DIV 2 100 = OSR_DIV 1 Others = Reserved , do not use
[12]	**MISP500**	**Indicates '1' when MCLK_SRC/FS = 500**
[11]	**OSR100**	**Reads '1' when OSR = 100x**
[10:7]	**DEMDITHER**	**DEM dither control** Set Probability of DEM Dithering Set probability of first order DEM dithering. Each level increments probability by 1/16 **0000 = No dithering** **0001= 1/16** **0010 = 2/16**

		. ~ . **1111 = 15/16**
[6:2]	**SDDITHER**	**SDMOD dither control** **Number of bits of dithering on SD Modulator . Each level increments dithering by 1 bit** **0000 = No Dithering** **0001 = 1** . ~ . **1111 = 15**
[1]	**DACENR**	**SDMOD enable control for right channel**
[0]	**DACENL**	**SDMOD enable control for left channel**

DAC Control Register 3 (DAC_CTL3)

Register	Offset	R/W	Description	Reset Value
DAC_CTL3	DAC_BA+0x20C	R/W	DAC control register 3	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**ZCEN**	**DAC zero cross enable** 0 = Disable 1 = Enable
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**UNMUTECTL**	**Power-up soft unmute control** 0 = 512 MCLK per step soft unmute 1 = 32 MCLK per step soft unmute
[1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**SMCTL**	**Soft mute control** 0 = Gradually increase DAC volume to volume register setting 1 = Gradually decrease DAC volume to zero

DAC Analog Block Control Register 0 (DAC_ANA0)

Register	Offset	R/W	Description	Reset Value
DAC_ANA0	DAC_BA+0x300	R/W	DAC Analog Block Control Register 0	0x0000_40B9

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CLKINV	**VREFSEL**		**IBADJV1P5**		**CKDLYV1P5**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CAPV1P5		**BV1P5**					

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**CLKINV**	**Clock input inverse** 0 = not inverse. 1 = inverse.
[14:13]	**VREFSEL**	**DAC Vref select control** 00 = vccx 01 = 2.2V(@VCC=3.3V) 10 = 2.4V(@VCC=3.3V) (default) 11 = 2.6V(@VCC=3.3V)
[12:11]	**IBADJV1P5**	**BIAS current adjust control** 00 = 20uA. 01 = 25uA. 10 = 17uA. 11 = 10.8uA.

[10:8]	**CKDLYV1P5**	**Delay clock choice for DAC** 000 = clk_3. 001 = clk_4. 010 = clk_5. 011 = clk_6. 100 = clk_7. 101 = clk_0. 110 = clk_1. 111 = clk_2.
[7:6]	**CAPV1P5**	**Bypass cap setting** 00 = 0C. 01 = 1C. 10 = 2C. 11 = 3C.
[5:0]	**BV1P5**	**HP output Volume Control** 0'h = -57dB ~ 39'h = 0dB ~ 3F'h = +6dB 1dB/step Default = 39'h

DAC Analog Block Control Register 1 (DAC_ANA1)

Register	Offset	R/W	Description	Reset Value
DAC_ANA1	DAC_BA+0x304	R/W	DAC Analog Block Control Register 1	0x0001_FF00

31	30	29	28	27	26	25	24
Reserved						**CLPVMID**	**PRECHG**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**VROI**	**VOLMUTE**	**VOLEN2**	**VOLEN1**	**PDBDAC2**	**PDBDAC1**	**Reserved**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**PDIBGEN**	**PDVBUF2**	**PDVBUF1**	**PDFLITSM2**	**PDFLITSM1**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ENHP2		**ENHP1**		**ENDAC2**	**ENDAC1**	**ENCLK2**	**ENCLK1**

Bits	Description	
[31:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25]	**CLPVMID**	**Clamp VMID Control** Whenever the DAC is disabled, if this bit is set to 1, the HPOL/R will remain connected to VMIDH to keep the DC level on external DC blocking capacitors. This helps to prevent pop noise when the DAC is re-enabled. 0 = Off. 1 = On.
[24]	**PRECHG**	**Precharge Control** Before enabling headphone driver, the external DC blocking capacitors needs to be charged to the same DC level as the VMIDH to reduce the pop noise when the headphone driver enabled. The bit controls the connection between the HPOL/R and the VMID buffer. 0 = Off. 1 = On.
[23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[22]	**VROI**	**VROI Control** Output resistance control option for tie-off of unused or disabled outputs. Unused outputs tie to internal voltage reference for reduced pops and clicks 0 = nominal tie-off impedance value of 1kΩ. 1 **=** nominal tie-off impedance value of 30kΩ.

[21]	**VOLMUTE**	**Volume mute control** 0 = unmute. 1 = mute
[20]	**VOLEN2**	**Right volume enable control** 0 = Disable. 1 = Enable.
[19]	**VOLEN1**	**Left volume enable control** 0 = Disable. 1 = Enable.
[18]	**PDBDAC2**	**Right DAC power down control** 0 = power off. 1 = power on.
[17]	**PDBDAC1**	**Left DAC power down control** 0 = power off. 1 = power on.
[16:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12]	**PDIBGEN**	**IBGEN block power down control** 0 = power on. 1 = power off.
[11]	**PDVBUF2**	**Right channel VMID buffer block power down control** 0 = power on. 1 = power off.
[10]	**PDVBUF1**	**Left channel VMID buffer block power down control** 0 = power on. 1 = power off.
[9]	**PDFLITSM2**	**Right channel smooth filter block power down control** 0 = power on. 1 = power down.
[8]	**PDFLITSM1**	**Left channel smooth filter block power down control** 0 = power on. 1 = power down.
[7:6]	**ENHP2**	**Right channel headphone driver block enable control** 00 = Disable. 11 **=** Enable. Others = Reserved. Do not use.

[5:4]	**ENHP1**	**Left channel headphone driver block enable control** 00 = Disable. 11 **=** Enable. Others = Reserved. Do not use.
[3]	**ENDAC2**	**Right channel DAC enable control** 0 = Disable. 1 = Enable
[2]	**ENDAC1**	**Left channel DAC enable control** 0 = Disable. 1 = Enable.
[1]	**ENCLK2**	**Right channel DAC clock enable control** 0 = Disable. 1 = Enable.
[0]	**ENCLK1**	**Left channel DAC clock enable control** 0 = Disable. 1 = Enable.

5.19 Sigma- Delta Analog-to-Digital Converter (SDADC)

5.19.1 Overview

The ISD91500 series includes a Delta-Sigma Audio Analog-to-Digital converter.The converter can run at sampling rates up to 6.144MHz while a configurable decimation filter allows oversampling ratios of 64/128/256/384. A special oversampling ratio 62.5 is also supported when the input clock is 12M /24M/48M base.

In front of SDADC is an optional gain, the differential MIC input signal will be boost or decrease by the gain which is setting flexible as -12~60.5dB.

5.19.2 Features

- 32-level word FIFO data buffer with mono 32/24/16/8 bits data width
- Support FIFO threshold setting for interrupt
- Support sample rate 8K-48KHz.
- Oversampling ratio 64/128/256/384 and 62.5
- DMA support for minimal CPU intervention.
- Optional boost or minus gain for Audio Path
  - Programmable gain amplifier with 32 steps from -12 to 34.5dB in 1.5dB steps.
  - Boost gain stage of 26dB, giving maximum total gain of 60.5dB.
  - Mute function support

5.19.3 Block diagram

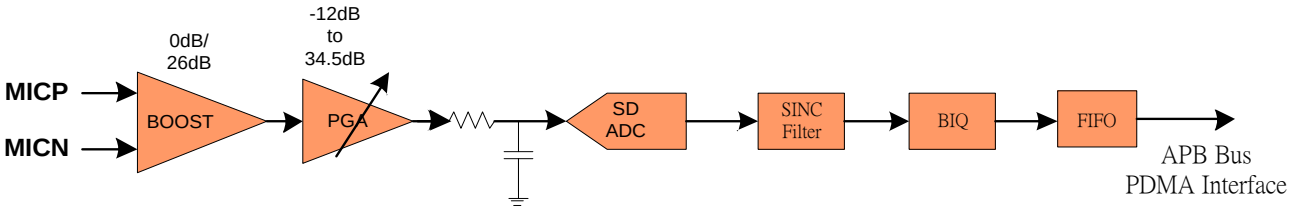


Figure 5.19-1 SD ADC Function Block Diagram

5.19.4 Operation

The SDADC is an Audio Sigma-Delta converter that operates by oversampling the analog input at low resolution and decimating the result by an over-sampling ratio to obtain a high resolution output which is pushed into the FIFO. The ultimate data rate is determined by the converter clock frequency, and the oversampling ratio. The audio signal stream generated by the SDADC is most conveniently handled by PDMA which can load data into a streaming audio buffer for further processing. Alternatively an interrupt driven approach can be used to monitor the FIFO.

5.19.4.1 SDADC Clock Generator

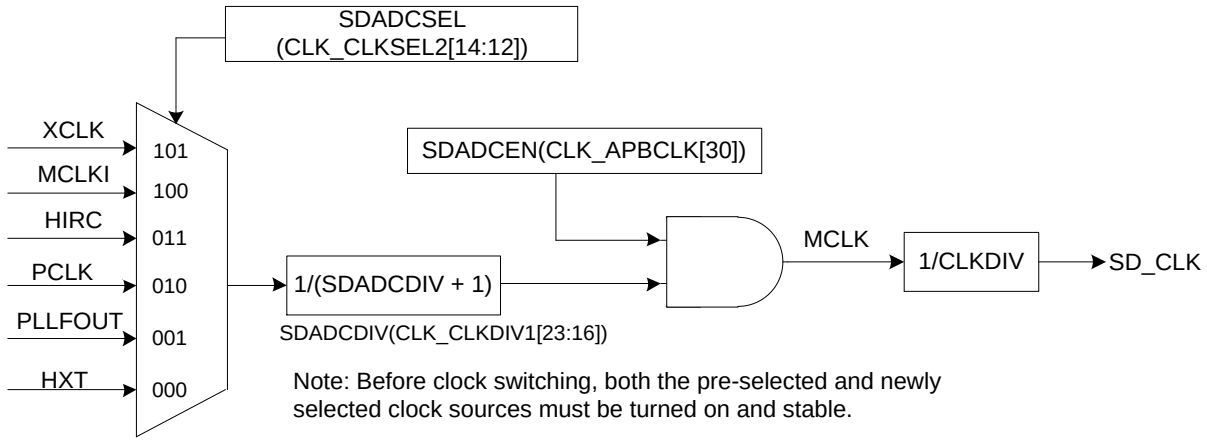


Figure 5.19-2 SDADC Clock Control Diagram

5.19.4.2 Determining Sample Rate

Sample rate is given by:

Fs = SD_CLK ÷ DSR

SD_CLK is the clock of the Sigma-Delta Converter, maximum is 6.144MHz.

SD_CLK = MCLK ÷ CLKDIV

Note: MCLK is the engine clock of SDADC logic, should be 4 times greater than SD_CLK (CLKDIV ≧ 4).

DSR is the over sampling rate on ADC conversion.

DSR = SDADC_CTL.DSRATE * BIQ_CTL.SDADCWNSR (when BIQ is on)

DSR = SDADC_CTL.DSRATE (when BIQ is off)

DSRATE and SDADCWNSR are the filter down sampling rates to expect sampling rate.

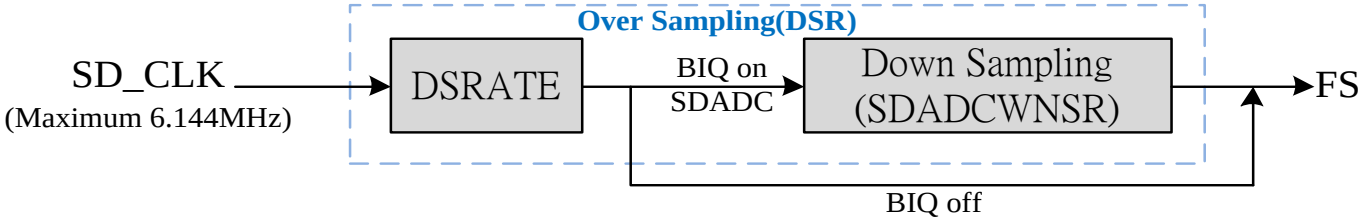


Figure 5.19-3 SDADC Sample Rate Diagram

Table 5.19-1 Sample Rates for MCLK 24.576MHz(BIQ on SDADC)

Fs	DSR	SD_CLK	CLKDIV	SDADC_CTL.DSRATE	BIQ_CTL.SDADCWNSR
48KHz	128	6.144MHz	4	64	2
				32	4
	64	3.072MHz	8	64	1
				32	2
				16	4
32KHz	192	6.144MHz	4	64	3
	128	4.096MHz	6	64	2
				32	4
	64	2.048MHz	12	64	1
				32	2
				16	4
16KHz	384	6.144MHz	4	64	6
	256	4.096MHz	6	64	4
	128	2.048MHz	12	64	2
				32	4
	64	1.024MHz	24	64	1
				32	2
				16	4
8KHz	384	3.072MHz	8	64	6
	256	2.048MHz	12	64	4
	128	1.024MHz	24	64	2
				32	4
	64	512KHz	48	64	1

Table 5.19-2 Sample Rates for MCLK 12.288MHz(BIQ on SDADC)

Fs	DSR	SD_CLK	CLKDIV	SDADC_CTL.DSRATE	BIQ_CTL.SDADCWNSR
48KHz	64	3.072MHz	4	64	1
				32	2
				16	4
32KHz	96	3.072MHz	4	32	3
				16	6
	64	2.048MHz	6	64	1
				32	2
				16	4
16KHz	128	2.048MHz	6	64	2
				32	4
	64	1.024MHz	12	64	1
				32	2
				16	4
8KHz	384	3.072MHz	4	64	6
	256	2.048MHz	6	64	4
	128	1.024MHz	12	64	2
				32	4
	64	512KHz	24	64	1

Table 5.19-3 Sample Rates for SDADC source clock from HIRC 48MHz(BIQ on SDADC)

SDADC_DIV	MCLK	Fs	DSR	SD_CLK	CLKDIV	SDADC_CTL.DSRATE	BIQ_CTL.SDADCWNSR
1	48MHz	48KHz	125	6MHz	8	62.5	2
2	24MHz		62.5	3MHz			1
1	16MHz	32KHz	187.5	6MHz	8	62.5	3
3			62.5	2MHz			1
2	24MHz	16KHz	187.5	3MHz	8	62.5	3
3	16MHz		125	2MHz			2
6	8MHz		62.5	1MHz			1
2	24MHz	8KHz	375	3MHz	8	62.5	6
3	16MHz		250	2MHz			4
6	8MHz		125	1MHz			2
12	4MHz		62.5	500KHz			1

5.19.4.3 FIFO Data Operation

FIFO bits is 32 bits. The channel position is as below. The bit-width of audio data in a channel block can be 8, 16, 24 or 32bits. The memory arrangements of audio data for various settings are shown in Figure 5.19-4. Due to the audio data in FIFO must be word alignment, user should write the audio data to FIFO register by word at one time.

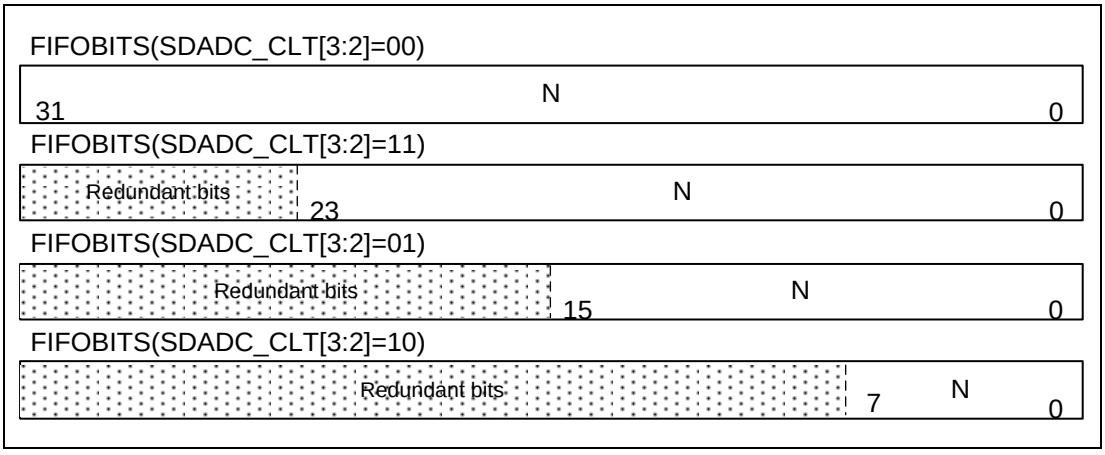


Figure 5.19-4 Audio SDADC FIFO Contents

5.19.4.4 Configuring Analog Path

To operate the SDADC the entire analog path from analog input to SDADC needs to be configured for correct operation without BIQ:

- Selecting and powering up VMID reference: power on VMID generator, power on both low and high value resistor.
- Wait 7 RC time, then turn off lower value resistor, and then wait 1or 2 RC time.
- Enable SDADC clock source (CLK_APBCLK0.SDADCCKEN, CLKSEL1.SDADCSEL).
- Reset SDADC block. (SYS_IPRST1.SDADCRST).
- Enable SDADC power(SDCHOP.PD = 0)
- Power up Boost, FEPGA and MICBIAS.
- Setup sample rate based on current MCLK(in Table) frequency and Set SDADC_CLKDIV.
- Setup FIFO data width SDADC_CTL.FIFOBITS
- Setup down sampling rate, set SDADC_CTL.RATESEL=0 then set SDADC_CTL.DSRATE for expected DSR & sampling rate(refer to Table)
- Setup PDMA channel to receive data from SDADC.
- Enable PDMA request.
- Enable SDADC conversion(SDADC_EN.SDADCEN).

To operate the SDADC the entire analog path from analog input to SDADC needs to be configured for correct operation with BIQ:

- Selecting and powering up VMID reference: power on VMID generator, power on both low and high value resistor.
- Wait 7 RC time, then turn off lower value resistor, and then wait 1or 2 RC time.
- Enable SDADC clock source (CLK_APBCLK0.SDADCCKEN, CLKSEL1.SDADCSEL).
- Reset SDADC block. (SYS_IPRST1.SDADCRST).
- Enable SDADC power (SDCHOP.SDADC_PD = 0)
- Power up Boost, FEPGA and MICBIAS.
- Setup sample rate based on current MCLK frequency and Set SDADC_CLKDIV
- Setup FIFO data width SDADC_CTL.FIFOBITS
- Setup down sampling rate, set SDADC_CTL.RATESEL=0 then set SDADC_CTL.DSRATE & BIQ_CTL.SDADCWNSR for expected DSR & sampling rate(refer to Table).
- Enable BIQ clock source (CLK_APBCLK0.BIQALCKEN).
- Enable BIQ on SDADC path (BIQ_CTL.DLCOEFF, BIQ_CTRL.BIQEN, BIQ_CTL.PATHSEL=0, BIQ_CTRL.STAGE, BIQ_CTRL.HPFON).
  Set BIQ coefficient(note: setup BIQ_CTL.DLCOEFF =1 and BIQ enable BIQ_CTL.BIQEN =1 for BIQ operation).
- Setup PDMA channel to receive data from SDADC.
- Enable PDMA request.
- Enable SDADC conversion(SDADC_EN.SDADCEN).

5.19.4.5 Interrupt Sources

The SDADC can be configured to generate an interrupt when the data level in the FIFO exceeds a defined threshold. The interrupt condition is only cleared by disabling the interrupt or reading values from the FIFO. In addition two comparators can monitor the SDADC FIFO output to generate interrupts when set levels are exceeded.

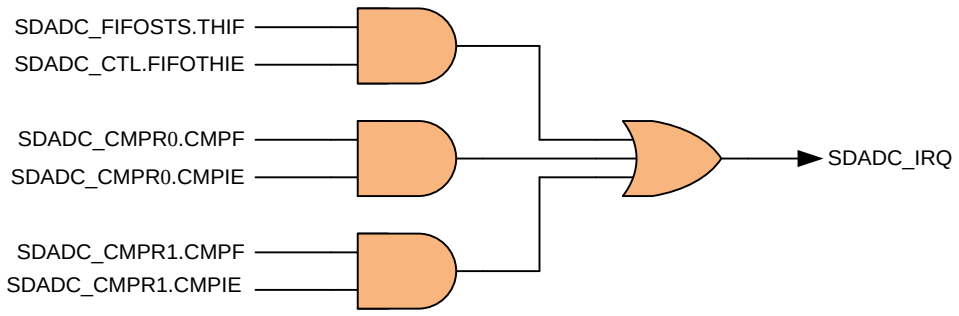


Figure 5.19-5 SDADC controller interrupt

5.19.4.6 Peripheral DMA Request

Normal use Normal use of the SDADC is with PDMA. In this mode SDADC requests PDMA service whenever data is in FIFO. PDMA channel will copy this data to a buffer and alert the CPU when buffer is full. In this way an entire buffer of data can be collected without any CPU intervention.

5.19.4.7 Programmable Gain Amplifier

The ISD91500 provides a Programmable Gain Amplifier (PGA) as the front-end to the SDADC to allow the adjustment of signal path gain. Figure 5.19-1 shows the signal path diagram. The PGA provides a gain from -12dB to 34.5dB in increments of 1.5dB steps using a 5-bit control, SDADC_ANA2[4:0]. The gain is monotonically increasing with 0x00 for lowest gain (-12dB) and 0x1f for the maximum gain (34.5dB). The signal path is enabled by powering up the gain elements (PUPGA, PUBOOST). The PGA and IP BOOST blocks can be muted with the SDADC_ANA0[5] / SDADC_ANA1[13] register. Input to the PGA can be either differential or single-ended on the PGA_INN input. The Analog MUX controls connection of the signal path to external pins.

The IP BOOST block can provide 0dB or 26dB of gain to provide a maximum gain of 60.5dB in the signal path.

5.19.5 SDADC Control Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
SDADC Base Address: **SDADC_BA = 0x400D_0000**				
SDADC_DAT	SDADC_BA+0x00	R	SD ADC FIFO Data Read Register	0xXXXX_XXXX
SDADC_EN	SDADC_BA+0x04	R/W	SD ADC Enable Register	0x0000_0000
SDADC_CLKDIV	SDADC_BA+0x08	R/W	SD ADC Clock Divider Register	0x0000_0000
SDADC_CTL	SDADC_BA+0x0C	R/W	SD ADC Control Register	0x0000_0000
SDADC_FIFOSTS	SDADC_BA+0x10	R/W	SD ADC FIFO Status Register	0x0000_0002
SDADC_PDMACTL	SDADC_BA+0x14	R/W	SD ADC PDMA Control Register	0x0000_0000
SDADC_CMPR0	SDADC_BA+0x18	R/W	SD ADC Comparator 0 Control Register	0x0000_0000
SDADC_CMPR1	SDADC_BA+0x1C	R/W	SD ADC Comparator 1 Control Register	0x0000_0000
SDADC_ANA0	SDADC_BA+0x20	R/W	SD ADC Analog Block Control Register 0	0x001C_9021
SDADC_ANA1	SDADC_BA+0x28	R/W	SD ADC Analog Block Control Register 1	0x0000_2004
SDADC_ANA2	SDADC_BA+0x2C	R/W	SD ADC Analog Block Control Register 2	0x0000_0008

5.19.6 SDADC Control Register Description

SD ADC FIFO Data Register(SDADC_DAT)

Register	Offset	R/W	Description	Reset Value
SDADC_DAT	SDADC_BA+0x00	R	SD ADC FIFO Data Read Register	0xXXXX_XXXX

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

Bits	Description	
[31:0]	**RESULT**	**Delta-Sigma ADC DATA FIFO Read** A read of this register will read data from the audio FIFO and increment the read pointer. A read past empty will repeat the last data. Can be used with SDADC_FIFOSTS.THIF interrupt to determine if valid data is present in FIFO. Data width can be selected by SDADC_CTL.FIFOBITS

SD ADC Enable Register(SDADC_EN)

Register	Offset	R/W	Description	Reset Value
SDADC_EN	SDADC_BA+0x04	R/W	SD ADC Enable Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**DINEDGE**	**SDADC data input clock edge selection** 1 = ADC clock positive edge latch 0 = ADC clock negetive edge latch
[0]	**SDADCEN**	**SDADC Enable** 1 = ADC Conversion enabled. 0 = Conversion stopped and ADC is reset including FIFO pointers.

SD ADC Clock Divider Register (SDADC_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SDADC_CLKDIV	SDADC_BA+0x08	R/W	SD ADC Clock Divider Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**CLKDIV**	**SDADC Clock Divider** This register determines the clock division ration between the incoming SD_CLK and the Sigma-Delta sampling clock of the ADC. This together with the over-sampling ratio (OSR) determines the audio sample rate of the converter. CLKDIV should be set to give a SD_CLK frequency in the range of 1.024-6.144MHz. CLKDIV = **SDADC Clock/Sample Rate/Down Sample Rate** **Note:** When SDADC_CTL.SPDS = 0'b , CLKDIV must ≧ 4 . When SDADC_CTL.SPDS = 1'b , CLKDIV must = 8 .

SD ADC Control Register (SDADC_CTL)

Register	Offset	R/W	Description	Reset Value
SDADC_CTL	SDADC_BA+0x0C	R/W	SD ADC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**SPDS**	**Reserved**			**THIE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FIFOTH				**FIFOBITS**		**DSRATE**	

Bits	Description	
[31:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12]	**SPDS**	**Specific down sampling ratio control** 0= Disable specific DS rate 1= Enable specific DS rate **Note:** When SPDS = 0'b , CLKDIV must ≧ 4 . When SPDS = 1'b , CLKDIV must = 8 .
[11:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**FIFOTHIE**	**FIFO Threshold Interrupt Enable** 0 = disable interrupt whenever FIFO level exceeds that set in FIFOTH. 1 = enable interrupt whenever FIFO level exceeds that set in FIFOTH.
[7:4]	**FIFOTH**	**FIFO Threshold:** Determines at what level the ADC FIFO will generate a interrupt. Interrupt will be generated when number of words present in ADC FIFO is > FIFOTH.

[3:2]	**FIFOBITS**	**FIFO Data Bits Selection** 00 = 32 bits 01 = 16 bits 10 = 8 bits 11 = 24 bits
[1:0]	**DSRATE**	**Down Sampling Ratio** 00 = reserved 01 = down sample X 16 10 = down sample X 32 11 = down sample X 64 when **SPDS = 0 .** 11 = down sample X 62.5 when **SPDS = 1 .**

SD ADC FIFO Status Register(SDADC_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
SDADC_FIFOSTS	SDADC_BA+0x10	R/W	SD ADC FIFO Status Register	0x0000_0002

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:4]	**POINTER**	**ADC FIFO Pointer (Read Only)** The FULL bit and FIFOPOINTER[4:0] indicates the field that the valid data count within the SDADC FIFO buffer. The Maximum value shown in FIFOPOINTER is 31. When the using level of SDADC FIFO Buffer equal to 32, The FULL bit is set to 1.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**THIF**	**ADC FIFO Threshold Interrupt Status (Read Only)** 1 = The valid data count within the ADC FIFO buffer is larger than or equal the setting value of FIFOTH. 0 = The valid data count within the transmit FIFO buffer is less than to the setting value of FIFOTH.
[1]	**EMPTY**	**FIFO Empty** 1= FIFO is empty. 0= FIFO is not empty.
[0]	**FULL**	**FIFO Full** 1 = FIFO is full. 0 = FIFO is not full.

SD ADC PDMA Control Register(SDADC_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SDADC_PDMACTL	SDADC_BA+0x14	R/W	SD ADC PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**PDMAEN**	**Enable SDADC PDMA Receive Channel** 1 = Enable SDADC PDMA. 0 = Disable SDADC PDMA.

SD ADC Compare Register 0 (SDADC_CMPR0)

Register	Offset	R/W	Description	Reset Value
SDADC_CMPR0	SDADC_BA+0x18	R/W	SD ADC Comparator 0 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CMPOEN	CMPD						
23	22	21	20	19	18	17	16
CMPD							
15	14	13	12	11	10	9	8
CMPD							
7	6	5	4	3	2	1	0
CMPMATCNT				CMPF	CMPCOND	CMPIE	ADCMPEN

Bits	Description	
[31]	**CMPOEN**	**Compare Match output FIFO zero** 1 = compare match then FIFO out zero 0 = FIFO data keep original one.
[30:8]	**CMPD**	**Comparison Data** 23 bit value to compare to FIFO output word.
[7:4]	**CMPMATCNT**	**Compare Match Count** When the A/D FIFO result matches the compare condition defined by CMPCOND, the internal match counter will increase by 1. When the internal counter reaches the value to (CMPMATCNT +1), the CMPF bit will be set.
[3]	**CMPF**	**Compare Flag** When the conversion result meets condition in ADCMPR0 this bit is set to 1. It is cleared by writing 1 to self.
[2]	**CMPCOND**	**Compare Condition** 1= Set the compare condition that result is greater or equal to CMPD 0= Set the compare condition that result is less than CMPD **Note:** When the internal counter reaches the value (CMPMATCNT +1), the CMPF bit will be set.

[1]	**CMPIE**	**Compare Interrupt Enable** 1 = Enable compare function interrupt. 0 = Disable compare function interrupt. If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMATCNT, CMPF bit will be asserted, if CMPIE is set to 1, a compare interrupt request is generated.
[0]	**ADCMPEN**	**Compare Enable** 0 = Disable compare. 1 = Enable compare. Set this bit to 1 to enable compare CMPDAT with FIFO data output.

SD ADC Compare Register 1 (SDADC_CMPR1)

Register	Offset	R/W	Description	Reset Value
SDADC_CMPR1	SDADC_BA+0x1C	R/W	SD ADC Comparator 1 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CMPOEN	CMPD						
23	22	21	20	19	18	17	16
CMPD							
15	14	13	12	11	10	9	8
CMPD							
7	6	5	4	3	2	1	0
CMPMATCNT				CMPF	CMPCOND	CMPIE	ADCMPEN

Bits	Description	
[31]	**CMPOEN**	**Compare Match output FIFO zero** 1 = compare match then FIFO out zero 0 = FIFO data keep original one.
[30:8]	**CMPD**	**Comparison Data** 23 bit value to compare to FIFO output word.
[7:4]	**CMPMATCNT**	**Compare Match Count** When the A/D FIFO result matches the compare condition defined by CMPCOND, the internal match counter will increase by 1. When the internal counter reaches the value to (CMPMATCNT +1), the CMPF bit will be set.
[3]	**CMPF**	**Compare Flag** When the conversion result meets condition in ADCMPR0 this bit is set to 1. It is cleared by writing 1 to self.
[2]	**CMPCOND**	**Compare Condition** 1= Set the compare condition that result is greater or equal to CMPD 0= Set the compare condition that result is less than CMPD **Note:** When the internal counter reaches the value (CMPMATCNT +1), the CMPF bit will be set.

[1]	**CMPIE**	**Compare Interrupt Enable** 1 = Enable compare function interrupt. 0 = Disable compare function interrupt. If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMATCNT, CMPF bit will be asserted, if CMPIE is set to 1, a compare interrupt request is generated.
[0]	**ADCMPEN**	**Compare Enable** 0 = Disable compare. 1 = Enable compare. Set this bit to 1 to enable compare CMPDAT with FIFO data output.

SD ADC Analog Block Control Register 0(SDADC_ANA0)

Register	Offset	R/W	Description	Reset Value
SDADC_ANA0	SDADC_BA+0x20	R/W	SD ADC Analog Block Control Register 0	0x001C_9021

31	30	29	28	27	26	25	24
Reserved		CHOPEN	CHOPPH	CHOPORD	CHOPFIX	CHOPCKPH	CHOPF
23	22	21	20	19	18	17	16
CHOPF	Reserved				CLASSA	Reserved	
15	14	13	12	11	10	9	8
CMLCK	Reserved						
7	6	5	4	3	2	1	0
MODE		MUTE	PU	VREF	BIAS		PD

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29]	**CHOPEN**	**SDADC chopper enable** 0 = disable (default) 1 = enable
[28]	**CHOPPH**	**SDADC chopper phase** When chopper is off: 0 = chopper switches in default state 1 = invert chopper switches
[27]	**CHOPORD**	**SDADC Chopper Order** 0 = 1st order dithering of chopper frequency (default) 1 = 2nd order dithering of chopper frequency
[26]	**CHOPFIX**	**SDADC Chopper Fixed Frequency** 0 = dither chopper frequency (default) 1 = choose fixed frequency
[25]	**CHOPCKPH**	**SDADC Chopper Clock phase selection** 0 = chopper transition after falling edge of SD_CLK (default) 1 = chopper transition after rising edge of SD_CLK

[24:23]	**CHOPF**	**SDADC Chopper Frequency in fixed chop mode** 00 = Fs/2 (default) 01 = Fs/4 10 = Fs/8 11 = Fs/16
[22:19]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[18]	**CLASSA**	**Enable PGA Class A mode of operation** 0 = Class AB 1 = Class A (default)
[17:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**CMLCK**	**PGA Common mode Threshold lock adjust enable** 0 = Enable 1 = Disable
[14:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:6]	**MODE**	**PGA mode selection;** MODE[0] = Disable anti-aliasing filter adjust MODE[1] = Noise enhancement enable. 0 = Disable 1 = Enable
[5]	**MUTE**	**PGA Mute control signal** 0 = disable 1 = enable
[4]	**PU**	**Power up PGA** 0 = disable 1 = enable
[3]	**VREF**	**SDADC Chopper in Reference Buffer** 0 = chopper off 1 = chopper on
[2:1]	**BIAS**	**SDADC Bias Current Selection** 00 = 1.35 01 = 1 10 = 0.67 11 = 1.68

[0]	**PD**	**SDADC Power Down** 0 = SDADC power on 1 = SDADC power off

SD ADC Analog Block Control Register 1(SDADC_ANA1)

Register	Offset	R/W	Description	Reset Value
SDADC_ANA1	SDADC_BA+0x28	R/W	SD ADC Analog Block Control Register 1	0x0000_2004

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						ACDC	
15	14	13	12	11	10	9	8
Reserved	BSTPUP	BSTMUTE	BSTMODE				Reserved
7	6	5	4	3	2	1	0
Reserved			DISCHRG	CLASSAEN	CMLCKEN	CMLCKADJ	

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17:16]	**ACDC**	**BST ACDC Control register** Action takes effect when **DISCHRG** =1 Bit[16]->ACDC[0] charges MICP to VMID Bit[17]->ACDC[1] charges MICN to VMID 00=Default
[15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14]	**BSTPUP**	Boost power on 0 = Power off(default) 1 = Power on
[13]	**BSTMUTE**	Boost mute 0 = Unmute 1 = Mute(default)

[12:9]	**BSTMODE**	**BST mode selection;** BSTMODE[0] = Gain setting (0 = 0dB , 1 = 26dB) BSTMODE[1] = AntiAliasing Filter Adjust. BSTMODE[2] = Short MICP/N. BSTMODE[3] = Noise Enhancement 0 = Disable 1 = Enable
[8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**DISCHRG**	**BST Charge inputs selected by ACDC[1:0] to VMID** 0 = Disable 1 = Enable
[3]	**CLASSAEN**	**Default 0'b**
[2]	**CMLCKEN**	**BST Common mode Threshold lock adjust enable** 0 = Enable 1 = Disable (default)
[1:0]	**CMLCKADJ**	**Default 00'b**

SD ADC Analog Block Control Register 2(SDADC_ANA2)

Register	Offset	R/W	Description	Reset Value
SDADC_ANA2	SDADC_BA+0x2C	R/W	SD ADC Analog Block Control Register 2	0x0000_0008

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

Bits	Description	
[31:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4:0]	**GAINSET**	**Select The PGA Gain Setting** From -12dB to 34.5dB in 1.5dB step size. 0x00 is lowest gain setting at -12dB and 0x1F is largest gain at 34.5dB.(0x8 is 0 dB)

5.20 Analog Functional Blocks

5.20.1 Overview

The ISD91500 contains an analog functional blocks that facilitate audio processing. These blocks are controlled by registers in the analog block address space. This section describes these functions and registers.

5.20.2 Features

- VMIDH / VMIDL reference voltage generation.
  - Pull down discharge controllable.
  - High/Low impedance pullup charge selectable.
- Microphone Bias generator.
  - Optional 8 level output voltage(50%, 60%, 75%, 90% of VCCA , 2.4V,1.7V,.2.0V, 1.2V)

5.20.3 VMID Reference Voltage Generation

The analog path and blocks require a low noise, mid-rail, Voltage reference for operation, the VMIDH and VMIDL generation block provides this. Control of this block allows user to power down the block, select its power down condition and control over the reference impedance. The block consists of a switchable resistive divider connected to the device VMIDH or VMIDL pin. A 4.7µF capacitor should be placed on this pin and returned to analog ground (VSSA) as shown in Figure 5.20-1.

Before using the MICBIAS, DAC or other analog blocks, the VMIDH reference needs to be enabled. Before using the SDADC, the VMIDL reference needs to be enabled.A low impedance option allows fast charging of the external noise de-coupling capacitor, while a higher impedance options provides lower power consumption. A pulldown option allows the reference to be discharged when off.

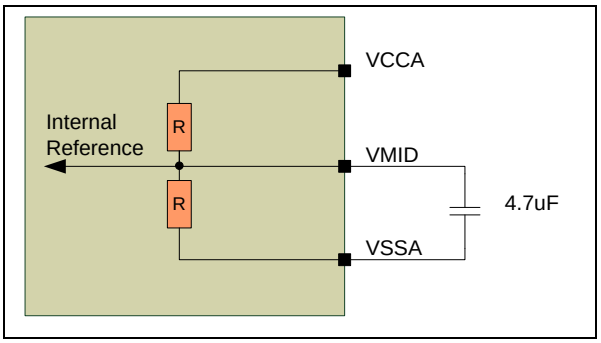


Figure 5.20-1 VMIDH/L Reference Generation

5.20.4 Microphone Bias Generator

The ISD91500 provides a microphone bias generator (MICBIAS) for improved recording quality. The MICBIAS can provide a maximum current of 1mA with a -60dB power supply rejection. The MICBIAS output voltage can be configured with ANA_MICBCTR[2:1] to select bias voltages from 50% to 90% of the VCCA supply voltage or fixed voltage output with 4 level (see description below). The user should consider the microphone manufacturers specification in deciding on the optimum MICBIAS voltage to use. Generally, a microphone will require a current of 0.1mA to a maximum 0.5mA and a voltage of 1V to 3V across it.

Referring to the application diagram of Figure 5.20-3, external resistor R_1 and R_2 values are selected to limit the current to a maximum that can be provided by MICBIAS; 1mA. On the ISD91500, the minimum total resistance $(R_1 + R_2)$ is 4Kohms. MICBIAS output voltage should be such that the following condition is met:

$$V_{MICBIAS} > V_S + (R_1 + R_2) \times I_{MIC}$$

where V_S is the desired voltage across the microphone from specification and I_{MIC} is the current through the microphone (0.1-0.5mA)

From Figure 5.20-3, MIC_IN1 and MIC_IN2 are AC coupled to the ISD91500 MIC+ and MIC- respectively for differential inputs. In single-ended operation, MIC_IN1 should go to MIC+ or MIC- of the ISD91500. C_1 and C_2 are AC coupling capacitors. In single-ended application, R_2 can be removed and R_1 increased to at least 4Kohms. For improved performance, it is recommended to keep R_2 to provide additional rejection from ground noise.

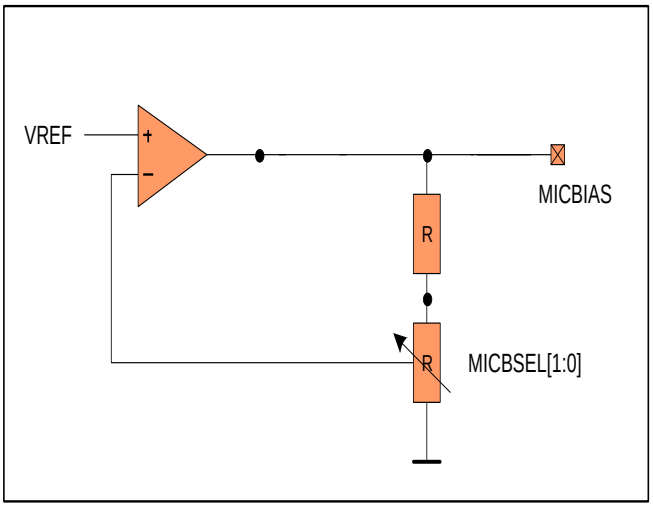


Figure 5.20-2 MICBIAS Block Diagram

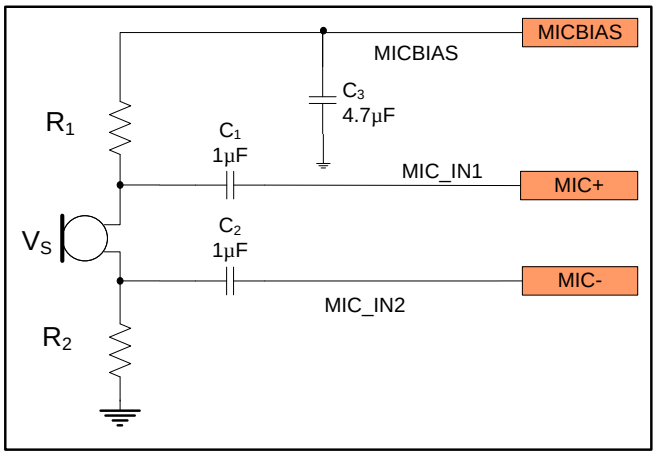


Figure 5.20-3 MICBIAS Application Diagram

5.20.5 Analog Function Control Register Map

R: read only, **W**: write only, **R/W**: read/write

Register	Offset	R/W	Description	Reset Value
ANA Base Address: **ANA_BA = 0x4008_0000**				
ANA_VMID	ANA_BA+0x00	R/W	VMID Reference Control Register	0x0000_0077
ANA_MICBCTR	ANA_BA+0x04	R/W	Microphone Bias Control Register	0x0000_0000

5.20.6 Analog Function Control Register Description

VMID Control Register (ANA_VMID)

Register	Offset	R/W	Description	Reset Value
ANA_VMID	ANA_BA+0x00	R/W	VMID Reference Control Register	0x0000_0077

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**VMIDLRH**	**VMIDLRL**	**VMIDLPD**	**Reserved**	**VMIDHRH**	**VMIDHRL**	**VMIDHPD**

Bits	Description	
[31:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**VMIDLRH**	**Power Down VMIDL High Resistance Reference** 0= Connect the High Resistance reference to VMIDL. Use this setting for minimum power consumption. 1= The High Resistance reference is disconnected from VMIDL. Default power down and reset condition.
[5]	**VMIDLRL**	**Power Down VMIDL Low Resistance Reference** 0= Connect the Low Resistance reference to VMIDL. Use this setting for fast power up of VMIDH. Can be turned off after 50ms to save power. 1= The Low Resistance reference is disconnected from VMIDL. Default power down and reset condition.
[4]	**VMIDLPD**	**VMIDH Pulldown** 0= Release VMIDL pin for reference operation. 1= Pull VMIDL pin to ground. Default power down and reset condition.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[2]	**VMIDHRH**	**Power Down VMIDH High Resistance Reference** 0= Connect the High Resistance reference to VMIDH. Use this setting for minimum power consumption. 1= The High Resistance reference is disconnected from VMIDH. Default power down and reset condition.
[1]	**VMIDHRL**	**Power Down VMIDH Low Resistance Reference** 0= Connect the Low Resistance reference to VMIDH. Use this setting for fast power up of VMIDH. Can be turned off after 50ms to save power. 1= The Low Resistance reference is disconnected from VMIDH. Default power down and reset condition.
[0]	**VMIDHPD**	**VMIDH Pulldown** 0= Release VMIDH pin for reference operation. 1= Pull VMIDH pin to ground. Default power down and reset condition.

Note1: VMIDH is mainly reference volatage for DAC , HP driver and MICBIAS.

Note2: VMIDL is mainly reference volatage for Boost , PGA and SDADC.

Microphone Bias Control (ANA_MICBCTR)

Register	Offset	R/W	Description	Reset Value
ANA_MICBCTR	ANA_BA+0x04	R/W	Microphone Bias Control Register	0x0000_0000

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

Bits	Description	
[31:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3]	**MICBMODE**	**Select MICBIAS Generator Mode** This bit can select MICBIAS as a fixed output or ratio of VCCA. 0= MICBIAS output ratio of VCCA 1= MICBIAS output fixed DC voltage
[2:1]	**MICBVSEL**	**Select Microphone Bias Voltage** MICBMODE = 0 0: 90% VCCA 1: 65% VCCA 2: 75% VCCA 3: 50% VCCA MICBMODE = 1 0: 2.4V 1: 1.7V 2: 2.0V 3: 1.2V

[0]	**MICBEN**	**MICBIAS enable** 0: Disable MICBIAS 1: Enable MICBIAS

5.21 Biquad Filter (BIQ)

5.21.1 Overview

A coefficient programmable 6-stage Biquad filter (12th-Order IIR filter) is available which can be used on either SDADC path to further reduce unwanted noise or filter the signal.

5.21.2 Features

- Support 6 Band Biquad filter
- Support SDADC path.
- 3.16 format

5.21.3 Funtion Description

Each biquad filter has the transfer function as ***H(z)*** and is implemented in Direct Form II Transpose structure as.

$$H(z) = \frac{\mathrm{b}_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + \mathrm{a}_1 \mathrm{z}^{-1} + \mathrm{a}_2 \mathrm{z}^{-2}}$$

Upon power on reset or when the BIQ_CTL.DLCOEFF =1 is released, a set of default coefficients b_{n0}, b_{n1}, b_{n2}, a_{n1}, a_{n2} (n = 1,2,3 which is the stage number of the filter) will be written to the coefficient RAM automatically. And these coefficients can be over-written by the processor for different filter specifications.

Note that the fixed point coefficients have the format of 3.16 (19 bits) and are stored in the coefficient RAM under normal operation. It takes 32 internal system clocks for the automatic write to finish when the BIQ_CTL.DLCOEFF bit is released; it is important that the processor has enough delay before start the coefficient programming or enabling biquad (BIQ_CTL.EN). Attempting to program the coefficients before the auto programming is done will result in unsuccessful programming. The default coefficient setting is a low pass filter with 3db cut-off frequency with 16KHz Fs (Sample Rate) and 256 OSR

Biquad is released from reset by setting BIQ_CTL.DLCOEFF =1. After 32 clock cycles, processor can setup other Biquad parameters or re-program coefficients before enabling filter.

The BIQ can down sample the data rate by programming BIQ_CTL.SDADCWNSR, register which has default value at 1.

The BIQ filter is in reset state in default. To use the BIQ function, the following sequence is recommended:

1. Set BIQ_CTL.DLCOEFF bit. By releasing the reset, the filter controller will download default coefficients automatically to the RAM.
2. Turn on the BIQ_CTL.PRGCOEFF bit if intending to change the coefficients. Otherwise skip to next step.
3. Turn off PRGCOEFF bit (if it was turned on in step #2).
4. Setup BIQ stage (BIQ->BIQ_CTL.STAGE), set “1” BIQ with 5 stages, set “0” BIQ with 6 stages.
5. Setup 6th stage filter on or off (BIQ->BIQ_CTL.SIXTHFON. If 6th stage filter is ON, the BIQ Stage automatically set 6 stages).
6. Turn on BIQ_CTL.EN. BIQ will start filter function.

Configuring Coefficient

1. BIQ function work on 6 stages, set BIQ->BIQ_CTL.STAGE=0, BIQ function will call 30 coefficients from BIQ_BA+0x0 ~0x074
2. BIQ function work on 5 stages, set BIQ->BIQ_CTL.STAGE=1, BIQ function will call 25 coefficients from BIQ_BA+0x00 ~0x060
3. If 6th stage filter is on (BIQ->BIQ_CTL.SIXTHFON=1), BIQ function automatically set 6 stages. BIQ function will call 30 coefficients.
4. BIQ function work on one stage, set first stage coefficient and bypass other stages set b0 = 0x1000, b1 =0,b2=0,a1 =0,a2=0.
5. BIQ function work on two stages, set first and second stages coefficient and bypass other stages set b0 = 0x1000, b1 =0,b2=0,a1 =0,a2=0.
6. BIQ function work on three stages, set first, second and third stages coefficient and bypass other stages set b0 = 0x1000, b1 =0,b2=0,a1 =0,a2=0
7. BIQ function work on four stages, set first, second, third and fourth stages coefficient and bypass other stages set b0 = 0x1000, b1 =0,b2=0,a1 =0,a2=0

5.21.4 Biquad Filter Control Register Map

R: read only, **W**: write only, **R/W**: read/write

Register	Offset	R/W	Description	Reset Value
BIQ Base Address: **BIQ_BA = 0x400B_0000**				
BIQ_COEFF0	BIQ_BA+0x00	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF1	BIQ_BA+0x004	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF2	BIQ_BA+0x008	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF3	BIQ_BA+0x00c	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF4	BIQ_BA+0x010	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF5	BIQ_BA + 0x14	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF6	BIQ_BA+0x018	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF7	BIQ_BA+0x01c	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF8	BIQ_BA+0x020	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF9	BIQ_BA+0x024	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF10	BIQ_BA + 0x28	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF11	BIQ_BA+0x02c	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF12	BIQ_BA+0x030	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF13	BIQ_BA+0x034	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF14	BIQ_BA+0x038	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000

BIQ_COEFF15	BIQ_BA + 0x3c	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF16	BIQ_BA+0x040	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF17	BIQ_BA+0x044	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF18	BIQ_BA+0x048	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF19	BIQ_BA+0x04c	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF20	BIQ_BA + 0x50	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF21	BIQ_BA+0x054	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF22	BIQ_BA+0x058	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF23	BIQ_BA+0x05c	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF24	BIQ_BA+0x060	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF25	BIQ_BA + 0x64	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF26	BIQ_BA+0x068	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF27	BIQ_BA+0x06c	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF28	BIQ_BA+0x070	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF29	BIQ_BA+0x074	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000
BIQ_CTL	BIQ_BA+0x080	R/W	BIQ Control Register	0x0000_0110

5.21.5 Biquad Filter Control Register Description

BIQ Coefficient Register (BIQ_COEFFn)

Register	Offset	R/W	Description	Reset Value
BIQ_COEFF0	BIQ_BA+0x00	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF1	BIQ_BA+0x004	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF2	BIQ_BA+0x008	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF3	BIQ_BA+0x00c	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF4	BIQ_BA+0x010	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF5	BIQ_BA + 0x14	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF6	BIQ_BA+0x018	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF7	BIQ_BA+0x01c	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF8	BIQ_BA+0x020	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF9	BIQ_BA+0x024	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF10	BIQ_BA + 0x28	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF11	BIQ_BA+0x02c	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF12	BIQ_BA+0x030	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF13	BIQ_BA+0x034	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF14	BIQ_BA+0x038	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF15	BIQ_BA + 0x3c	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000

BIQ_COEFF16	BIQ_BA+0x040	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF17	BIQ_BA+0x044	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF18	BIQ_BA+0x048	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF19	BIQ_BA+0x04c	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 4st stage BIQ Coefficients	0x0000_0000
BIQ_COEFF20	BIQ_BA + 0x50	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF21	BIQ_BA+0x054	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF22	BIQ_BA+0x058	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF23	BIQ_BA+0x05c	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF24	BIQ_BA+0x060	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 5nd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF25	BIQ_BA + 0x64	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF26	BIQ_BA+0x068	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF27	BIQ_BA+0x06c	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF28	BIQ_BA+0x070	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000
BIQ_COEFF29	BIQ_BA+0x074	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 6rd stage BIQ Coefficients	0x0000_0000

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

Bits	Description	
[31:0]	**COEFFDAT**	**Coefficient Data**

BIQ Control Register (BIQ_CTL)

Register	Offset	R/W	Description	Reset Value
BIQ_CTL	BIQ_BA+0x080	R/W	BIQ Control Register	0x0000_0110

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**STAGE**	**Reserved**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRGCOEFF	**SDADCWNSR**			**DLCOEFF**	**Reserved**	**SIXTHFON**	**BIQEN**

Bits	Description	
[31:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11]	**STAGE**	**BIQ Stage Number Control** 0 = 6 stage. 1 = 5 stage. **Note :** When set as 5 stage, 6th stage on/off is control by **SIXTHFON** .
[10:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**PRGCOEFF**	**Programming Mode Coefficient Control Bit** 0 = Coefficient RAM is in normal mode. 1 = coefficient RAM is under programming mode. This bit must be turned off when BIQEN in on.
[6:4]	**SDADCWNSR**	**SDADC Down Sample** 001 --- 1x (no down sample) 010 --- 2x 011 --- 3x 100 --- 4x 110 --- 6x Others reserved

[3]	**DLCOEFF**	**Move BIQ Out of Reset State** 0 = BIQ filter is in reset state. 1 = When this bit is on, the default coefficients will be downloaded to the coefficient ram automatically in 32 internal system clocks. Processor must delay enough time before changing the coefficients or turn the BIQ on.
[2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**SIXTHFON**	**6th stage Filter On/Off Control** 0 = disable 6th stage filter. 1 = enable 6th stage filter. **Note1 :** This register only work when STAGE set 5 stage. **Note2 :** SDADC path sixth stage coefficient is for this filter coefficient.
[0]	**BIQEN**	**BIQ Filter Start to Run** 0 = BIQ filter is not processing. 1 = BIQ filter is on.

6 REVISION HISTORY

VERSION	DATE	DESCRIPTION	NOTE
1.0	Jul 1, 2021	Formal version release	
1.1	Jul 30, 2021	Add note for cache in FMC chapter	
2.0	Oct 6, 2021	Update to V2.0 for RevC 1.Update pin diagram & pin description chapter 2.Update USB chapter 3.Update DAC chapter for power on/off HP sequence	
2.1	Mar 22, 2022	Update feature for BOD & LVR description Update pin diagram & pin description Update DAC function description chapter	
2.2	Aug 18, 2022	Update format Update General Description	

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.