ISD ARM® Cortex®-M0 SoC
ISD91200 Series
Technical Reference Manual

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 ISD ARM® Cortex®-M0 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.

Release Date: Mar. 4, 2023

Rev 2.9

Table of Contents

List of Figures

List of Tables

1 GENERAL DESCRIPTION

The ISD91200 series is a system-on-chip product optimized for low power, audio record and playback with an embedded ARM® Cortex™-M0 32-bit microcontroller core.

The ISD91200 device embeds a Cortex™-M0 core running up to 50 MHz with 64K/128Kbyte of non-volatile flash memory and 12K-byte of embedded SRAM. It also comes equipped with a variety of peripheral devices, such as Timers, Watchdog Timer (WDT), Real-time Clock (RTC), Peripheral Direct Memory Access (PDMA), a variety of serial interfaces (UART, SPI/SSP, I^2C, I^2S), PWM modulators, GPIO, LDO, SDADC, SARADC, DPWM, Low Voltage Detector and Brown-out detector.

The ISD91200 comes equipped with a rich set of power saving modes including a Deep Power Down (DPD) mode drawing less than 1μA. A micro-power 10KHz oscillator can periodically wake up the device from deep power down to check for other events. A Standby Power Down (SPD) mode can maintain a real time clock function at less than 5 μA.

For audio functionality the ISD91200 includes a Sigma-Delta ADC with 91dB SNR performance coupled with a Programmable Gain Amplifier (PGA with 0-6/12dB gain) and volume control (36dB to -108dB) in digital domain to enable direct connection of a microphone. Audio output is provided by a Differential Class D amplifier (DPWM) that can deliver 0.5W of power to an 8Ω speaker.

The ISD91200 provides 16 analog enabled general purpose IO pins (GPIO). These pins can be configured to connect to an analog comparator, can be configured as analog current sources or can be routed to the SDADC for analog conversion. They can also be used as a relaxation oscillator to perform capacitive touch sensing.

2 FEATURES

- Core
  - ARM® Cortex™-M0 core running up to 50 MHz for normal speed.
  - One 24-bit System tick timer for operating system support.
  - Supports a variety of low power sleep and power down modes.
  - Single-cycle 32-bit hardware multiplier.
  - NVIC (Nested Vector Interrupt Controller) for 32 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 1.8V to 5.5V.
  - Power management Unit (PMU) providing four levels of power controls.
  - Deep Power Down (DPD) mode with sub micro-amp leakage (<2µA).
  - Wakeup from Deep Power Down via dedicated WAKEUP pin or timed operation from internal low power 10KHz oscillator.
  - Standby current in SPD mode with limited RAM (256byte SBRAM) retention and RTC operation <5µA.
  - Standby current in STOP mode with full SRAM retention <10 µA.
  - Wakeup from Standby can be from any GPIO interrupt, RTC or BOD.
  - Sleep mode with minimal dynamic power consumption.
  - 3V LDO for operation of external 3V devices such as serial flash.
- Flash EPROM Memory
  - 64K/128K bytes Flash EPROM for program code and data storage.
  - Mini-cache to maintain near zero-wait state memory access.
  - Support In-system program (ISP) and In-circuit program (ICP) application code update
  - 512 byte page erase for flash.
  - Configurable boundary to delineate code and data flash.
  - Support 2 wire In-circuit Programming (ICP) update from SWD ICE interface
- SRAM Memory
  - 12K bytes embedded SRAM.
- Clock Control
  - High speed and low speed oscillators providing flexible selection for different applications. No external components necessary.
  - Built-in trimmable oscillator, factory trimmed HIRC within 1% to settings of 49.152MHz and 32.768MHz. User trimmable with in-built frequency measurement block (OSCFM) using reference clock of 32kHz crystal or external reference source.
  - Ultra-low power (<1uA) 10KHz oscillator for watchdog and wakeup from power-down or sleep operation.
  - External 32kHz crystal input for RTC function and low power system operation.
  - External 12 MHz crystal input for precise timing operation.
- GPIO
  - Four I/O modes:
    - ◆ Quasi bi-direction
    - ◆ Push-Pull output
    - ◆ Open-Drain output
    - ◆ Input only with high impendence
  - Schmitt trigger input selectable.
  - I/O pin can be configured as interrupt source with edge/level setting.
  - Supports Driver and Sink IO mode.
  - Capacitive Touch: 16

- Maximal 32 GPIO

- Audio Analog to Digital converter (no function in ISD91200B series)
  - Sigma Delta ADC with configurable decimation filter and 16 bit output.
  - 90dB Signal-to-Noise (SNR) performance.
  - Programmable gain amplifier with 32 steps from -12 to 35.25dB in 0.75dB steps.
  - Boost gain stage of 26dB, giving maximum total gain of 61dB.
  - Input selectable from dedicated MIC pins or analog enabled GPIO.
  - Programmable biquad filter to support multiple sample rates from 8-32kHz.
  - DMA support for minimal CPU intervention.
- Bridge Sense ADC (only supports in ISD91200B series)
  - On chip calibaration
  - 8 steps Programmable Gain Amplifier (BSPGA)
  - Programmable data output rate
  - 21 bit precision @ moving average 6.4 SPS
- Differential Audio PWM Output (DPWM)
  - Direct connection of speaker
  - 0.5W drive capability into 8Ω load.
  - Configurable up-sampling to support sample rates from 8-48kHz.
  - Programmable volume control from -108dB to +36dB in 0.5 dB step
  - Programmable biquad filter to support multiple sample rates from 8-48kHz
  - DMA support for minimal CPU intervention.
- Timers
  - Two timers with 8-bit pre-scaler and 24-bit resolution.
  - Counter auto reload.
- Watch Dog Timer
  - Default ON/OFF by configuration setting
  - Multiple clock sources
  - 8 selectable time out period from micro seconds to seconds (depending on clock source)
  - WDT can wake up power down/sleep.
  - Interrupt or reset selectable on watchdog time-out.
- RTC
  - Real Time Clock counter (second, minute, hour) and calendar counter (day, month, year)
  - Alarm registers (second, minute, hour, day, month, year)
  - Selectable 12-hour or 24-hour mode
  - Automatic leap year recognition
  - Time tick and alarm interrupts.
  - Device wake up function.
  - Supports software compensation of crystal frequency by compensation register (FCR)
- PWM/Capture
  - Four 16-bit PWM generators provide four single ended PWM outputs or two complementary paired PWM outputs.
  - The PWM generator equipped with a clock source selector, a clock divider, an 8-bit pre-scaler and Dead-Zone generator for complementary paired PWM.
  - PWM interrupt synchronous to PWM period.
  - 16-bit digital Capture timers (shared with PWM timers) provide rising/falling capture inputs.
  - Support Capture interrupt
- UART
  - Up to two uart controllers

  - UART ports with flow control (TX, RX, CTS and RTS)
  - 8-byte FIFO.
  - Support IrDA (SIR) and LIN function
  - Programmable baud-rate generator up to 1/16 of system clock.

- SPI
  - Up to two SPI controllers
  - SPI Clock up to 25 MHz.
  - SPI data rate in Quad mode up to 100 Mbps
  - Support MICROWIRE/SPI master/slave mode (SSP)
  - Full duplex synchronous serial data transfer
  - Variable length of transfer data from 1 to 4 bytes
  - MSB or LSB first data transfer
  - 2 slave/device select lines when used in master mode.
  - DMA support.(64bit (16x4) data FIFO)
  - Quad/Dual SPI support.

- I2C
  - 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 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.
  - Programmable clock allowing versatile rate control.
  - Supports multiple address recognition (four slave address with mask option)
  - Supports wake-up by address recognition (for 1st slave address only)

- I^{2}S
  - Interface with external audio CODEC.
  - Operate as either master or slave.
  - Capable of handling 8, 16, 24 and 32 bit word sizes
  - Mono and stereo audio data supported
  - I^{2}S and MSB justified data format supported
  - Two 8 word FIFO data buffers are provided, one for transmit and one for receive
  - Generates interrupt requests when buffer levels cross a programmable boundary
  - Supports DMA requests, for transmit and receive

- SARADC
  - 12-bit SAR ADC with 700K SPS
  - Up to 12-ch single-end input
  - Single scan/single cycle scan/continuous scan
  - Each channel with individual result register
  - Scan on enabled channels
  - Threshold voltage detection
  - Conversion start by software programming or external input
  - Supports PDMA mode

- Low Voltage Reset
  - Threshold voltage typical level: 1.6V

- Brown-out detector

- Supports 16-level brown-out setting.
- Supports time-multiplex operation to minimize power consumption.
- Supports Brownout Interrupt and Reset option

- Built in Low Dropout Voltage Regulator (LDO)
  - Capable of delivering 30mA load current.
  - Configurable 8 output voltage selections from 1.5V – 3.3V.
  - LDO output powers IO ring for GPIOA<7:0> and can supply power to external SPI Flash.
  - Can be bypassed and voltage domain supplied directly from system power.
- Additional Features
  - Digital Microphone interface.
- Standby current in STOP mode with SRAM retention <= 10µA at 25°C.
- Operating Temperature: -40C~85C
- Package:
  - LQFP64
  - QFN32
  - Package is Halogen-free, RoHS-compliant and TSCA-compliant

3 PART INFORMATION AND PIN CONFIGURATION

3.1 Pin Configuration

3.1.1 ISD91200 LQFP 64 pin (Non Bridge Sense)

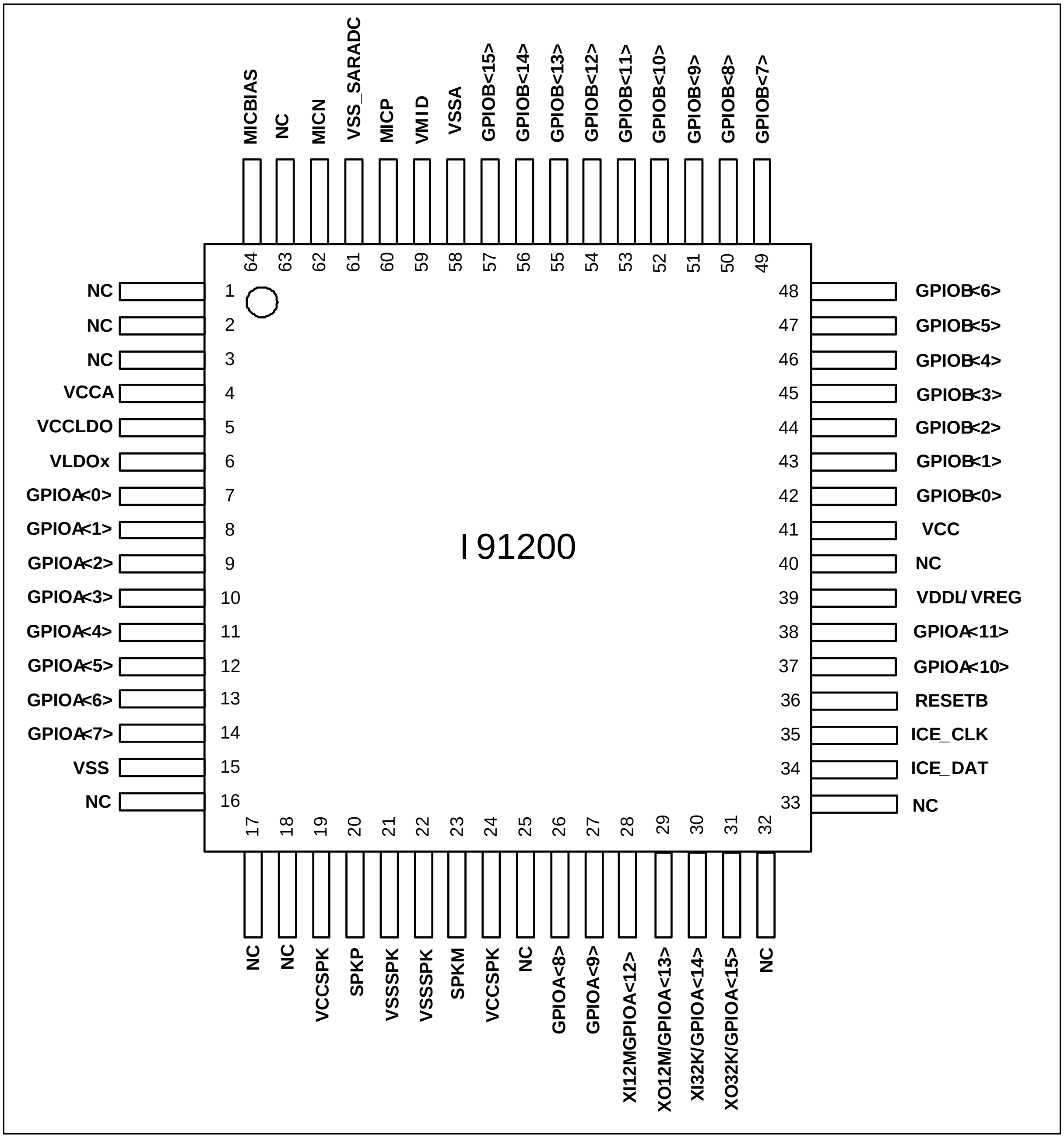


Figure 3-1 ISD91200 LQFP 64 pin

Note:

1. No MIC function in ISD91200P series
2. No MIC & SPK function in ISD91200G series

3.1.2 ISD91200B LQFP 64 pin (Bridge Sense series)

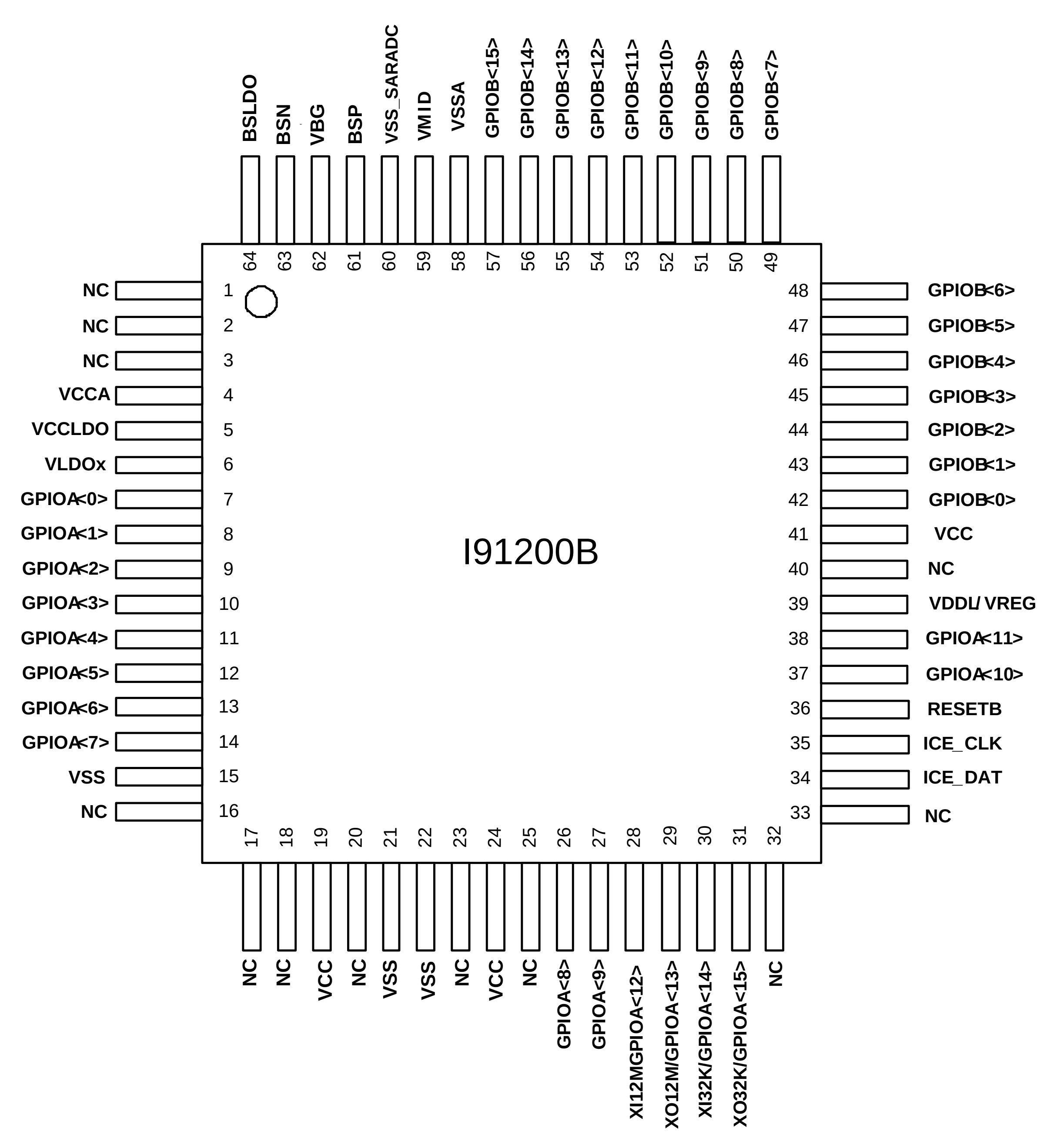


Figure 3-2 ISD91200B LQFP 64 pin

3.1.3 ISD91200 QFN 32 pin (Non Bridge Sense series)

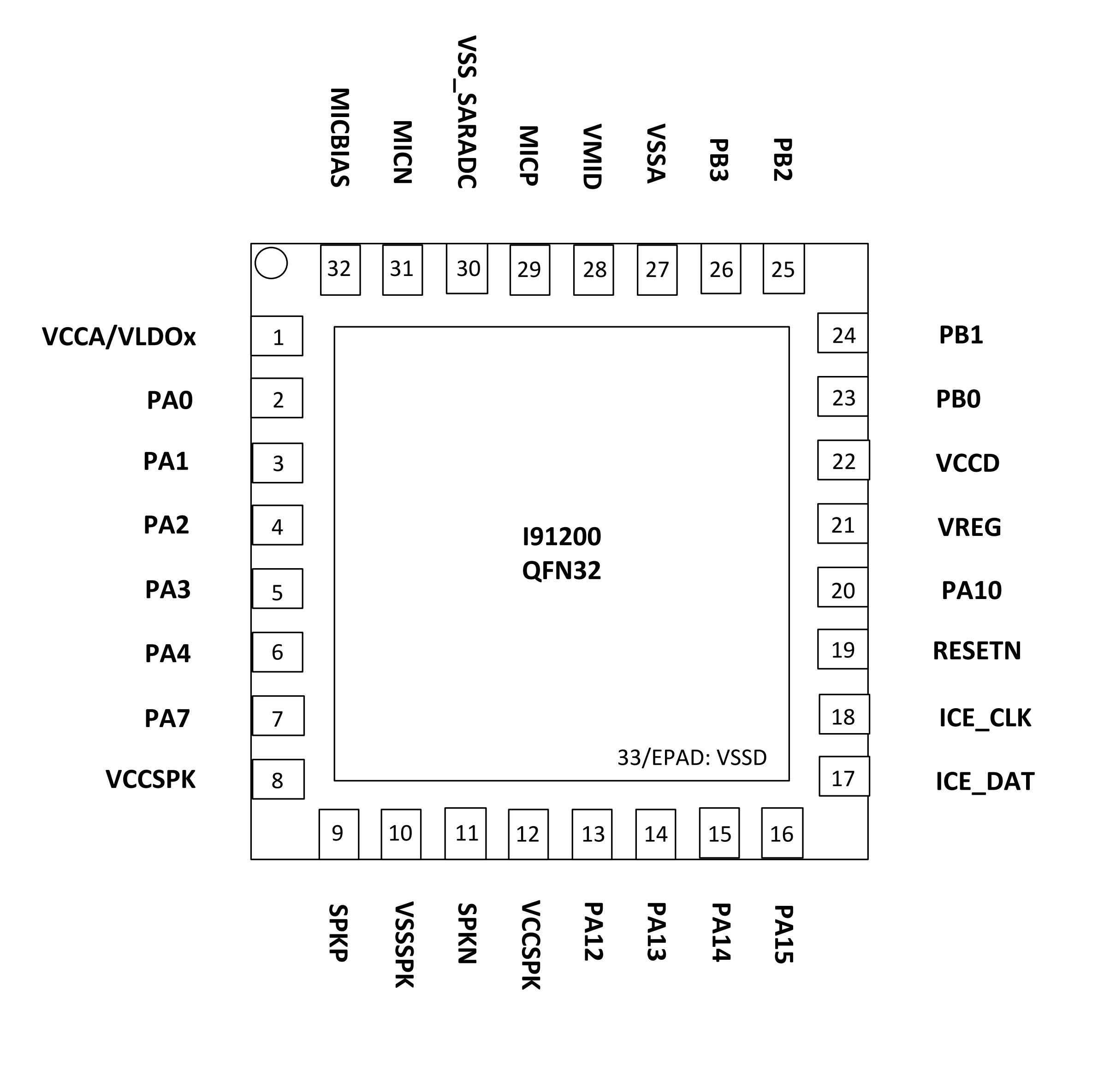


Figure 3-3 ISD91200 QFN 32 pin

Note:

1. No MIC function in ISD91200P series
2. No MIC & SPK function in ISD91200G series

3.2 Pin Description

The ISD91200 is a low pin count device where many pins are configurable to alternative functions. All General Purpose Input/Output (GPIO) pins can be configured to alternate functions as described in the table below.

3.2.1 ISD91200RI/CRI/PRI/GRI sereies (Normal Series)

Pin No.	Pin Name	Pin Type	Alt CFG	Description
LQFP 64				
1	NC			
2	NC			
3	NC			
4	VCCA	**P**		Analog power supply.
5	VCCLDO	**P**		Power supply for LDO, should be connected to VCCD.
6	VLDOx	**P**		LDO regulator output. if used, a 1µF decoupling capacitor must be placed. If not used then tie to VCCD.
7	PA.0	**I/O**	0	General purpose input/output pin; Port A, bit 0
	SPI0_MISO1	**O**	1	2nd Master In, Slave Out for SPI0 interface
	SPI0_SSB1	**O**	2	Slave Select Bar 1 for SPI0 interface
	I2S0_FS	**I/O**	3	Frame Sync Clock for I2S interface
8	PA.1	I/O	0	General purpose input/output pin; Port A, bit 1
	SPI0_MOSI0	**O**	1	1st Master Out, Slave In for SPI0 interface
	I2S0_BCLK	**I/O**	3	Bit Clock for I2S interface
9	PA.2	**I/O**	0	General purpose input/output pin; Port A, bit 2
	SPI0_SCLK0	**I/O**	1	Serial Clock for SPI 0interface
	DMIC_DAT	**I**	2	DMIC data
	I2S0_SDI	**I**	3	Serial Data Input for I2S interface
10	PA.3	**I/O**	0	General purpose input/output pin; Port A, bit 3
	SPI0_SSB0	**O**	1	Slave Select Bar 0 for SPI0 interface
	SARADC_TRIG	**I**	2	SARADC Trigger
	I2S0_SDO	**O**	3	Serial Data Output for I2S interface
11	PA.4	**I/O**	0	General purpose input/output pin; Port A, bit 4
	SPI0_MISO0	**O**	1	Master In, Slave Out channel 0 for SPI interface
	UART0_TX	**O**	2	Transmit channel of UART 0
	SPI1_MOSI	**O**	3	Master Out, Slave In for SPI1 interface
12	PA.5	**I/O**	0	General purpose input/output pin; Port A, bit 5
	SPI0_MOSI1	**O**	1	2nd Master Out, Slave In for SPI0 interface
	UART0_RX	**I**	2	Receive channel of UART 0
	SP1_SCLK	**I/O**	3	Serial Clock for SPI1 interface

Pin No.	Pin Name	Pin Type	Alt CFG	Description
LQFP 64				
13	PA.6	**I/O**	0	General purpose input/output pin; Port A, bit 6
	UART0_TX	**O**	1	Transmit channel of UART 0
	I2C0_SDA	**I/O**	2	Serial Data, I2C interface
	SPI1_SSB	**O**	3	Slave Select Bar for SPI1 interface
14	PA.7	**I/O**	0	General purpose input/output pin; Port A, bit 7
	UART0_RX	**I**	1	Receive channel of UART 0
	I2C0_SCL	**I/O**	2	Serial Clock, I2C interface
	SPI1_MISO	**I**	3	Master In, Slave Out for SPI1 interface
15	VSSD	**P**		Digital Ground.
16	NC			
17	NC			
18	NC			
19	VCCSPK	**P**		Power Supply for PWM Speaker Driver
20	SPKP	**O**		Positive Speaker Driver Output
21	VSSSPK	**P**		Ground for PWM Speaker Driver
22	VSSSPK	**P**		Ground for PWM Speaker Driver
23	SPKN	**O**		Positive Speaker Driver Output
24	VCCSPK	**P**		Power Supply for PWM Speaker Driver
25	NC			
26	PA.8	**I/O**	0	General purpose input/output pin; Port A, bit 8
	I2C0_SDA	**I/O**	1	Serial Data, I2C interface
	UART1_TX	**O**	2	Transmit channel of UART 1
	UART0_RTS	**O**	3	UART 0 Request to Send Output.
27	PA.9	**I/O**	0	General purpose input/output pin; Port A, bit 9
	I2C0_SCL	**I/O**	1	Serial Clock, I2C interface
	UART1_RX	**I**	2	Receive channel of UART 1
	UART0_CTS	**I**	3	UART 0 Clear to Send Input.
28	I2C0_SCL	**I/O**	3	Serial Clock, I2C interface
	PA.12	**I/O**	0	General purpose input/output pin; Port A, bit 12
	PWM0CH2	**O**	1	PWM0CH2 Output.
	XI12M	**I**	2	12MHz Crystal Oscillator Input. Max Voltage 1.8V
29	PA.13	**I/O**	0	General purpose input/output pin; Port A, bit 13
	PWM0CH3	**O**	1	PWM0CH3 Output.
	XO12M	**O**	2	12MHz Crystal Oscillator Output
	I2C0_SCL	**I/O**	3	Serial Clock, I2C interface

Pin No.	Pin Name	Pin Type	Alt CFG	Description
LQFP 64				
30	PA.14	I/O	0	General purpose input/output pin; Port A, bit 14
	UART1_TX	O	1	Transmit channel of UART 1
	DMIC_CLK	IO	2	DMIC clock
	XI32K	I	3	32.768kHz Crystal Oscillator Input. Max Voltage 1.8V
31	PA.15	I/O	0	General purpose input/output pin; Port A, bit 15
	UART1_RX	I	1	Receive channel of UART 1
	MCLK	O	2	Master clock output for synchronizing external device
	XO32K	O	3	32.768kHz Crystal Oscillator Output
32	NC			
33	NC			
34	ICE_DAT	I/O		Serial Wire Debug port clock pin. Has internal weak pull-up.
35	ICE_CLK	O		Serial Wire Debug port data pin. Has internal weak pull-up.
36	RESETN	I		External reset input. Pull this pin low to reset device to initial state. Has internal weak pull-up.
37	PA.10	I/O	0	General purpose input/output pin; Port A, bit 10
	PWM0CH0	O	1	PWM0CH0 Output.
	TM0	I	2	External input to Timer 0
	DPWM_P	O	3	Audio PWM positive
38	PA.11	I/O	0	General purpose input/output pin; Port A, bit 11
	PWM0CH1	O	1	PWM0CH1 Output.
	TM1	I	2	External input to Timer 1
	DPWM_N	O	3	Audio PWM negative
39	VREG	P		Logic regulator output decoupling pin. A 1µF capacitor returning to VSSD must be placed on this pin.
40	NC			
41	VCCD	P		Main Digital Supply for Chip. Supplies all IO except analog, Speaker Driver
42	PB.0	I/O	0	General purpose input/output pin, analog capable; Port B, bit 0. Triggers external interrupt 0 (EINT0/IRQ2)
	SPI1_MOSI	O	1	Master Out, Slave In for SPI1 interface
	CS0	AI		Touch scan channel 0
	A0P	AI		Operational Amplifier 0 positive input
43	PB.1	I/O	0	General purpose input/output pin, analog capable; Port B, bit 1. Triggers external interrupt 1 (EINT1/IRQ3)
	SP1_SCLK	I/O	1	Serial Clock for SPI1 interface
	CS1	AI		Touch scan channel 1
	A0N	AI		Operational Amplifier 0 negative input

Pin No.	Pin Name	Pin Type	Alt CFG	Description
LQFP 64				
44	PB.2	I/O	0	General purpose input/output pin, analog capable; Port B, bit 2
	SPI1_SSB	O	1	Slave Select Bar for SPI1 interface
	CS2	AI		Touch scan channel 1
	A0E	AO		Operational Amplifier 0 output
	SAR11	AI		SARADC channel 11
45	PB.3	I/O	0	General purpose input/output pin, analog capable; Port B, bit 3
	SPI1_MISO	I	1	Master In, Slave Out for SPI1 interface
	CS3	AI		Touch scan channel 3
	A1P	AI		Operational Amplifier 1 positive input
46	PB.4	I/O	0	General purpose input/output pin, analog capable; Port B, bit 4
	I2S0_FS	I/O	1	Frame Sync Clock for I2S interface
	CS4	AI		Touch scan channel 4
	A1N	AO		Operational Amplifier 1 negative input
47	PB.5	I/O	0	General purpose input/output pin, analog capable; Port B, bit 5
	I2S0_BCLK	I/O	1	Bit Clock for I2S interface
	CS5	AI		Touch scan channel 5
	A1E	AO		Operational Amplifier 1 output
	SAR10	AI		SARADC channel 10
48	PB.6	I/O	0	General purpose input/output pin, analog capable; Port B, bit 6
	I2S0_SDI	I	1	Serial Data Input for I2S interface
	CS6	AI		Touch scan channel 6
	CNP	AI		Comparator 1 positive input and comparator 2 negtive input
	SAR8	AI		SARADC channel 8
49	PB.7	I/O	0	General purpose input/output pin, analog capable; Port B, bit 7
	I2S0_SDO	O	1	Serial Data Output for I2S interface
	CS7	AI		Touch scan channel 7
	C1N	AI		Comparator 1 negtive input
	SAR9	AI		SARADC channel 9
50	PB.8	I/O	0	General purpose input/output pin, analog capable; Port B, bit 8.
	I2C0_SDA	I/O	1	Serial Data, I2C interface
	I2S0_FS	I/O	2	Frame Sync Clock for I2S interface
	UART1_RTS	O	3	UART 1 Request to Send Output.
	CS8	AI		Touch scan channel 8
	C2P	AI		Comparator 2 positive input
	SAR0	AI		SARADC channel 0

Pin No.	Pin Name	Pin Type	Alt CFG	Description
LQFP 64				
51	PB.9	I/O	0	General purpose input/output pin, analog capable; Port B, bit 9.
	I2C0_SCL	I/O	1	Serial Clock, I2C interface
	I2S0_BCLK	I/O	2	Bit Clock for I2S interface
	UART1_CTS	I	3	UART 1 Clear to Send Input.
	CS9	AI		Touch scan channel 9
	SAR1	AI		SARADC channel 1
52	PB.10	I/O	0	General purpose input/output pin, analog capable; Port B, bit 10
	CMP1	O	1	Compare 1 Output
	I2S0_SDI	I	2	Serial Data Input for I2S interface
	UART1_TX	O	3	Transmit channel of UART 1
	CS10	AI		Touch scan channel 10
	SAR2	AI		SARADC channel 2
53	PB.11	I/O	0	General purpose input/output pin, analog capable; Port B, bit 11
	CMP2	O	1	Compare 2 Output
	I2S0_SDO	O	2	Serial Data Output for I2S interface
	UART1_RX	I	3	Receive channel of UART 1
	CS11	AI		Touch scan channel 11
	SAR3	AI		SARADC channel 3
54	PB.12	I/O	0	General purpose input/output pin, analog capable; Port B, bit 12
	SPI0_MISO0	I/O	1	1st Master In, Slave Out for SPI0 interface
	SPI1_MOSI	I/O	2	Master Out, Slave In for SPI1 interface
	DMIC_DAT	I	3	DMIC data
	CS12	AI		Touch scan channel 12
	SAR4	AI		SARADC channel 4
55	PB.13	I/O	0	General purpose input/output pin, analog capable; Port B, bit 13
	SPI0_MOSI0	I/O	1	1st Master Out, Slave In for SPI0 interface
	SPI1_SCLK	I/O	2	Serial Clock for SPI1 interface
	SARADC_TRIG	I	3	SARADC Trigger
	CS13	AI		Touch scan channel 13
	SAR5	AI		SARADC channel 5
56	PB.14	I/O	0	General purpose input/output pin, analog capable; Port B, bit 14
	SPI0_SCLK0	I/O	1	1st Serial Clock for SPI0 interface
	SPI1_SSB	O	2	Slave Select Bar 1 for SPI1 interface
	DMIC_CLK	O	3	DMIC clock
	CS14	AI		Touch scan channel 14
	SAR6	AI		SARADC channel 6

Pin No.	Pin Name	Pin Type	Alt CFG	Description
LQFP 64				
57	PB.15	**I/O**	0	General purpose input/output pin, analog capable; Port B, bit 15
	SPI0_SSB0	**O**	1	Slave Select Bar 0 for SPI0 interface
	SPI1_MISO	**I/O**	2	Master In, Slave Out for SPI1 interface
	MCLK	**O**	3	Master clock output for synchronizing external device
	CS15	**AI**		Touch scan channel 15
	SAR7	**AI**		SARADC channel 7
58	VSSA	**AP**		Ground for analog circuitry.
59	VMID	**O**		Mid rail reference. Connect 4.7µF to VSSA.
60	MICP	**AI**		SDADC positive input
61	VSS_SARADC	**AP**		SARADC ground
62	MICN	**AI**		SDADC negative input
63	NC			
64	MICBIAS	**O**		Microphone bias output.

Note:

1. Pin Type I=Digital Input, O=Digital Output; AI=Analog Input; AO=Analog Output; P=Power Pin; AP=Analog Power
2. No MIC function in ISD91200P series
3. No MIC & SPK function in ISD91200G series

3.2.2 ISD91200B sereies difference

60	VSS_SARADC	AP		SARADC & BSADC ground
61	BSP	AI		Bridge sense positive input
62	VBG	AP		BS band-gap output.
63	BSN	AI		Bridge sense negative input
64	BSLDO	AO		BS LDO output

Note:

1. Pin Type I=Digital Input, O=Digital Output; AI=Analog Input; AO=Analog Output; P=Power Pin; AP=Analog Power

3.2.3 ISD91200YI/ISD91200CYI (QFN32)

LQFP64 Pin Name		Pin Type	Alt CFG	Description
1	VCCA	P		Analog power supply.
	VLDOx	P		LDO power plane, internal bonded to VCCA. LDO is no used.
2	PA.0	I/O	0	General purpose input/output pin; Port A, bit 0
	SPI0_MISO1	O	1	2nd Master In, Slave Out for SPI0 interface
	SPI0_SSB1	O	2	Slave Select Bar 1 for SPI0 interface
	I2S0_FS	I/O	3	Frame Sync Clock for I2S interface
3	PA.1	I/O	0	General purpose input/output pin; Port A, bit 1
	SPI0_MOSI0	O	1	1st Master Out, Slave In for SPI0 interface
	I2S0_BCLK	I/O	3	Bit Clock for I2S interface
4	PA.2	I/O	0	General purpose input/output pin; Port A, bit 2
	SPI0_SCLK0	I/O	1	Serial Clock for SPI 0interface
	DMIC_DAT	I	2	DMIC data. Not support for G & P series.
	I2S0_SDI	I	3	Serial Data Input for I2S interface
5	PA.3	I/O	0	General purpose input/output pin; Port A, bit 3
	SPI0_SSB0	O	1	Slave Select Bar 0 for SPI0 interface
	SARADC_TRIG	I	2	SARADC Trigger
	I2S0_SDO	O	3	Serial Data Output for I2S interface
6	PA.4	I/O	0	General purpose input/output pin; Port A, bit 4
	SPI0_MISO0	O	1	Master In, Slave Out channel 0 for SPI interface
	UART0_TX	O	2	Transmit channel of UART 0
	SPI1_MOSI	O	3	Master Out, Slave In for SPI1 interface
7	PA.7	I/O	0	General purpose input/output pin; Port A, bit 7
	UART0_RX	I	1	Receive channel of UART 0
	I2C0_SCL	I/O	2	Serial Clock, I2C interface
	SPI1_MISO	I	3	Master In, Slave Out for SPI1 interface
8	VCCSPK	P		Power Supply for PWM Speaker Driver. Need VCC connection for G series.
9	SPKP	O		Positive Speaker Driver Output. Not support for G series.
10	VSSSPK	P		Ground for PWM Speaker Driver. Need VSS connection for G series.
11	SPKN	O		Negative Speaker Driver Output. Not support for G series.
12	VCCSPK	P		Power Supply for PWM Speaker Driver. Need VCC connection for G series.
13	I2C0_SCL	I/O	3	Serial Clock, I2C interface
	PA.12	I/O	0	General purpose input/output pin; Port A, bit 12
	PWM0CH2	O	1	PWM0CH2 Output.
	XI12M	I	2	12MHz Crystal Oscillator Input. Max Voltage 1.8V
14	PA.13	I/O	0	General purpose input/output pin; Port A, bit 13
	PWM0CH3	O	1	PWM0CH3 Output.
	XO12M	O	2	12MHz Crystal Oscillator Output
	I2C0_SCL	I/O	3	Serial Clock, I2C interface
15	PA.14	I/O	0	General purpose input/output pin; Port A, bit 14

LQFP64 Pin Name		Pin Type	Alt CFG	Description
	UART1_TX	O	1	Transmit channel of UART 1
	DMIC_CLK	IO	2	DMIC clock. Not support for G & P series.
	XI32K	I	3	32.768kHz Crystal Oscillator Input. Max Voltage 1.8V
16	PA.15	I/O	0	General purpose input/output pin; Port A, bit 15
	UART1_RX	I	1	Receive channel of UART 1
	MCLK	O	2	Master clock output for synchronizing external device
	XO32K	O	3	32.768kHz Crystal Oscillator Output
17	ICE_DAT	I/O		Serial Wire Debug port clock pin. Has internal weak pull-up.
18	ICE_CLK	O		Serial Wire Debug port data pin. Has internal weak pull-up.
19	RESETN	I		External reset input. Pull this pin low to reset device to initial state. Has internal weak pull-up.
20	PA.10	I/O	0	General purpose input/output pin; Port A, bit 10
	PWM0CH0	O	1	PWM0CH0 Output.
	TM0	I	2	External input to Timer 0
	DPWM_P	O	3	Audio PWM positive. Not support for G series.
21	VREG	P		Logic regulator output decoupling pin. A 1µF capacitor returning to VSSD must be placed on this pin.
22	VCCD	P		Main Digital Supply for Chip. Supplies all IO except analog, Speaker Driver
23	PB.0	I/O	0	General purpose input/output pin, analog capable; Port B, bit 0. Triggers external interrupt 0 (EINT0/IRQ2)
	SPI1_MOSI	O	1	Master Out, Slave In for SPI1 interface
	CS0	AI		Touch scan channel 0
	A0P	AI		Operational Amplifier 0 positive input
24	PB.1	I/O	0	General purpose input/output pin, analog capable; Port B, bit 1. Triggers external interrupt 1 (EINT1/IRQ3)
	SP1_SCLK	I/O	1	Serial Clock for SPI1 interface
	CS1	AI		Touch scan channel 1
	A0N	AI		Operational Amplifier 0 negative input
25	PB.2	I/O	0	General purpose input/output pin, analog capable; Port B, bit 2
	SPI1_SSB	O	1	Slave Select Bar for SPI1 interface
	CS2	AI		Touch scan channel 1
	A0E	AO		Operational Amplifier 0 output
	SAR11	AI		SARADC channel 11
26	PB.3	I/O	0	General purpose input/output pin, analog capable; Port B, bit 3
	SPI1_MISO	I	1	Master In, Slave Out for SPI1 interface
	CS3	AI		Touch scan channel 3
	A1P	AI		Operational Amplifier 1 positive input
27	VSSA	AP		Ground for analog circuitry.
28	VMID	O		Mid rail reference. Connect 4.7µF to VSSA.
29	MICP	AI		SDADC positive input. Not support for G & P series.
30	VSS_SARADC	AP		SARADC ground
31	MICN	AI		SDADC negative input. Not support for G & P series.

LQFP64 Pin Name		Pin Type	Alt CFG	Description
32	MICBIAS	O		Microphone bias output
EPAD	VSSD	P		Digital power ground

Note:

1. Pin Type I=Digital Input, O=Digital Output; AI=Analog Input; AO=Analog Output; P=Power Pin; AP=Analog Power
2. EPAD is VSSD, different power plane as VSSA

4 BLOCK DIAGRAM

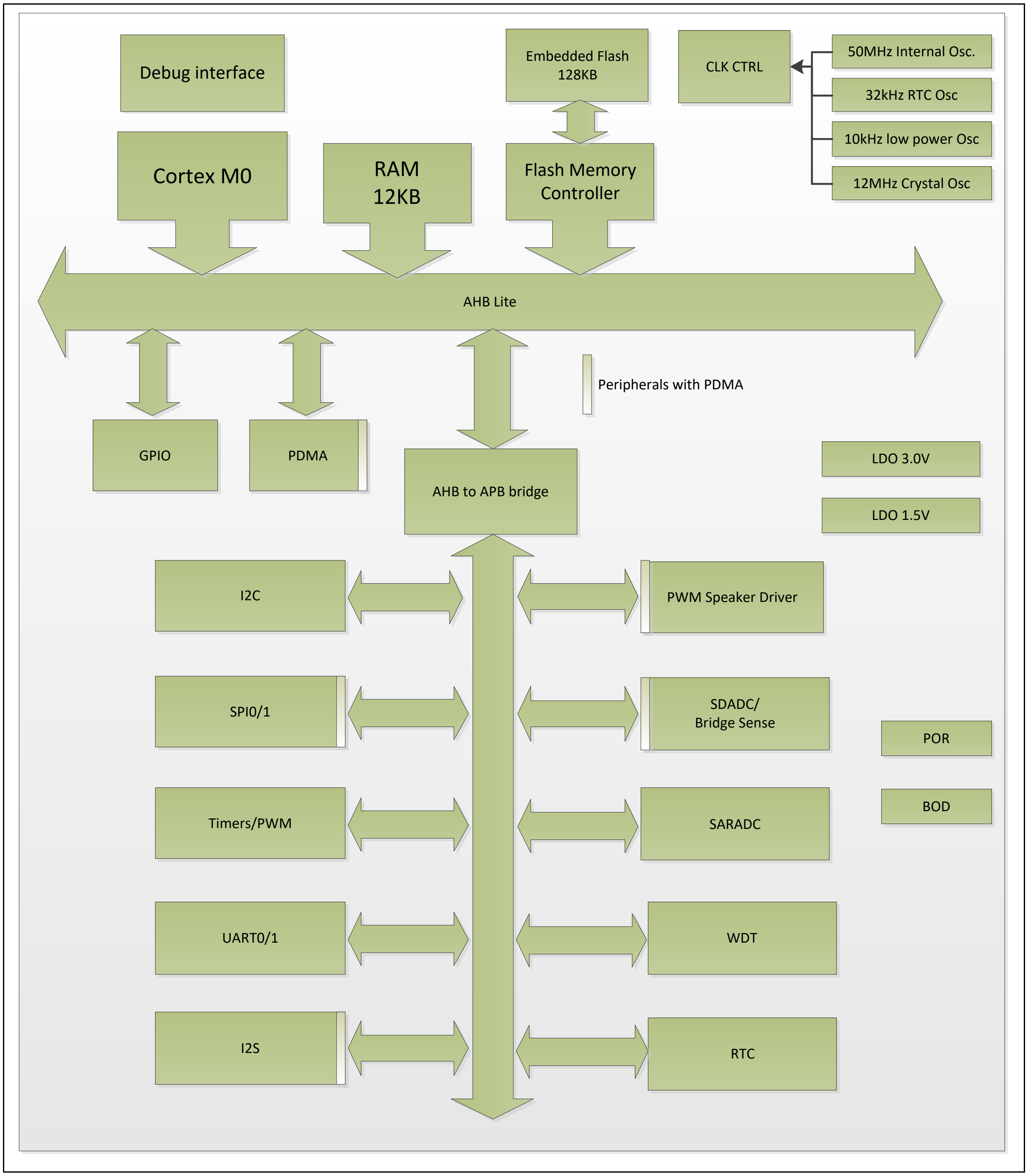


Figure 4-1 ISD91200 Block Diagram

5 FUNCTIONAL DESCRIPTION

5.1 ARM® Cortex™-M0 core

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

Figure 5-1 shows the functional blocks of processor.

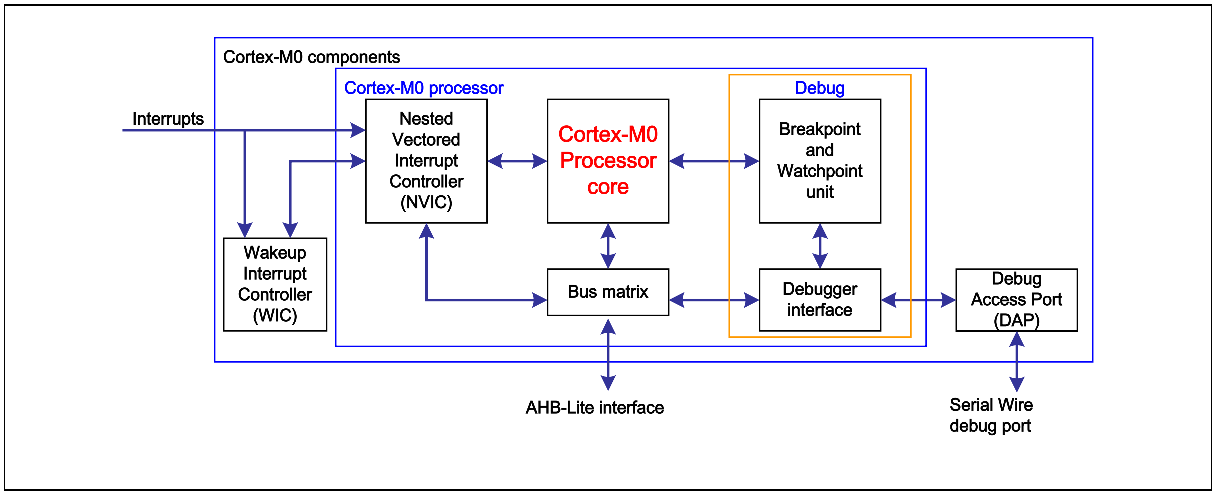


Figure 5-1 Functional Block Diagram

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 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 that 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 (Debug Access Port).

5.2 System Manager

5.2.1 Overview

The following functions are included in system manager section

- System Memory Map
- System Timer (SysTick)
- Nested Vectored Interrupt Controller (NVIC)
- System management registers for product ID
- System management registers for chip and module functional reset and multi-function pin control
- Brown-Out and chip miscellaneous Control Register
- Combined peripheral interrupt source identify

5.2.2 System Reset

The system reset includes one of the list below event occurs. For these reset event flags can be read by SYS_RSTSTS register.

- The Power-On Reset
- The low level on the RESETN pin
- Watchdog Time Out Reset
- Low Voltage Reset
- Cortex-M0 MCU Reset
- PMU Reset – for details of wakeup events, also examine CLK_PWRCTL register.
- SWD Debug interface.

A power-on reset (POR) will occur if the main external supply rail ramps from 0V or the voltage of the main supply drops below reset threshold. A low voltage reset monitors the regulated core logic (1.5V) supply and will assert if the voltage on this rail drops below reliable logic threshold.

5.2.3 System Power Distribution

The ISD91200 implements several power domains:

- Analog power from VCCA and VSSA provides the power for analog module operation.
- Digital power from VCCD and VSSD supplies the power to the IO ring and the internal regulator which provides 1.5V power for digital operation.
- VCCLDO supplies the LDO regulator whose output is available on pin VLDOx. This supply powers the IO ring for GPIOA<7:0>.
- An internal Standby reference (SB REG) generates a 1.5V rail to part of the logic including the IO ring, Standby RAM and RTC during standby mode for low power operation.

The outputs of internal voltage regulators; VREG and VDDB, require external decoupling capacitors which should be located close to the corresponding pin. The following diagram shows the power distribution of this device.

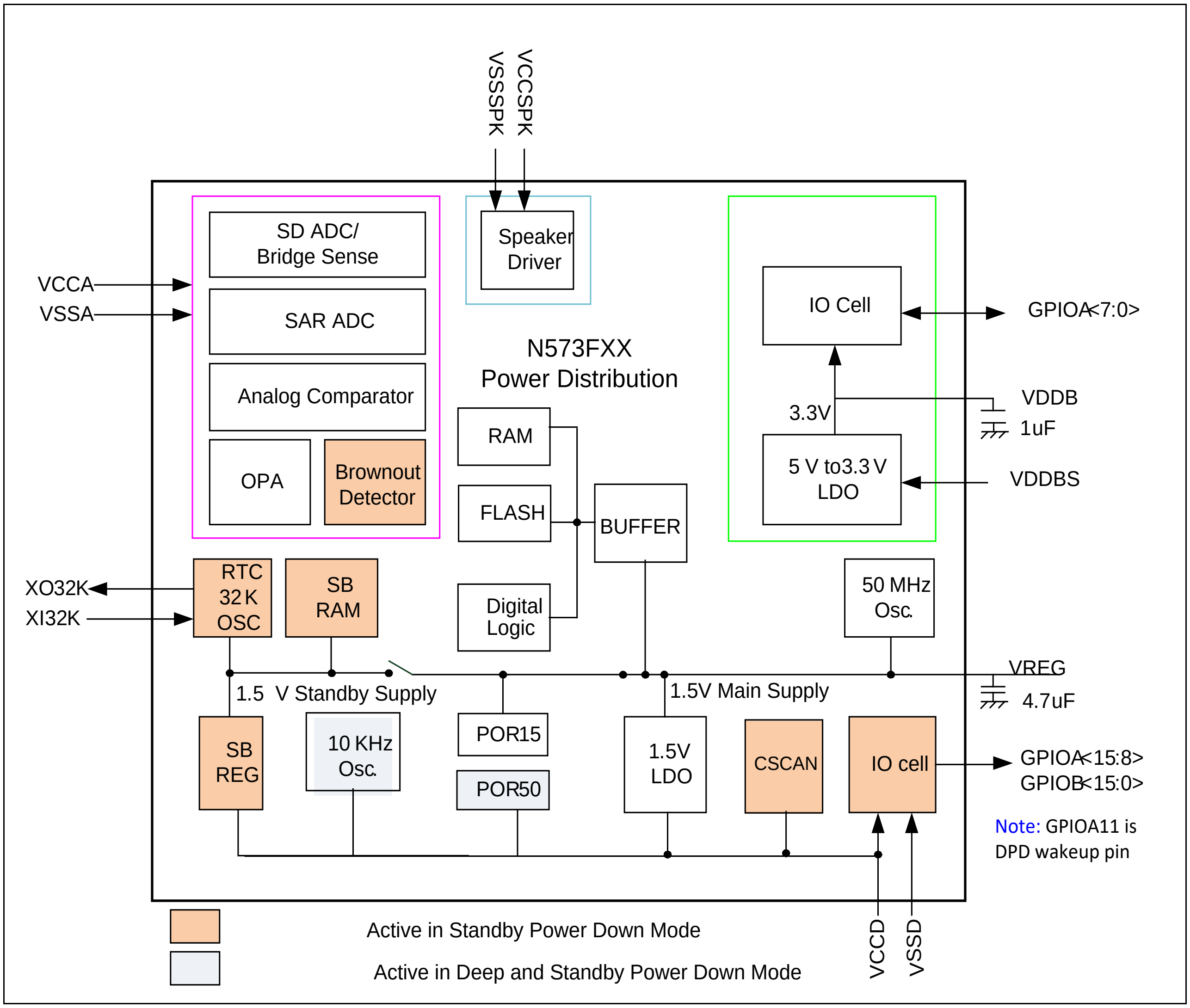


Figure 5-2 ISD91200 Power Distribution Diagram

5.2.4 System Memory Map

The ISD91200 provides 4G-byte address space. The memory locations assigned to each on-chip module is shown in Table 5-1. The detailed register definition, memory space, and programming detailed will be described in the following sections for each on-chip module. The ISD91200 supports little-endian data format.

Table 5-1 Address Space Assignments for On-Chip Modules

Address Space	**Token**	**Modules**
Flash & SRAM Memory Space		
0x0000_0000 – 0x0002_33FF	FLASH_BA	FLASH Memory Space (141KB)
0x2000_0000 – 0x2000_2FFF	SRAM_BA	SRAM Memory Space (12KB)
AHB Modules Space (0x5000_0000 – 0x501F_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_0400 – 0x5000_04FF	Reserved	
0x5000_4000 – 0x5000_7FFF	GPIO_BA	GPIO Control Registers
0x5000_8000 – 0x5000_BFFF	PDMA_BA	SRAM_APB DMA Control Registers
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash Memory Control Registers
APB1 Modules Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_7FFF	WDT_BA	Watch-Dog Timer Control Registers
0x4000_8000 – 0x4000_BFFF	RTC_BA	Real Time Clock (RTC) Control Register
0x4001_0000 – 0x4001_3FFF	TIMER0_BA	Timer0/Timer1 Control Registers
0x4002_0000 – 0x4002_3FFF	I2C_BA	I2C Interface Control Registers
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 Serial Interface Control Registers
0x4003_8000 – 0x4003_FFFF	SPI1_BA	SPI1 Serial Interface Control Registers
0x4004_0000 – 0x4004_3FFF	PWM_BA	PWM0/1 Control Registers
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0 Control Registers
0x4005_8000 – 0x4005_FFF	UART1_BA	UART1 Control Registers
0x4006_0000 – 0x4006_3FFF	SARADC_BA	SARADC Controller Registers
0x4007_0000 – 0x4007_3FFF	DPWM_BA	Differential Audio PWM Speaker Driver
0x4008_0000 – 0x4008_3FFF	ANA_BA	Analog Block Control Registers
0x4008_4000 – 0x4008_7FFF	BOD_BA	Brown Out Detector Control Registers
0x400A_0000 - 0x400A_FFFF	I2S_BA	I2S Interface Control registers
0x400B_0000 - 0x400B_009F	BIQ_BA	Biquad Filter Control Registers

0x400B_0090 – 0x400B_009F	ALC_BA	Analog Block Control Registers
0x400B_00A0 - 0x400B_00AF	VOLCTRL_BA	Volume Control Registers
0x400D_0000 – 0x400D_3FFF	CSCAN_BA	CAP SENSE Control Registers
0x400E_0000 – 0x400E_FFFF	SDADC_BA	Analog-Digital-Converter (ADC) Registers
0x400F_0000 – 0x400F_7FFF	SBRAM_BA	Standby RAM Block Address space
System Control Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SYSTICK_BA	System Timer Control Registers
0xE000_E100 – 0xE000_ECFF	SCS_BA	External Interrupt Controller Control Registers
0xE000_ED00 – 0xE000_ED8F	SYSINFO_BA	System Control Registers

5.2.5 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 ID	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_GPSMTEN	SYS_BA+0x30	R/W	GPIOA/B input type control register	0xFFFF_0000
SYS_GPA_MFP	SYS_BA+0x38	R/W	GPIOA multiple alternate functions control register	0x0000_0000
SYS_GPB_MFP	SYS_BA+0x3C	R/W	GPIOB multiple alternate functions control register	0x0000_0000
SYS_WKCTL	SYS_BA+0x54	R/W	WAKEUP pin control register	0x0000_0006
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control	0x0000_0000
SYS_IRCTCTL	SYS_BA+0x110	R/W	Oscillator Frequency Adjustment control register	0xXXXX_XXXX
SYS_OSC10KTRIM	SYS_BA+0x114	R/W	10kHz Oscillator (LIRC) Trim Register	0xXXXX_XXXX
SYS_OSCTRIM0	SYS_BA+0x118	R/W	Internal oscillator trim register 0	0xXXXX_XXXX
SYS_OSCTRIM1	SYS_BA+0x11C	R/W	Internal oscillator trim register 1	0xXXXX_XXXX
SYS_OSCTRIM2	SYS_BA+0x120	R/W	Internal oscillator trim register 2	0xXXXX_XXXX
SYS_XTALTRIM	SYS_BA+0x124	R/W	External Crystal oscillator trim register	0xXXXX_XXXX
Reserved	SYS_BA+0x128	R/W	System reserved, keep POR value	0xXXXX_XXXX

Note: In BSP register structure, the prefix is structure name, and register be no prefix, for example SYS_ is the prefix, SYS_XTALTRIM will be SYS->XTALTRIM

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 ID	0xXXXX_XXXX

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">PDID</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">PDID</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="8">PDID</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">PDID</td></tr>
</table>

Bits	Description	
[31:0]	**PDID**	**Product Identifier** Chip identifier for ISD91200 series.

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					**PORF**	**DPDRSTF**	**WKRSTF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CPURF	**PMURSTF**	**SYSRF**	**Reserved**	**LVRF**	**WDTRF**	**PADRF**	**CORERSTF**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**PORF**	**Power on Reset Flag** The PORF flag is set by hardware if device has powered up from a power on reset condition or standby power down. 0=No detected. 1= A power on Reset has occurred. This bit is cleared by writing 1 to itself. Writing 1 to this bit will clear bits PORF, DPDRSTF, and WKRSTF
[9]	**DPDRSTF**	**Deep Power Down Reset Flag** The DPDRSTF flag is set by hardware if device has powered up due to the DPD timer function. 0=No detected. 1= A power on was triggered by DPD timer. This bit is cleared by writing 1 to itself. Writing 1 to this bit will clear bits PORF, DPDRSTF, and WKRSTF
[8]	**WKRSTF**	**Wakeup Pin Reset Flag** The WKRSTF flag is set by hardware if device has powered up from deep power down (DPD) due to action of the WAKEUP pin. 0=No detected. 1= A power on was triggered by WAKEUP pin. This bit is cleared by writing 1 to itself. Writing 1 to this bit will clear bits PORF, DPDRSTF, and WKRSTF

[7]	**CPURF**	**Reset Source From CPU** The CPURF flag is set by hardware if software writes CPURST (SYS_IPRST0[1]) with a “1” to reset Cortex-M0 CPU kernel and Flash memory controller (FMC). 0= No reset from CPU. 1= The Cortex-M0 CPU kernel and FMC has been reset by software setting CPURST to 1. This bit is cleared by writing 1 to itself.
[6]	**PMURSTF**	**Reset Source From PMU** The PMURSTF flag is set if the PMU. 0= No reset from PMU. 1= PMU reset the system from a power down/standby event. This bit is cleared by writing 1 to itself.
[5]	**SYSRF**	**Reset Source From MCU** The SYSRF flag is set if the previous reset source originates from the Cortex_M0 kernel. 0= No reset from MCU. 1= The Cortex_M0 MCU issued a reset signal to reset the system by software writing 1 to bit SYSRESTREQ(SYSINFO_AIRCTL[2], Application Interrupt and Reset Control Register) in system control registers of Cortex_M0 kernel. This bit is cleared by writing 1 to itself.
[4]	**Reserved**	Reserved.
[3]	**LVRF**	**Low Voltage Reset Flag** The LVRF flag is set if pervious reset source originates from the LVR module. 0 = No reset from LVR 1 = The LVR module issued the reset signal to reset the system. This bit is cleared by writing 1 to itself.
[2]	**WDTRF**	**Reset Source From WDT** 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. This bit is cleared by writing 1 to itself.
[1]	**PADRF**	**The RSTS_PAD Flag Is If Pervious Reset Source Originates From the /RESET Pin** 0 = No reset from Pin /RESET. 1 = Pin /RESET had issued the reset signal to reset the system. This bit is cleared by writing 1 to itself.
[0]	**CORERSTF**	**Reset Source From CORE** The CORERSTF flag is set if the core has been reset. Possible sources of reset are a Power-On Reset (POR), RESETn Pin Reset or PMU reset. 0 = No reset from CORE. 1 = Core was reset by hardware block. This bit is cleared by writing 1 to itself.

IP Reset Control Register1(SYS_IPRST0)

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Controller Reset** Set “1” will generate a reset signal to the PDMA Block. User needs to set this bit to “0” to release from the reset state 0= Normal operation. 1= PDMA IP reset.
[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 This bit is a protected bit, to program first issue the unlock sequence 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 the 2 clock cycles. CHIPRST has same behavior as POR reset, all the chip modules are reset and the chip configuration settings from flash are reloaded. This bit is a protected bit, to program first issue the unlock sequence 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
Reserved	ANARST	I2S0RST	SDADCRST	SARADCRST	Reserved		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			PWM0RST	Reserved	BIQRST	UART1RST	UART0RST
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	Reserved	DPWMRST	SPI0RST	SPI1RST	Reserved	Reserved	I2C0RST
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TMR1RST	TMR0RST	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**ANARST**	**Analog Block Control Reset** 0=Normal Operation. 1=Reset.
[29]	**I2S0RST**	**I2S Controller Reset** 0=Normal Operation. 1=Reset.
[28]	**SDADCRST**	**SDADC Controller Reset** 0=Normal Operation. 1=Reset.
[27]	**SARADCRST**	**SAR ADC Controller Reset,** 0 =Normal Operation. 1 = Reset.
[26:21]	**Reserved**	Reserved.
[20]	**PWM0RST**	**PWM0 Controller Reset** 0=Normal Operation. 1=Reset.
[19]	**Reserved**	**Reserved**
[18]	**BIQRST**	**Biquad Filter Block 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.
[13]	**DPWMRST**	**DPWM Speaker Driver Reset** 0=Normal Operation. 1=Reset.
[12]	**SPI0 RST**	**SPI0 Controller Reset** 0=Normal Operation. 1=Reset.
[11]	**SPI1 RST**	**SPI1 Controller Reset** 0=Normal Operation. 1=Reset.
[10:9]	**Reserved**	Reserved.
[8]	**I2C0RST**	**I2C0 Controller Reset** 0=Normal Operation. 1=Reset.
[7]	**TMR1RST**	**Timer1 Controller Reset** 0=Normal Operation. 1=Reset.
[6]	**TMR0RST**	**Timer0 Controller Reset** 0=Normal Operation. 1=Reset.
[5:0]	**Reserved**	Reserved.

GPIOA Input Type Control Register (SYS_GPSMTEN)

Register	Offset	R/W	Description	Reset Value
SYS_GPSMTEN	SYS_BA+0x30	R/W	GPIOA/B input type control register	0xFFFF_0000

31	30	29	28	27	26	25	24
HSGPBG3	SSGPBG3	HSGPBG2	SSGPBG2	HSGPBG1	SSGPBG1	HSGPBG0	SSGPBG0
23	22	21	20	19	18	17	16
HSGPAG3	SSGPAG3	HSGPAG2	SSGPAG2	HSGPAG1	SSGPAG1	HSGPAG0	SSGPAG0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31]	**HSSGPBG3**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOB 15/14/13/12 Output high slew rate. 0 = GPIOB 15/14/13/12 Output low slew rate.
[30]	**SSGPBG3**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOB 15/14/13/12 input Schmitt Trigger enabled. 0 = GPIOB 15/14/13/12 input CMOS enabled.
[29]	**HSSGPBG2**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOB 11/10/9/8 Output high slew rate. 0 = GPIOB 11/10/9/8 Output low slew rate.
[28]	**SSGPBG2**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOB 11/10/9/8 input Schmitt Trigger enabled. 0 = GPIOB 11/10/9/8 input CMOS enabled.
[27]	**HSSGPBG1**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOB 7/6/5/4 Output high slew rate. 0 = GPIOB 7/6/5/4 Output low slew rate.

[26]	**SSGPBG1**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOB 7/6/5/4 input Schmitt Trigger enabled. 0 = GPIOB 7/6/5/4 input CMOS enabled.
[25]	**HSSGPBG0**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOB 3/2/1/0 Output high slew rate. 0 = GPIOB 3/2/1/0 Output low slew rate.
[24]	**SSGPBG0**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOB 3/2/1/0 input Schmitt Trigger enabled. 0 = GPIOB 3/2/1/0 input CMOS enabled.
[23]	**HSSGPAG3**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOA 15/14/13/12 Output high slew rate. 0 = GPIOA 15/14/13/12 Output low slew rate.
[22]	**SSGPAG3**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOA 15/14/13/12 input Schmitt Trigger enabled. 0 = GPIOA 15/14/13/12 input CMOS enabled.
[21]	**HSSGPAG2**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOA 11/10/9/8 Output high slew rate. 0 = GPIOA 11/10/9/8 Output low slew rate.
[20]	**SSGPAG2**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOA 11/10/9/8 input Schmitt Trigger enabled. 0 = GPIOA 11/10/9/8 input CMOS enabled.
[19]	**HSSGPAG1**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOA 7/6/5/4 Output high slew rate. 0 = GPIOA 7/6/5/4 Output low slew rate.
[18]	**SSGPAG1**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOA 7/6/5/4 input Schmitt Trigger enabled. 0 = GPIOA 7/6/5/4 input CMOS enabled.

[17]	**HSSGPAG0**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOA 3/2/1/0 Output high slew rate. 0 = GPIOA 3/2/1/0 Output low slew rate.
[16]	**SSGPAG0**	**This Register Controls Whether the GPIO Input Buffer Schmitt Trigger Is Enabled and Whether High or Low Slew Rate Is Selected for Output Dr.** Each bit controls a group of four GPIO pins 1 = GPIOA 3/2/1/0 input Schmitt Trigger enabled. 0 = GPIOA 3/2/1/0 input CMOS enabled.
[15:0]	**Reserved**	Reserved.

GPIO PA Alternative Function Control Register (SYS_GPA_MFP)

Each GPIO pin can take on multiple alternate functions depending on the setting of this register. Each pin has two bits of alternate function control. Set to 00 the pin is a standard GPIO pin whose attributes are defined by the GPIO control registers (See Section 0). Set to other values the pin is assigned to a peripheral as outlined in table below.

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP	SYS_BA+0x38	R/W	GPIOA multiple alternate functions 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**	**Alternate Function Setting for PA15MFP** 00 = GPIO. 01 = UART1_RX. 10 = MCLK. 11 = X32KO.
[29:28]	**PA14MFP**	**Alternate Function Setting for PA14MFP** 00 = GPIO. 01 = UART1_TX. 10 = DMIC_CLK. 11 = X32KI.
[27:26]	**PA13MFP**	**Alternate Function Setting for PA13MFP** 00 = GPIO. 01 = PWM0CH3. 10 = X12MO. 11 = I2C0_SCL.
[25:24]	**PA12MFP**	**Alternate Function Setting for PA12MFP** 00 = GPIO. 01 = PWM0CH2. 10 = X12MI. 11 = I2C0_SDA.

[23:22]	**PA11MFP**	**Alternate Function Setting for PA11MFP** 00 = GPIO. 01 = PWM0CH1. 10 = TM1. 11 = DPWM_M.
[21:20]	**PA10MFP**	**Alternate Function Setting for PA10MFP** 00 = GPIO. 01 = PWM0CH0. 10 = TM0. 11 = DPWM_P.
[19:18]	**PA9MFP**	**Alternate Function Setting for PA9MFP** 00 = GPIO. 01 = I2C0_SCL. 10 = UART1_RX. 11 = UART0_CTSn.
[17:16]	**PA8MFP**	**Alternate Function Setting for PA8MFP** 00 = GPIO. 01 = I2C0_SDA. 10 = UART1_TX. 11 = UART0_RTSn.
[15:14]	**PA7MFP**	**Alternate Function Setting for PA7MFP** 00 = GPIO. 01 = UART0_RX. 10 = I2C0_SCL. 11 = SPI1_MISO.
[13:12]	**PA6MFP**	**Alternate Function Setting for PA6MFP** 00 = GPIO. 01 = UART0_TX. 10 = I2C0_SDA. 11 = SPI1_SSB.
[11:10]	**PA5MFP**	**Alternate Function Setting for PA5MFP** 00 = GPIO. 01 = SPI0_MOSI1. 10 = UART0_RX. 11 = SPI1_SCLK.
[9:8]	**PA4MFP**	**Alternate Function Setting for PA4MFP** 00 = GPIO. 01 = SPI0_MISO0. 10 = UART0_TX. 11 = SPI1_MOSI

[7:6]	**PA3MFP**	**Alternate Function Setting for PA3MFP** 00 = GPIO. 01 = SPI0_SSB0. 10 = SARADC_TRIG. 11 = I2S0_SDO.
[5:4]	**PA2MFP**	**Alternate Function Setting for PA2MFP** 00 = GPIO. 01 = SPI0_SCLK0. 10 = DMIC_DAT. 11 = I2S0_SDI
[3:2]	**PA1MFP**	**Alternate Function Setting for PA1MFP** 00 = GPIO. 01 = SPI0_MOSI0. 11 = I2S0_BCLK.
[1:0]	**PA0MFP**	**Alternate Function Setting for PA0MFP** 00 = GPIO. 01 = SPI0_MISO1. 11 = I2S0_FS.

GPIO PB Alternative Function Control Register (SYS_GPB_MFP)

Each GPIO pin can take on multiple alternate functions depending on the setting of this register. Each pin has two bits of alternate function control. Set to 00 the pin is a standard GPIO pin whose attributes are defined by the GPIO control registers (See Section 0). Set to other values the pin is assigned to a peripheral as outlined in table below.

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFP	SYS_BA+0x3C	R/W	GPIOB multiple alternate functions control register	0x0000_0000

31	30	29	28	27	26	25	24
PB15MFP		PB14MFP		PB13MFP		PB12MFP	
23	22	21	20	19	18	17	16
PB11MFP		PB10MFP		PB9MFP		PB8MFP	
15	14	13	12	11	10	9	8
PB7MFP		PB6MFP		PB5MFP		PB4MFP	
7	6	5	4	3	2	1	0
PB3MFP		PB2MFP		PB1MFP		PB0MFP	

Bits	Description	
[31:30]	**PB15MFP**	**Alternate Function Setting for PB15MFP** 00 = GPIO. 01 = SPI0_SSB0 10 = SPI1_MISO. 11 = MCLK.
[29:28]	**PB14MFP**	**Alternate Function Setting for PB14MFP** 00 = GPIO. 01 = SPI0_SCLK0. 10 = SPI1_SSB. 11 = DMIC_CLK.
[27:26]	**PB13MFP**	**Alternate Function Setting for PB13MFP** 00 = GPIO. 01 = SPI0_MOSI0. 10 = SPI1_SCLK. 11 = SARADC_TRIG.
[25:24]	**PB12MFP**	**Alternate Function Setting for PB12MFP** 00 = GPIO. 01 = SP0_MISO1. 10 = SPI1_MOSI. 11 = DMIC_DAT.

[23:22]	**PB11MFP**	**Alternate Function Setting for PB11MFP** 00 = GPIO. 10 = I2S0_SDO. 11 = UART1_RX.
[21:20]	**PB10MFP**	**Alternate Function Setting for PB10MFP** 00 = GPIO. 10 = I2S0_SDI. 11 = UART1_TX.
[19:18]	**PB9MFP**	**Alternate Function Setting for PB9MFP** 00 = GPIO. 01 = I2C0_SCL. 10 = I2S0_BCLK. 11 = UART1_CTsn.
[17:16]	**PB8MFP**	**Alternate Function Setting for PB8MFP** 00 = GPIO. 01 = I2C0_SDA 10 = I2S0_FS. 11 = UART1_RSTn.
[15:14]	**PB7MFP**	**Alternate Function Setting for PB7MFP** 00 = GPIO. 01 = I2S0_SDO.
[13:12]	**PB6MFP**	**Alternate Function Setting for PB6MFP** 00 = GPIO. 01 = I2S0_SDI.
[11:10]	**PB5MFP**	**Alternate Function Setting for PB5MFP** 00 = GPIO. 01 = I2S0_BCLK.
[9:8]	**PB4MFP**	**Alternate Function Setting for PB4MFP** 00 = GPIO. 01 = I2S0_FS.
[7:6]	**PB3MFP**	**Alternate Function Setting for PB3MFP** 00 = GPIO. 01 = SPI1_MISO.
[5:4]	**PB2MFP**	**Alternate Function Setting for PB2MFP** 00 = GPIO. 01 = SPI1_SSB.
[3:2]	**PB1MFP**	**Alternate Function Setting for PB1MFP** 00 = GPIO. 01 = SPI1_SCLK.
[1:0]	**PB0MFP**	**Alternate Function Setting for PB0MFP** 00 = GPIO. 01 = SPI1_MOSI.

Table 5-2 GPIOA Alternate Function Register (SYS_GPA_MFP address 0x5000_0038)

GPIO	Power Domain	GPAn=01		GPAn =10		GPAn =11	
		Function	Type	Function	Type	Function	Type
GPIOA0	VDD33	SPI0_HOLD0 (MISO1)	IO			I2S0_FS	IO
GPIOA1	VDD33	SPI0_MOSI0	O			I2S0_BCLK	IO
GPIOA2	VDD33	SPI0_SCLK0	O	DMIC_DAT	I	I2S0_SDI	I
GPIOA3	VDD33	SPI0_SSB0	IO	SARADC_TRIG	I	I2S0_SDO	O
GPIOA4	VDD33	SPI0_MISO0	IO	UART0_TX	O	SPI1_MOSI	O
GPIOA5	VDD33	SPI0_WP0 (MOSI1)	IO	UART0_RX	I	SPI1_SCLK	O
GPIOA6	VDD33	UART0_TX	O	I2C0_SDA	IO	SPI1_SSB	IO
GPIOA7	VDD33	UART0_RX	I	I2C0_SCL	IO	SPI1_MISO	I
GPIOA8	VCCD	I2C0_SDA	IO	UART1_TX	O	UART0_RTSn	O
GPIOA9	VCCD	I2C0_SCL	IO	UART1_RX	I	UART0_CTSn	I
GPIOA10	VCCD	PWM0CH0	O	TM0	I	DPWM_P	O
GPIOA11	VCCD	PWM0CH1	O	TM1	I	DPWM_M	O
GPIOA12	VCCD	PWM0CH2	O	X12MI	I	I2C0_SDA	IO
GPIOA13	VCCD	PWM0CH3	O	X12MO	O	I2C0_SCL	IO
GPIOA14	VCCD	UART1_TX	O	DMIC_CLK	IO	X32KI	I
GPIOA15	VCCD	UART1_RX	IO	MCLK	O	X32KO	O

Table 5-3 GPIOB Alternate Function Register (SYS_GPB_MFP address 0x5000_003C)

GPIO	Power Domain	GPBn=01		GPBn =10		GPBn =11		Analog Functions		
		Function	Type	Function	Type	Function	Type			
GPIOB0	VCCD	SPI1_MOSI	O					CS0	A0P	
GPIOB1	VCCD	SPI1_SCLK	O					CS1	A0N	
GPIOB2	VCCD	SPI1_SSB	IO					CS2	A0E	SAR11
GPIOB3	VCCD	SPI1_MISO	I					CS3	A1P	
GPIOB4	VCCD	I2S0_FS	IO					CS4	A1N	
GPIOB5	VCCD	I2S0_BCLK	IO					CS5	A1E	SAR10
GPIOB6	VCCD	I2S0_SDI	I					CS6	CNP	SAR8

GPIOB7	VCCD	I2S0_SDO	O					CS7	C1N	SAR9
GPIOB8	VCCD	I2C0_SDA		I2S0_FS	IO	UART1_RSTn	IO	CS8	C2P	SAR0
GPIOB9	VCCD	I2C0_SCL		I2S0_BCLK	IO	UART1_CTsn	IO	CS9		SAR1
GPIOB10	VCCD	CMP1	O	I2S0_SDI	I	UART1_TX	I	CS10		SAR2
GPIOB11	VCCD	CMP2	O	I2S0_SDO	O	UART1_RX	I	CS11		SAR3
GPIOB12	VCCD	SPI0_MISO1	I	SPI1_MOSI	O	DMIC_DAT	I	CS12		SAR4
GPIOB13	VCCD	SPI0_MOSI0	O	SPI1_SCLK	O	SARADC_TRIG	O	CS13		SAR5
GPIOB14	VCCD	SPI0_SCLK0	O	SPI1_SSB	IO	DMIC_CLK	O	CS14		SAR6
GPIOB15	VCCD	SPI0_SSB0	O	SPI1_MISO	I	MCLK	O	CS15		SAR7

Protected Register Lock Key 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 open the lock. The unlock sequence is to write to SYS_REGLCTL the data 0x59, 0x16, 0x88. Any different sequence, data or a write to any other address will abort the unlock sequence.

MDK provides the defined function UNLOCKREG(x); which will execute this sequence.

The status of the lock can be determined by reading SYS_REGLCTL bit0: “1” is unlocked, “0” is locked. Once unlocked, user can update protected register values. To lock registers again, write any data to SYS_REGLCTL.

This register is write accessible for writing key values and read accessible to determine REGLCTL status.

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**REGLCTL**	**Protected Register Unlock Register** 0 = Protected registers are locked. Any write to the target register is ignored. 1 = Protected registers are unlocked.

Oscillator Trim Control Register (SYS_IRCTCTL)

The master oscillator of the ISD91200 has an adjustable frequency and can be controlled by the SYS_IRCTCTL register. This register contains the current scillator frequency trim value, which depends upon the setting of register CLK_CLKSEL0.HIRCFSEL register. If this register is 0, SYS_OSCTRIM0 trim is active, if 1 then SYS_OSCTRIM1 is active, if 2 then SYS_OSCTRIM2. Upon power on reset this register is loaded from flash memory with factory stored values to give oscillator frequencies of 49.152MHz for SYS_OSCTRIM0, 32.768MHz for SYS_OSCTRIM1 and no factory trimming for SYS_OSCTRIM2 (customer need to set a workable value before change to use). If users wish to change the default frequency, it is possible to do so by setting this register. A write to FREQ will change the active SYS_OSCTRIMn register, or writing active SYS_OSCTRIMn.TRIM (n=0,1,2) will change the FREQ.

This register is a protected register, to write to register first issue the unlock sequence

Register	Offset	R/W	Description	Reset Value
SYS_IRCTCTL	SYS_BA+0x110	R/W	Oscillator Frequency Adjustment control 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**
RANGE	**Reserved**					**FREQ**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FREQ							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**RANGE**	1: Low Frequency mode of oscillator active (2MHz). 0: High frequency mode (20-50MHz)
[14:10]	**Reserved**	Reserved.
[9:0]	**FREQ**	Current oscillator frequency trim value. (based on CLK_CLKSEL0.HIRCFSEL)

10kHz Oscillator Trim Register (SYS_OSC10KTRIM)

Register	Offset	R/W	Description	Reset Value
SYS_OSC10KTRIM	SYS_BA+0x114	R/W	10kHz Oscillator (LIRC) Trim Register	0xXXXX_XXXX

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

Bits	Description	
[31]	**TRMCLK**	Must be toggled to(from 0 => 1 => 0) load a new OSC10K_TRIM
[30:23]	**Reserved**	Reserved.
[22:0]	**TRIM**	23bit trim for LIRC.

Oscillator Trim 0 Control Register (SYS_OSCTRIM0)

Register	Offset	R/W	Description	Reset Value
SYS_OSCTRIM0	SYS_BA+0x118	R/W	Internal oscillator trim register 0	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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRIM							

Bits	Description	
[31]	**EN2MHZ**	1: Low Frequency mode of oscillator active (2MHz). 0: High frequency mode (20-50MHz)
[30:16]	**Reserved**	Reserved.
[15:0]	**TRIM**	16bit sign extended representation of 10bit trim.

Oscillator Trim 1 Control Register (SYS_OSCTRIM1)

Register	Offset	R/W	Description	Reset Value
SYS_OSCTRIM1	SYS_BA+0x11C	R/W	Internal oscillator trim register 1	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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRIM							

Bits	Description	
[31]	**EN2MHZ**	1: Low Frequency mode of oscillator active (2MHz). 0: High frequency mode (20-50MHz)
[30:16]	**Reserved**	Reserved.
[15:0]	**TRIM**	16bit sign extended representation of 10bit trim.

Oscillator Trim 2 Control Register (SYS_OSCTRIM2)

Register	Offset	R/W	Description	Reset Value
SYS_OSCTRIM2	SYS_BA+0x120	R/W	Internal oscillator trim register 2	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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRIM							

Bits	Description	
[31]	**EN2MHZ**	1: Low Frequency mode of oscillator active (2MHz). 0: High frequency mode (20-50MHz)
[30:16]	**Reserved**	Reserved.
[15:0]	**TRIM**	16bit sign extended representation of 10bit trim.

XTAL32K Oscillator Control Register (SYS_XTALTRIM)

Register	Offset	R/W	Description	Reset Value
SYS_XTALTRIM	SYS_BA+0x124	R/W	External Crystal oscillator trim register	0xXXXX_XXXX

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:24]	**XGS**	HXT Gain Select
[23:17]	**Reserved**	Reserved.
[16]	**SELXT**	HXT select external clock 0: Disable 1: Enable
[15:10]	**Reserved**	Reserved.
[9]	**LOWPWR**	1: low power mode. 0: normal mode.
[8:0]	**Reserved**	Leave at default. Do not modify

5.2.6 System Timer (SysTick)

The Cortex-M0 includes an integrated system timer, SysTick. SysTick 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 SysTick 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 SysTick Current Value Register (SYST_CVR) to zero, reload (wrap) to the value in the SysTick 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 SysTick 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”.

5.2.6.1 System Timer Control 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	SysTick Control and Status Register	0x0000_0000
SYST_RVR	SYSTICK_BA+0x14	R/W	SysTick Reload value Register	0xXXXX_XXXX
SYST_CVR	SYSTICK_BA+0x18	R/W	SysTick Current value Register	0xXXXX_XXXX

Note: In BSP register structure, the prefix is structure name, and register will be no prefix, for example SYSTICK_ is the prefix, SYSTICK_CSR will be SYSTICK->CSR

5.2.6.2 System Timer Control Register Description

SysTick Control and Status (SYST_CSR)

Register	Offset	R/W	Description	Reset Value
SYST_CSR	SYSTICK_BA+0x10	R/W	SysTick 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
[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
[2]	**CLKSRC**	**Clock Source** 0= Core clock unused. 1= Core clock used for SysTick, this bit will read as 1 and ignore writes.
[1]	**TICKINT**	**Enables SysTick Exception Request** 0 = Counting down to 0 does not cause the SysTick exception to be pended. Software can use COUNTFLAG to determine if a count to zero has occurred. 1 = Counting down to 0 will cause SysTick exception to be pended. Clearing the SysTick Current Value register by a register write in software will not cause SysTick to be pended.
[0]	**ENABLE**	**ENABLE** 0 = The counter is disabled 1 = The counter will operate in a multi-shot manner.

SysTick Reload Value Register (SYST_RVR)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved
[23:0]	**RELOAD**	**SysTick 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 SysTick interrupt is required every 200 clock pulses, set RELOAD to 199.

SysTick Current Value Register (SYST_CVR)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved
[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.

5.2.7 Nested Vectored Interrupt Controller (NVIC)

Cortex-M0 includes an interrupt controller the “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 32 (IRQ [31:0]) 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 corresponding 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 above mentioned registers from the stack and resume normal execution. This provides a high speed and deterministic time to process any interrupt request.

The NVIC supports “Tail Chaining” which handles back-to-back interrupts efficiently without the overhead of state saving and restoration and therefore reduces delay time in switching to a pending ISR at the end of the 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. This aids real-time, high priority, interrupt capability.

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

5.2.7.1 Exception Model and System Interrupt Map

The following table lists the exception model supported by ISD91200 series. 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 5-4 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 5-5 System Interrupt Map

Vector Number	**Interrupt Number** **(Bit in Interrupt Registers)**	**Interrupt Name**	**Source IP**	**Interrupt description**
0 ~ 15	-	**-**	-	System exceptions
16	0	**BOD_IRQn**	Brown-Out	Brownout low voltage detector interrupt
17	1	**WDT_IRQn**	WDT	Watch Dog Timer interrupt
18	2	**EINT0_IRQn**	GPIO	External signal interrupt from PB.0 pin
19	3	**EINT1_IRQn**	GPIO	External signal interrupt from PB.1 pin
20	4	**GPAB_IRQn**	GPIO	External signal interrupt from PA[15:0] / PB[7:2]
21	5	**ALC_IRQn**	ALC	Automatic Level Control Interrupt
22	6	**PWM0_IRQn**	PWM0	PWM0 channel 0/1/2/3 interrupt
23	7	Reserved		
24	8	**TMR0_IRQn**	TMR0	Timer 0 interrupt
25	9	**TMR1_IRQn**	TMR1	Timer 1 interrupt
26	10	**Reserved**		

27	11	**UART1_IRQn**	UART1	UART1interrupt
28	12	**UART0_IRQn**	UART0	UART0 interrupt
29	13	**SPI1_IRQn**	SPI1	SPI1 interrupt
30	14	**SPI0_IRQn**	SPI0	SPI0 interrupt
31	15	**DPWM_IRQn**	DPWM	DPWM interrupt
32	16	**Reserved**		
33	17	**Reserved**		
34	18	**I2C0_IRQn**	I2C0	I2C0 interrupt
35	19	**Reserved**		
36	20	**Reserved**		
37	21	**CMP_IRQn**	CMP	CMP interrupt
38	22	**MAC_IRQnReserved**	MAC	MAC interrupt
39	23	**Reserved**		
40	24	**Reserved**		
41	25	**SARADC_IRQn**	SARADC	SARADC interrupt
42	26	**PDMA_IRQn**	PDMA	PDMA interrupt
43	27	**I2S0_IRQn**	I2S	I2S0 interrupt
44	28	**CAPS_IRQn**	ANA	Capacitive Touch Sensing Relaxation Oscillator Interrupt
45	29	**ADC_INT**	SDADC	Audio ADC interrupt
46	30	**Reserved**		
47	31	**RTC_INT**	RTC	Real time clock interrupt

5.2.7.2 *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.

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

Table 5-6 Vector Table Format

5.2.7.3 *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.

5.2.7.4 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_E100**				
NVIC_ISER	SCS_BA+0x000	R/W	IRQ0 ~ IRQ31 Set-Enable Control Register	0x0000_0000
NVIC_ICER	SCS_BA+0x080	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000
NVIC_ISPR	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000
NVIC_ICPR	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000
NVIC_IPR0	SCS_BA+0x300	R/W	IRQ0 ~ IRQ3 Priority Control Register	0x0000_0000
NVIC_IPR1	SCS_BA+0x304	R/W	IRQ4 ~ IRQ7 Priority Control Register	0x0000_0000
NVIC_IPR2	SCS_BA+0x308	R/W	IRQ8 ~ IRQ11 Priority Control Register	0x0000_0000
NVIC_IPR3	SCS_BA+0x30C	R/W	IRQ12 ~ IRQ15 Priority Control Register	0x0000_0000
NVIC_IPR4	SCS_BA+0x310	R/W	IRQ16 ~ IRQ19 Priority Control Register	0x0000_0000
NVIC_IPR5	SCS_BA+0x314	R/W	IRQ20 ~ IRQ23 Priority Control Register	0x0000_0000
NVIC_IPR6	SCS_BA+0x318	R/W	IRQ24 ~ IRQ27 Priority Control Register	0x0000_0000
NVIC_IPR7	SCS_BA+0x31C	R/W	IRQ28 ~ IRQ31 Priority Control Register	0x0000_0000

Note: In BSP register structure, the prefix is structure name, and register will be no prefix, for example NVIC_ is the prefix, NVIC_ISER will be NVIC->ISER

IRQ0 ~ IRQ31 Set-Enable Control Register (NVIC_ISER)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER	SCS_BA+0x000	R/W	IRQ0 ~ IRQ31 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
SETENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETENA							

Bits	Description	
[31:0]	**SETENA**	**Set-enable Control** Enable one or more interrupts within a group of 32. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Writing 1 will enable the associated interrupt. Writing 0 has no effect. The register reads back the current enable state.

IRQ0 ~ IRQ31 Clear-Enable Control Register (NVIC_ICER)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER	SCS_BA+0x080	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**CLRENA**	**Clear-enable Control** Disable one or more interrupts within a group of 32. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Writing 1 will disable the associated interrupt. Writing 0 has no effect. The register reads back with the current enable state.

IRQ0 ~ IRQ31 Set-Pending Control Register (NVIC_ISPR)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**SETPEND**	**Set-pending Control** Writing 1 to a bit forces pending state of the associated interrupt under software control. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Writing 0 has no effect. The register reads back with the current pending state.

IRQ0 ~ IRQ31 Clear-Pending Control Register (NVIC_ICPR)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**CLRPEND**	**Clear-pending Control** Writing 1 to a bit to clear the pending state of associated interrupt under software control. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Writing 0 has no effect. The register reads back with the current pending state.

IRQ0 ~ IRQ3 Interrupt Priority Register (NVIC_IPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR0	SCS_BA+0x300	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+0x304	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+0x308	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+0x30C	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+0x310	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+0x314	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 ~ IRQ27 Interrupt Priority Register (NVIC_IPR6)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR6	SCS_BA+0x318	R/W	IRQ24 ~ IRQ27 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_27		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	
[31:30]	**PRI_27**	**Priority of IRQ27** “0” denotes the highest priority and “3” denotes lowest priority
[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

IRQ28 ~ IRQ31 Interrupt Priority Register (NVIC_IPR7)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR7	SCS_BA+0x31C	R/W	IRQ28 ~ IRQ31 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_31		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_30		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_29		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_28		**Reserved**					

Bits	Description	
[31:30]	**PRI_31**	**Priority of IRQ31** "0" denotes the highest priority and "3" denotes lowest priority
[23:22]	**PRI_30**	**Priority of IRQ30** "0" denotes the highest priority and "3" denotes lowest priority
[15:14]	**PRI_29**	**Priority of IRQ29** "0" denotes the highest priority and "3" denotes lowest priority
[7:6]	**PRI_28**	**Priority of IRQ28** "0" denotes the highest priority and "3" denotes lowest priority

5.2.7.5 Interrupt Source Control Registers

Along with the interrupt control registers associated with the NVIC, the ISD91200 also implements some specific control registers to facilitate the interrupt functions, including “interrupt source identify”, ”NMI source selection” and “interrupt test mode”. They are described as below.

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

Register	Offset	R/W	Description	Reset Value
INT Base Address: **INT_BA = 0x5000_0300**				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/B) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (ALC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWM0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (Reserved) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (Reserved) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (UART1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (SPI1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (DPWM) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (Reserved) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (Reserved) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I2C0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (Reserved) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (Reserved) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (CMP) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (MAC) Interrupt Source Identity Register	0xXXXX_XXXX

IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (Reserved) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	IRQ24 (Reserved) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (SARADC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (PDMA) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	IRQ27 (I2S0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (CAPS) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	IRQ30 (Reserved) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	R	IRQ31 (RTC) Interrupt Source Identity Register	0xXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_0000
MCU_IRQ	INT_BA+0x84	R/W	MCU IRQ Number Identify Register	0x0000_0000

Note: In BSP register structure, INT_ is structure name, for example INT->MCU_IRQ.

IRQ0(BOD) Interrupt Source Identify Register (IRQ0_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: BOD_INT

IRQ1(WDT) Interrupt Source Identify Register (IRQ1_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (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		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: WDT_INT

IRQ2(ENIT0) Interrupt Source Identify Register (IRQ2_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) 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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: INT0_INT

IRQ3(ENIT1) Interrupt Source Identify Register (IRQ3_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) 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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: INT1_INT

IRQ4(GPA/B) Interrupt Source Identify Register (IRQ4_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/B) 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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: GPB_INT Bit0: GPA_INT

IRQ5(ALC) Interrupt Source Identify Register (IRQ5_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (ALC) 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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: ALC_INT

IRQ6(PWM0) Interrupt Source Identify Register (IRQ6_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: PWM_INT

IRQ8(TMR0) Interrupt Source Identify Register (IRQ8_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) 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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: TMR0_INT

IRQ9(TMR1) Interrupt Source Identify Register (IRQ9_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) 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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: TMR1_INT

IRQ11(UART1) Interrupt Source Identify Register (IRQ11_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (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		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: UART1_INT

IRQ12(UART0) Interrupt Source Identify Register (IRQ12_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: UART0_INT

IRQ13(SPI1) Interrupt Source Identify Register (IRQ13_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (SPI1) Interrupt Source Identity Register	0xXXXX_XXXX

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">INT_SRC</td></tr>
</table>

Bits	Description	
[31:3]	**Reserved**	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: SPI1 INT

IRQ14(SPI0) Interrupt Source Identify Register (IRQ14_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: SPI0_INT

IRQ15(DPWM) Interrupt Source Identify Register (IRQ15_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (DPWM) 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**		

Bits	Description	
[31:3]	**Reserved**	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: Bit0: DPWM INT

IRQ18(I2C0) Interrupt Source Identify Register (IRQ18_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: I2C0_INT

IRQ21(CMP) Interrupt Source Identify Register (IRQ21_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (CMP) 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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: CMP_INT

IRQ22(MAC) Interrupt Source Identify Register (IRQ22_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (MAC) 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**		

Bits	Description	
[31:3]	**Reserved**	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: MAC INT

IRQ25(SARADC) Interrupt Source Identify Register (IRQ25_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: SARADC_INT

IRQ26(PDMA) Interrupt Source Identify Register (IRQ26_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: PDMA_INT

IRQ27(I2S0) Interrupt Source Identify Register (IRQ27_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ27_SRC	INT_BA+0x6C	R	IRQ27 (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		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: I2S_INT

IRQ28(CAPS) Interrupt Source Identify Register (IRQ28_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (CAPS) 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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: CAPS_INT

IRQ29(ADC) Interrupt Source Identify Register (IRQ29_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) 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**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: ADC_INT

IRQ31(RTC) Interrupt Source Identify Register (IRQ31_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ31_SRC	INT_BA+0x7C	R	IRQ31 (RTC) 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		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: RTC_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_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**
IRQ_TM	**Reserved**		**NMI_SEL**				

Bits	Description	
[31:7]	Reserved	Reserved
[7]	**IRQ_TM**	**IRQ Test Mode** If set to 1 then peripheral IRQ signals (0-31) are replaced by the value in the MCU_IRQ register. This is a protected register to program first issue the unlock sequence.
[4:0]	**NMI_SEL**	**NMI Source Interrupt Select** The NMI interrupt to Cortex-M0 can be selected from one of the interrupt[31:0] The NMI_SEL bit[4:0] used to select the NMI interrupt source

MCU Interrupt Request Source Test Mode Register (MCU_IRQ)

Register	Offset	R/W	Description	Reset Value
MCU_IRQ	INT_BA+0x84	R/W	MCU IRQ Number Identify Register	0x0000_0000

31	30	29	28	27	26	25	24
RTC	Reserved	SDADC	CAPS	I2S0	PDMA	SARADC	Reserved
23	22	21	20	19	18	17	16
Reserved	MAC	CMP	Reserved	Reserved	I2C0	Reserved	Reserved
15	14	13	12	11	10	9	8
DPWM	SPI0	SPI1	UART0	UART1	Reserved	TMR1	TMR0
7	6	5	4	3	2	1	0
Reserved	PWM0	ALC	GPAB	EINT1	EINT0	WDT	BOD

Bits	Description	
[31]	**RTC**	**IRQ31 (RTC) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[30]	**Reserved**	**IRQ30 (RESERVED) Interrupt Source Identity Register**
[29]	**SDADC**	**IRQ29 (SDADC) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[28]	**CAPS**	**IRQ28 (CAPS) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[27]	**I2S**	**IRQ27 (I2S0) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[26]	**PDMA**	**IRQ26 (PDMA) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[25]	**SARADC**	**IRQ25 (SARADC) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[24]	**Reserved**	**IRQ24 (RESERVED) Interrupt Source Identity Register**
[23]	**Reserved**	**IRQ23 (RESERVED) Interrupt Source Identity Register**

[22]	**MAC**	**IRQ22 (MAC) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[21]	**CMP**	**IRQ21 (CMP) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[20]	**Reserved**	**IRQ20 (RESERVED) Interrupt Source Identity Register**
[19]	**Reserved**	**IRQ19 (RESERVED) Interrupt Source Identity Register**
[18]	**I2C**	**IRQ18 (I2C0) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[17]	**Reserved**	**IRQ17 (RESERVED) Interrupt Source Identity Register**
[16]	**Reserved**	**IRQ16 (RESERVED) Interrupt Source Identity Register**
[15]	**DPWM**	**IRQ15 (DPWM) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[14]	**SPI0**	**IRQ14 (SPI0) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[13]	**SPI1**	**IRQ13 (SPI1) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[12]	**UART0**	**IRQ12 (UART0) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[11]	**UART1**	**IRQ11 (UART1) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[10]	**Reserved**	**IRQ10 (RESERVED) Interrupt Source Identity Register**
[9]	**TMR1**	**IRQ9 (TMR1) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[8]	**TMR0**	**IRQ8 (TMR0) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[7]	**Reserved**	**IRQ7 (RESERVED) Interrupt Source Identity Register**

[6]	**PWM**	**IRQ6 (PWM0) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[5]	**ALC**	**IRQ5 (ALC) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[4]	**GPAB**	**IRQ4 (GPA/B) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[3]	**EINT1**	**IRQ3 (EINT1) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[2]	**EINT0**	**IRQ2 (EINT0) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[1]	**WDT**	**IRQ1 (WDT) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt
[0]	**BOD**	**IRQ0 (BOD) Interrupt Source Identity Register** 0: No effect. 1: clear the interrupt

5.2.8 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

Note: In BSP register structure, the prefix is structure name, and register will be no prefix, for example SYSINFO_ is the prefix, SYSINFO_SHPR3 will be SYSINFO->SHPR3

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.
[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).
[20:29]	**Reserved**	Reserved.
[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.
[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.

[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.
[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.

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.
[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.
[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.

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.

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.
[23:22]	**PRI14**	**Priority of System Handler 14 – PendSV** “0” denotes the highest priority and “3” denotes lowest priority
[21:0]	**Reserved**	Reserved.

5.3 Clock Controller and Power Management Unit (PMU)

The clock controller generates the clock sources for the whole device, including all AMBA interface modules and all peripheral clocks. Clock gating is provided on all peripheral clocks to minimize power consumption. The Power Management Unit (PMU) implements power control functions which can place the device into various power saving modes. The device will enter these various modes by requesting a power mode then requesting the Cortex-M0 to execute the WFI or the WFE instruction.

5.3.1 Clock Generator

The clock generator consists of 4 sources listed below:

- An internal programmable high frequency oscillator (HIRC) factory trimmed to provide frequencies of 49.152MHz and 32.768MHz to 1% accuracy.
- An external 32kHz crystal oscillator (LXT)
- An internal low power 10 kHz oscillator (LIRC)
- An external 12MHz crystal oscillator (HXT)

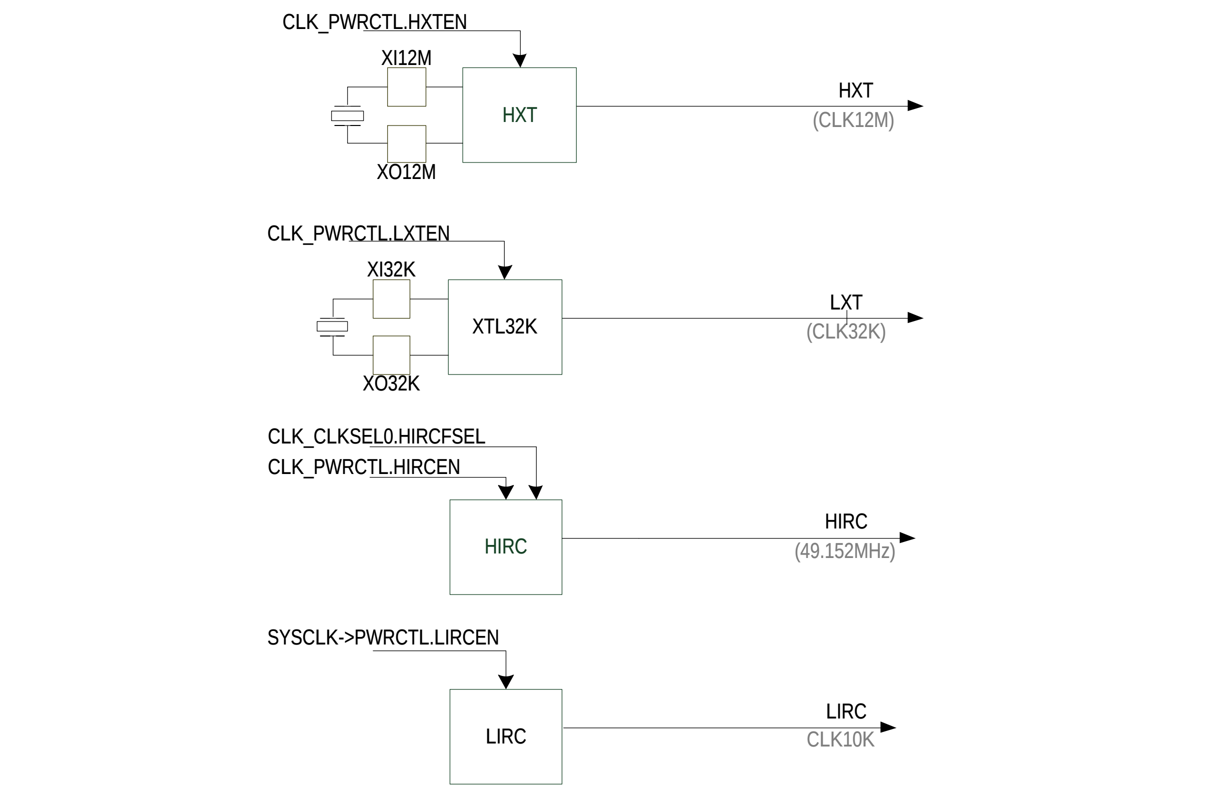


Figure 5-3 Clock generator block diagram

5.3.2 System Clock & SysTick Clock

The system clock has 4 clock sources from clock generator block. The clock source switch depends on the register HCLKSEL (CLK_CLKSEL0[2:0]). The clock is then divided by HCLKDIV+1 to produce the master clock for the device.

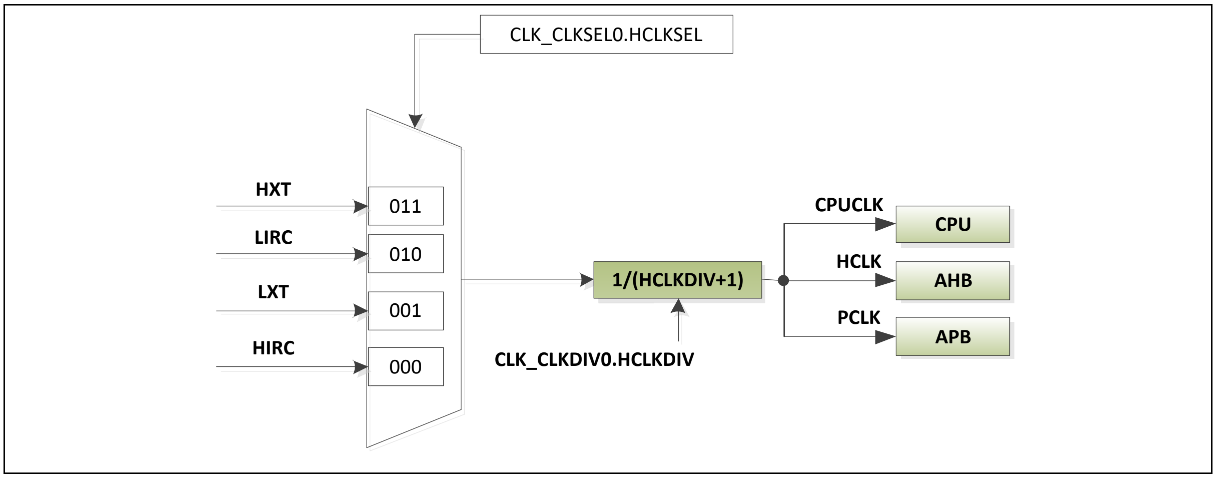


Figure 5-4 System Clock Block Diagram

The SysTick clock (STCLK) has five clock sources. The clock source switch depends on the setting of the register STCLKSEL (CLK_CLKSEL0[5:3]).

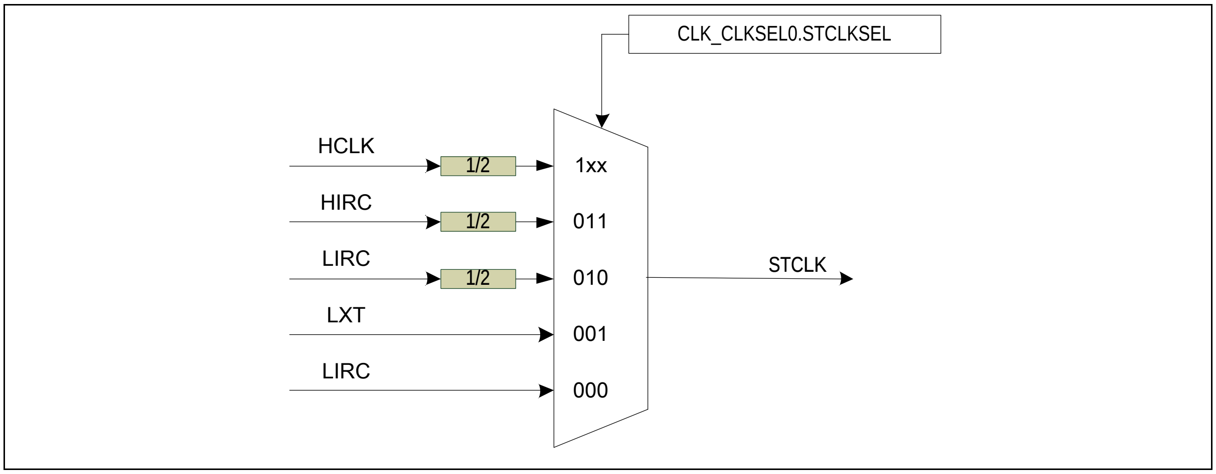


Figure 5-5 SysTick Clock Control Block Diagram

5.3.3 Peripheral Clocks

Each peripheral has a selectable clock gate. The register CLK_APBCLK0 determines whether the clock is active for each peripheral. In addition, the CLK_SLEEPCTL register determines whether these clocks remain on during M0 sleep mode. Certain peripheral clocks have selectable sources these are controlled by the CLK_CLKSEL1 & CLK_CLKSEL2 register.

5.3.4 Power Management

The ISD91200 is equipped with a Power Management Unit (PMU) that implements a variety of power saving modes. There are five levels of power control with increasing functionality (and power consumption):

- Level0 : Deep Power Down (DPD)
- Level1 : Standby Power Down (SPD)
- Level2: STOP
- Level3 : Deep Sleep
- Level4 : Sleep
- Level5 : Normal Operation

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

5.3.4.1 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 WAKEUP pin.
2. A timed wakeup where the LIRC 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 CLK_DPDSTATE [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:

- CLK_PWRCTL.WKPINEN: If set to ‘1’ then the WAKEUP pin is disabled and will not wake up the chip.
- CLK_PWRCTL.LIRCDPDEN: If set to ‘1’ then the LIRC will power down in DPD. No timed wakeup is possible.
- CLK_WAKE10K.SELWKTMR: Each bit in this register will trigger a wakeup event after a certain number of LIRC cycles.

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 CLK_PWRCTL.WKPINWKF, CLK_PWRCTL.TMRWKF and CLK_PWRCTL.PORWKF.

To enter the DPD state the user must set the register bit CLK_PWRCTL.DPDEN 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. A register bit, CLK_DBGPD.DISPDREQ is included for this purpose that will disable power down features. A check such as:

```
void Reset_Handler(void){
/*  check ICE_CLK and ICE_DAT to disable power down to the chip */
      if (CLK_DBGPD.ICECLKST == 0 && CLK_DBGPD.ICEDATST == 0)
            CLK_DBGPD.DISPDREQ = 1;
      __main();
}
```

Can check the SWD pin state on boot and prevent power down from occurring.

5.3.4.2 Level1: Standby Power Down (SPD) mode.

Standby Power Down mode is the lowest power state that some logic operation can be performed. In this mode power is removed from the majority of the core logic, including the Cortex-M0 and main RAM. A low power standby reference is enabled however that supplies power to a subset of logic including the IO ring, GPIO control, RTC module, 32kHz Crystal Oscillator, Brownout Detector and a 256Byte Standby RAM.

In Standby mode there are three ways to wake up the device:

1. An interrupt from the GPIO block, for instance a pin transition.
2. An interrupt from the RTC module, for instance an alarm or timer event.
3. A power cycle of main chip supply triggering a POR event.

When a wake up event occurs the PMU will start the Cortex-M0 processor and execute the reset vector. Software can determine whether the device woke up from SPD by interrogating the register bit CLK_PWRSTSF.SPDF.

To enter the SPD state the user must set the register bit CLK_PWRCTL.SPDEN then execute a WFI or WFE instruction. Note that when debug interface is active, device will not enter SPD. Also once device enters SPD the debug interface will be inactive.

5.3.4.3 Level2:STOP mode (special characteristic).

STOP mode is the lowest power state that some logic operation can be performed with full SRAM retention. In this mode power is removed from the majority of the core logic, including the Cortex-M0. A low power standby reference is enabled however that supplies power to a subset of logic including the IO ring, GPIO control, RTC module, LXT 32kHz Crystal Oscillator (if enabled), Brownout Detector, LIRC 10kHz Oscillator, and SRAM.

In STOP mode there are three ways to wake up the device:

1. An interrupt from the GPIO block, for instance a pin transition.
2. An interrupt from the RTC module, for instance an alarm or timer event.
3. A power cycle of main chip supply triggering a POR event.

When a wake up event occurs the PMU will start the Cortex-M0 processor and execute the next instrument. Software can determine whether the device woke up from STOP by interrogating the register bit CLK_PWRSTSF.STOPF.

To enter the STOP state the user must set the register bit CLK_PWRCTL.STOPEN 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.

5.3.4.4 Level3: Deep Sleep mode.

The Deep Sleep mode is the lowest power state where the Cortex-M0 and all logic state are preserved. In Deep Sleep mode the HIRC oscillator is shut down and a low speed oscillator is selected, if LXT is active this source is selected, if not then LIRC is enabled and selected. All clocks to the Cortex-M0 core are gated eliminating dynamic power in the core. Clocks to peripheral are gated according to the CLK_SLEEPCTL register, note however that HCLK is operating at a low frequency and HIRC is not available. Deep Sleep mode is entered by setting System Control register bit 2: SYSINFO_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 CLK_PWRSTSF.DSF.

5.3.4.5 Level4: Sleep mode.

The Sleep mode gates all clocks to the Cortex-M0 eliminating dynamic power in the core. In addition, clocks to peripherals are gated according to the CLK_SLEEPCTL register. 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 ADC, once PDMA has finished the core can be woken up to process the data

5.3.5 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	0xXX00_000D
CLK_AHBCLK	CLK_BA + 0x04	R/W	AHB Device Clock Enable Control Register	0x0000_0005
CLK_APBCLK0	CLK_BA + 0x08	R/W	APB Device Clock Enable Control Register	0x0000_0001
CLK_DPDSTATE	CLK_BA + 0x0C	R/W	Deep Power Down State Register	0x0000_XX00
CLK_CLKSEL0	CLK_BA + 0x10	R/W	Clock Source Select Control Register 0	0x0000_0038
CLK_CLKSEL1	CLK_BA + 0x14	R/W	Clock Source Select Control Register 1	0xF000_7703
CLK_CLKDIV0	CLK_BA + 0x18	R/W	Clock Divider Number Register	0x0000_1010
CLK_CLKSEL2	CLK_BA + 0x1C	R/W	Clock Source Select Control Register 2	0x0000_0000
CLK_SLEEPCTL	CLK_BA + 0x20	R/W	Sleep Clock Source Select Register	0xFFFF_FFFF
CLK_PWRSTSF	CLK_BA + 0x24	R/W	Power State Flag Register	0x0000_0000
CLK_DBGPD	CLK_BA + 0x28	R/W	Debug Port Power Down Disable Register	0x0000_00XX
CLK_WAKE10K	CLK_BA + 0x2C	R/W	Deep Power Down 10K Wakeup Timer	0x0000_0001

Note: In BSP register structure, the prefix is structure name, and register be no prefix, for example CLK_ is the prefix, CLK_WAKE10K will be CLK->WAKE10K

5.3.6 Register Description

System Power Control Register (CLK_PWRCTL)

This is a protected register, to write to register, first issue the unlock sequence.

Register	Offset	R/W	Description	Reset Value
CLK_PWRCTL	CLK_BA + 0x00	R/W	System Power Control Register	0xXX00_000D

31	30	29	28	27	26	25	24
Reserved				WKPUEN	PORWKF	TMRWKF	WKPINWKF
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				FLASHEN		LIRCDPDEN	WKPINEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	IOSTATE	RELEASEIO	HOLDIO	DPDEN	SPDEN	STOPEN	Reserved
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			HXTEN	LIRCEN	HIRCEN	LXTEN	Reserved

Bits	Description	
[31:27]	**Reserved**	Reserved.
[27]	**WKPUEN**	**Wakeup Pin Pull-up Control** This signal is latched in deep power down and preserved. 0 = pull-up enable. 1 = tri-state (default).
[26]	**PORWKF**	**POR Wakeup Flag** Read Only. This flag indicates that wakeup of device was requested with a power-on reset. Flag is cleared when DPD mode is entered.
[25]	**TMRWKF**	**Timer Wakeup Flag** 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.
[24]	**WKPINWKF**	**Pin Wakeup Flag** 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.
[23:20]	**Reserved**	Reserved.
[19:18]	**FLASHEN**	Flash ROM Control Enable/Disable Bit [19]: for Stop mode operation Bit [18]: for Sleep mode operation 1: Turn off flash 0: Normal Note: It takes 10us to turn on the flash to normal

[17]	**LIRCDPDEN**	**OSC10k Enabled Control** Determines whether OSC10k is enabled in DPD mode. If OSC10k is disabled, device cannot wake from DPD with SELWKTMR delay. 0 = enabled. 1 = disabled.
[16]	**WKPINEN**	**Wakeup Pin Enabled Control** Determines whether WAKEUP pin is enabled in DPD mode. 0 = enabled. 1 = disabled.
[15]	**Reserved**	Reserved.
[14]	**IOSTATE**	‘1’: IO held from SPD ‘0’: IO released.
[13]	**RELEASEIO**	Write ‘1’ to this bit to release IO state after exiting SPD if hold request was made with the HOLD_IO bit.
[12]	**HOLDIO**	**When entering SPD mode, IO state is automatically held** If this bit is set to ‘1’ then this sate upon resuming full power mode will be hold until the RELEASE_IO bit is written ‘1’
[11]	**DPDEN**	**Deep Power Down (DPD) Bit** Set to ‘1’ and issue WFI/WFE instruction to enter DPD mode.
[10]	**SPDEN**	**Standby Power Down (SPD) Bit** Set to ‘1’ and issue WFI/WFE instruction to enter SPD mode.
[9]	**STOPEN**	**Stop** Set to ‘1’ and issue WFI/WFE instruction to enter STOP mode.
[8:5]	**Reserved**	Reserved.
[4]	**HXTEN**	**HXT Oscillator Enable Bit** 0 = disable (default). 1 = enable.
[3]	**LIRCEN**	**LIRC Oscillator Enable Bit** 0 = disable. 1 = enable (default).
[2]	**HIRCEN**	**HIRC Oscillator Enable Bit** 0 = disable. 1 = enable (default).
[1]	**LXTEN**	**External 32.768 KHz Crystal Enable Bit** 0 = disable (default). 1 = enable.
[0]	**Reserved**	Reserved.

AHB Device Clock Enable Control Register (CLK_AHBCLK)

These register bits are used to enable/disable the clock source for AHB (Advanced High-Performance Bus) blocks. This is a protected register, to write to register, first issue the unlock sequence.

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**	**PDMACKEN**	**HCLKEN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ISPCKEN**	**Flash ISP Controller Clock Enable Control** 0 = To disable the Flash ISP engine clock. 1 = To enable the Flash ISP engine clock.
[1]	**PDMACKEN**	**PDMA Controller Clock Enable Control** 0 = To disable the PDMA engine clock. 1 = To enable the PDMA engine clock.
[0]	**HCLKEN**	**CPU Clock Enable (HCLK)** Must be left as 1 for normal operation.

APB Device Clock Enable Control Register (CLK_APBCLK0)

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_APBCLK0	CLK_BA + 0x08	R/W	APB Device Clock Enable Control Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved	ANACKEN	I2S0CKEN	SDADCCKEN	Reserved			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		PWM0CH23CKEN	PWM0CH01CKEN	Reserved	BIQALCKEN	SARADCKEN	UART0CKEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UART1CKEN	Reserved	DPWMCKEN	SPI0CKEN	SPI1CKEN	Reserved		I2C0CKEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TMR1CKEN	TMR0CKEN	RTCCKEN	Reserved				WDTCKEN

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**ANACKEN**	**Analog Block Clock Enable Control** 0=Disable. 1=Enable.
[29]	**I2S0CKEN**	**I2S Clock Enable Control** 0=Disable. 1=Enable.
[28]	**SDADCCKEN**	**Delta-Sigma Analog-digital-converter (ADC) Enable Control** 0=Disable. 1=Enable.
[27:22]	**Reserved**	Reserved.
[21]	**PWM0CH23CKEN**	**PWM0CH2 and PWM0CH3 Block Clock Enable Control** 0=Disable. 1=Enable.
[20]	**PWM0CH01CKEN**	**PWM0CH0 and PWM0CH1 Block Clock Enable Control** 0=Disable. 1=Enable.
[19]	**Reserved**	Reserved.
[18]	**BIQALCKEN**	**BIQ and ALC Clock Enable Control** 0=Disable. 1=Enable.

[17]	**SARADCKEN**	**SAR Analog-digital-converter (ADC) Clock Enable Control** 0=Disable. 1=Enable.
[16]	**UART0CKEN**	**UART0 Clock Enable Control** 0=Disable. 1=Enable.
[15]	**UART1CKEN**	**UART1 Clock Enable Control** 0=Disable. 1=Enable.
[14]	**Reserved**	Reserved.
[13]	**DPWMCKEN**	**Differential PWM Speaker Driver Clock Enable Control** 0=Disable. 1=Enable.
[12]	**SPI0CKEN**	**SPI0 Clock Enable Control** 0=Disable. 1=Enable.
[11]	**SPI1CKEN**	**SPI1 Clock Enable Control** 0=Disable 1=Enable
[10:9]	**Reserved**	Reserved.
[8]	**I2C0CKEN**	**I2C0 Clock Enable Control** 0=Disable. 1=Enable.
[7]	**TMR1CKEN**	**Timer1 Clock Enable Control** 0=Disable. 1=Enable.
[6]	**TMR0CKEN**	**Timer0 Clock Enable Control** 0=Disable. 1=Enable.
[5]	**RTCCKEN**	**Real-time-clock APB Interface Clock Control** 0=Disable. 1=Enable.
[4:1]	**Reserved**	Reserved.
[0]	**WDTCKEN**	**Watchdog Clock Enable Control** 0=Disable. 1=Enable.

DPD State Register (CLK_DPDSTATE)

The Deep Power Down State register is a user settable register that is preserved during Deep Power Down (DPD). Software can use this register to store a single byte during a DPD event. The DPDSTSRD register reads back the current state of the CLK_DPDSTATE register. To write to this register, set desired value in the DPDSTSWR register, this value will be latched in to the CLK_DPDSTATE register on next DPD event.

Register	Offset	R/W	Description	Reset Value
CLK_DPDSTATE	CLK_BA + 0x0C	R/W	Deep Power Down State Register	0x0000_XX00

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**
DPDSTSRD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DPDSTSWR							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**DPDSTSRD**	**DPD State Read Back** Read back of CLK_DPDSTATE register. This register was preserved from last DPD event .
[7:0]	**DPDSTSWR**	**DPD State Write Register** Read back of CLK_DPDSTATE register. This register was preserved from last DPD event .

Clock Source Select Control Register 0 (CLK_CLKSEL0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL0	CLK_BA + 0x10	R/W	Clock Source Select Control Register 0	0x0000_0038

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**
HIRCFSEL		**STCLKSEL**			**HCLKSEL**		

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:6]	**HIRCFSEL**	**High Frequency RC Oscilltor Frequency Select Register.** These bits are protected, to write to bits first perform the unlock sequence. 00 = Trim for 49.152MHz selected. 01 = Trim for 32.768MHz selected. 10 = Trim for reserved.
[5:3]	**STCLKSEL**	**MCU Cortex_M0 SYST Clock Source Select** These bits are protected, to write to bits first perform the unlock sequence. 000 = clock source from LIRC. 001 = clock source from LXT. 010 = clock source from LIRC divided by 2. 011 = clock source from HIRC divided by 2. 1xx = clock source from HCLK÷2 (Default). Note that to use STCLKSEL as source of SysTic timer the CLKSRC bit of SYST_CSR must be set to 0.
[2:0]	**HCLKSEL**	**HCLK Clock Source Select** Ensure that related clock sources (pre-select and new-select) are enabled before updating register. These bits are protected, to write to bits first perform the unlock sequence. 000 = clock source from HIRC. (deafult) 001 = clock source from LXT. 010 = clock source from LIRC. 011 = clock source from HXT. Others = Reserved.

Clock Source Select Control Register 1 (CLK_CLKSEL1)

Clock multiplexors are a glitch free design to ensure smooth transitions between asynchronous clock sources. As such, both the current clock source and the target clock source must be enabled for switching to occur. Beware when switching from a low speed clock to a high speed clock that low speed clock remains on for at least one period before disabling.

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL1	CLK_BA + 0x14	R/W	Clock Source Select Control Register 1	0xF000_7703

31	30	29	28	27	26	25	24
PWM0CH23SEL		PWM0CH01SEL		Reserved		SARADCSEL	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	TMR1SEL			Reserved	TMR0SEL		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		DPWMSEL		ADCSEL		WDTSEL	

Bits	Description	
[31:30]	**PWM0CH23SEL**	**PWM0CH23 Clock Source Select** PWM0 CH2 and CH3 uses the same clock source, and pre-scaler 00 = clock source from LIRC. 01 = clock source from LXT. 10 = clock source from HCLK. 11 = clock source from HIRC.(default)
[29:28]	**PWM0CH01SEL**	**PWM0CH01 Clock Source Select** PWM0 CH0 and CH1 uses the same clock source, and pre-scaler 00 = clock source from LIRC. 01 = clock source from LXT. 10 = clock source from HCLK. 11 = clock source from HIRC.(default)
[27:26]	**Reserved**	Reserved.
[25:24]	**SARADCSEL**	**SAR ADC Clock Source Select** 00 = clock source from HCLK (default) 01 = clock source from LIRC. 10 = clock source from HIRC. 11 = clock source from LXT.
[23:15]	**Reserved**	Reserved.

[14:12]	**TMR1SEL**	**TIMER1 Clock Source Select** 000 = clock source from LIRC. 001 = clock source from LXT. 010 = clock source from HXT. 011 = clock source from external pin (GPIOA[11]). 1xx = clock source from HCLK.(default)
[11]	**Reserved**	Reserved.
[10:8]	**TMR0SEL**	**TIMER0 Clock Source Select** 000 = clock source from LIRC. 001 = clock source from LXT. 010 = clock source from HXT. 011 = clock source from external pin (GPIOA[10]). 1xx = clock source from internal HCLK.(default)
[7:6]	**Reserved**	Reserved.
[5:4]	**DPWMSEL**	**Differential Speaker Driver PWM Clock Source Select** 00 = clock source from HCLK/(DPWMDIV+1). (default) 01 = clock source from HXT 1x = Reserved.
[3:2]	**SDADCSEL**	**SD ADC Clock Source Select** (output is MCLK after clock enable) 00 = clock source from HCLK/(SDADCDIV+1). (default) 01 = clock source from HXT 1x = Reserved.
[1:0]	**WDTSEL**	**WDT Clock Source Select** 00 = clock source from HIRC. 01 = clock source from LXT. 10 = clock source from HCLK/2048 clock. 11 = clock source from LIRC.(default)

Clock Divider Register (CLK_CLKDIV0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV0	CLK_BA + 0x18	R/W	Clock Divider Number Register	0x0000_1010

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

Bits	Description	
[31:24]	**SARADCDIV**	**SARADC Clock Divider** The SARADC clock frequency = (SARADC clock source frequency) / (SARADCDIV + 1).
[23:16]	**SDADCDIV**	**SDADC Clock Source Divider** The SDADC clock source frequency = (HCLK frequency) / (SDADCDIV + 1)
[15:12]	**DPWMDIV**	**DPWM Clock Source Divider** The DPWM clock source frequency = (HCLK frequency) / (DPWMDIV + 1).
[11:8]	**UARTDIV**	**UART Clock Divider** The UART clock frequency = (HCLK frequency) / (UARTDIV + 1).
[7:4]	**BIQDIV**	**BIQ Clock Divider** The BIQ clock frequency = (HCLK frequency) / (BIQDIV + 1). Suggestion: BIQDIV =1, if HCLK is 49MHz.
[3:0]	**HCLKDIV**	**HCLK Clock Divider** The HCLK clock frequency = (System clock source frequency) / (HCLKDIV + 1).

Clock Source Select Control Register 2 (CLK_CLKSEL2)

Before changing clock source, ensure that related clock sources (pre-select and new-select) are enabled.

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL2	CLK_BA + 0x1C	R/W	Clock Source Select Control Register 2	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				**UART1DIV**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**I2S0SEL**	

Bits	Description	
[32:12]	**Reserved**	Reserved.
[11:8]	**UART1DIV**	**UART1 Clock Divide Number From UART Clock Source** The UART clock frequency = (UART clock source frequency) / (UART1DIV + 1).
[7:2]	**Reserved**	Reserved.
[1:0]	**I2S0SEL**	**I2S0 Clock Source Select** 00 = clock source from internal 10kHz oscillator.(default) 01 = clock source from external 32kHz crystal clock. 10 = clock source from HCLK 11 = clock source from HIRC.

Sleep Clock Enable Control Register (CLK_SLEEPCTL)

These register bits are used to enable/disable clocks during sleep mode. It works in conjunction with CLK_AHBCLK and CLK_APBCLK0 clock register to determine whether a clock source remains active during CPU Sleep mode. For a clock to be active in Sleep mode, the appropriate clock must be enabled in the CLK_AHBCLK or CLK_APBCLK0 register and the bit must also be enabled in the CLK_SLEEPCTL register. In other words, to disable a clock in Sleep mode, write '0' to the appropriate bit in CLK_SLEEPCTL.

Register	Offset	R/W	Description	Reset Value
CLK_SLEEPCTL	CLK_BA + 0x20	R/W	Sleep Clock Source Select Register	0xFFFF_FFFF

31	30	29	28	27	26	25	24
Reserved	**ANACKEN**	**I2SCKEN**	**SDADCCKEN**	**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**PWM0CH23CKEN**	**PWM0CH01CKEN**	**Reserved**	**BQALCKEN**	**SARADCCKEN**	**UART0CKEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UART1CKEN	**Reserved**	**DPWMCKEN**	**SPI0CKEN**	**SPI1CKEN**	**Reserved**	**Reserved**	**I2C0CKEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TMR1CKEN	**TMR0CKEN**	**RTCCKEN**	**WDTCKEN**	**Reserved**	**ISPCKEN**	**PDMACKEN**	**HCLKCKEN**

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**ANACKEN**	**Analog Block Sleep Clock Enable Control** 0=Disable. 1=Enable.
[29]	**I2SCKEN**	**I2S Sleep Clock Enable Control** 0=Disable. 1=Enable.
[28]	**SDADCCKEN**	**Delta-Sigma Analog-digital-converter (ADC) Sleep Clock Enable Control** 0=Disable. 1=Enable.
[27:22]	**Reserved**	Reserved.
[21]	**PWM0CH23CKEN**	**PWM0CH2 and PWM0CH3 Block Sleep Clock Enable Control** 0=Disable. 1=Enable.
[20]	**PWM0CH01CKEN**	**PWM0CH0 and PWM0CH1 Block Sleep Clock Enable Control** 0=Disable. 1=Enable.
[19]	**Reserved**	Reserved.

[18]	**BIQALCKEN**	**BIQ and ALCSleep Clock Enable Control** 0=Disable. 1=Enable.
[17]	**SARADCCKEN**	**SARADC Sleep Clock Enable Control** 0=Disable. 1=Enable.
[16]	**UART0CKEN**	**UART0 Sleep Clock Enable Control** 0=Disable. 1=Enable.
[15]	**UART1CKEN**	**UART1 Sleep Clock Enable Control** 0=Disable. 1=Enable.
[14]	**Reserved**	Reserved.
[13]	**DPWMCKEN**	**Differential PWM Speaker Driver Sleep Clock Enable Control** 0=Disable. 1=Enable.
[12]	**SPI0CKEN**	**SPI0 Sleep Clock Enable Control** 0=Disable. 1=Enable.
[11]	**SPI1CHEN**	**SPI1 Sleep Clock Enable Control** 0=Disable. 1=Enable.
[10:9]	**Reserved**	Reserved.
[8]	**I2C0CKEN**	**I2C0 Sleep Clock Enable Control** 0=Disable. 1=Enable.
[7]	**TMR1CKEN**	**Timer1 Sleep Clock Enable Control** 0=Disable. 1=Enable.
[6]	**TMR0CKEN**	**Timer0 Sleep Clock Enable Control** 0=Disable. 1=Enable.
[5]	**RTCCKEN**	**Real-time- Sleep Clock APB Interface Clock Control** 0=Disable. 1=Enable.
[4]	**WDTCKEN**	**Watchdog Sleep Clock Enable Control** 0=Disable. 1=Enable.
[2]	**ISPCKEN**	**Flash ISP Controller Sleep Clock Enable Control** 0=Disable. 1=Enable.

[1]	**PDMACKEN**	**PDMA Controller Sleep Clock Enable Control** 0=Disable. 1=Enable.
[0]	**HCLKCKEN**	**CPU Clock Sleep Enable (HCLK)** Must be left as ‘1’ for normal operation. 0=Disable. 1=Enable.

Power State Flag Register (CLK_PWRSTSF)

Register	Offset	R/W	Description	Reset Value
CLK_PWRSTSF	CLK_BA + 0x24	R/W	Power State 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					SPDF	STOPF	DSF

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**SPDF**	**Powered Down Flag** This flag is set if core logic was powered down to Standby (SPD). Write '1' to clear flag.
[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.

Debug Power Down Register (CLK_DBGPD)

Register	Offset	R/W	Description	Reset Value
CLK_DBGPD	CLK_BA + 0x28	R/W	Debug Port Power Down Disable Register	0x0000_00XX

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**
ICEDATST	ICECLKST	Reserved					DISPDREQ

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**ICEDATST**	**ICEDATST Pin State** Read Only. Current state of ICE_DAT pin.
[6]	**ICECLKST**	**ICECLKST Pin State** Read Only. Current state of ICE_CLK pin.
[5:1]	**Reserved**	Reserved.
[0]	**DISPDREQ**	**Disable Power Down** 0 = Enable power down requests. 1 = Disable power down requests.

Deep Power Down 10K Wakeup Timer (CLK_WAKE10K)

Register	Offset	R/W	Description	Reset Value
CLK_WAKE10K	CLK_BA + 0x2C	R/W	Deep Power Down 10K Wakeup Timer	0x0000_0001

31	30	29	28	27	26	25	24
WAKE10KEN	**Reserved**	**WKTMRSTS**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
WKTMRSTS							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**SELWKTMR**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SELWKTMR							

Bits	Description	
[31]	**WAKE10KEN**	**Enable WAKE from DPD on 10kHz timer**
[30]	**Reserved**	Reserved.
[29:16]	**WKTMRSTS**	**Current Wakeup Timer Setting** Read-Only. Read back of the current WAKEUP timer setting. This value is updated with SELWKTMR upon entering DPD mode.
[15:14]	**Reserved**	Reserved.
[13:0]	**SELWKTMR**	**Select Wakeup Timer** WAKEUP after 64* (SELWKTMR+1) OSC10k clocks (6.4 * (SELWKTMR+1) ms)

5.4 General Purpose I/O

5.4.1 Overview and Features

Up to 32 General Purpose I/O pins are available on the ISD91200 series. These are shared peripheral special function pins under control of the alternate configuration registers. These 32 pins are arranged in 2 ports named with GPIOA, and GPIOB. Each one of the 32 pins is independent and has corresponding register bits to control the pin mode function and data.

The I/O type of each GPIO pin can be independently configured as an input, output, open-drain or in a quasi-bidirectional mode. Upon chip reset, all GPIO pins are configured in quasi-bidirectional mode and port data register resets high.

When device is in deep power down (DPD) mode, all GPIO pins become high impedance.

GPIO can generate interrupt signals to the core as either level sensitive or edge sensitive inputs. Edge sensitive inputs can also be de-bounced.

In quasi-bidirectional mode, each GPIO pin has a weak pull-up resistor which is approximately 110KΩ~300KΩ for V_{DD} from 5.0V to 2.4V.

Each pin can generate and interrupt exception to the Cortex M0 core. GPIOB[0] and GPIOB[1] can generate interrupts to system interrupt number IRQ2 and IRQ3 respectively (see Table 5-4). All other GPIO generate and exception to interrupt number IRQ4.

5.4.2 GPIO I/O Modes

The I/O mode of each GPIO pin is controlled by the register Px. *(x=A or B).* Each pin has two bits of control giving four possible states:

5.4.2.1 Input Mode

For Px_MODE*n* = 00b the GPIO*x* port [*n*] pin is in Input Mode. The GPIO pin is in a tri-state (high impedance) condition without output drive capability. The Px_PIN value reflects the status of the corresponding port pins.

5.4.2.2 Output Mode

For Px_MODE*n* = 01b the GPIO*x* port [*n*] pin is in Output Mode. The GPIO pin supports a digital output function with current source/sink capability. The bit value in the corresponding bit [n] of P*x*_DOUT is driven to the pin.

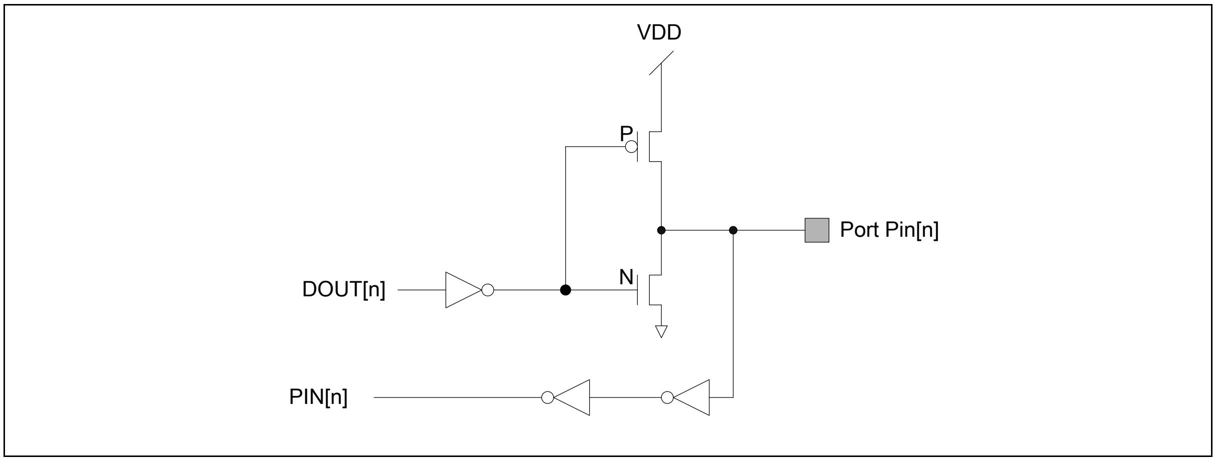


Figure 5-6 Output Mode: Push-Pull Output

5.4.2.3 *Open-Drain Mode*

For Px_MODE*n* = 10b the GPIO*x* port [*n*] pin is in Open-Drain mode. The GPIO pin supports a digital output function but only with sink current capability, an additional pull-up resister is needed for defining a high state. If the bit value in the corresponding bit [n] of Px_DOUT is “0”, pin is driven low. If the bit value in the corresponding bit [n] of Px_DOUT is “1”, the pin state is defined by the external load on the pin.

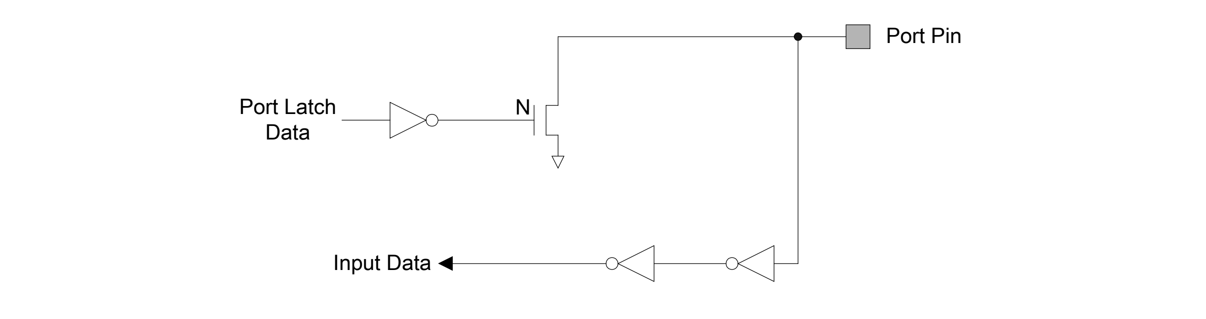


Figure 5-7 Open-Drain Output

5.4.2.4 *Quasi-bidirectional Mode Explanation*

For Px_MODE*n* = 11b the GPIO*x* port [*n*] pin is in Quasi-bidirectional mode and the I/O pin supports digital output and input function where the source current is only between 30-200uA.Before input function is performed the corresponding bit in Px_DOUT must be set to 1. The quasi-bidirectional output is common on the 80C51 and most of its derivatives. If the bit value in the corresponding bit [n] of Px_DOUT is “0”, the pin will drive a “low” output to the pin. If the bit value in the corresponding bit [n] of Px_DOUT is “1”, the pin will check the pin value. If pin value is high, no action is taken. If pin state is low, then pin will drive a strong high for2 clock cycles. After this the pin has an internal pull-up resistor connected. Note that the source current capability in quasi-bidirectional mode is approximately200uA to 30uA for VDD form 5.0V to 2.4V.

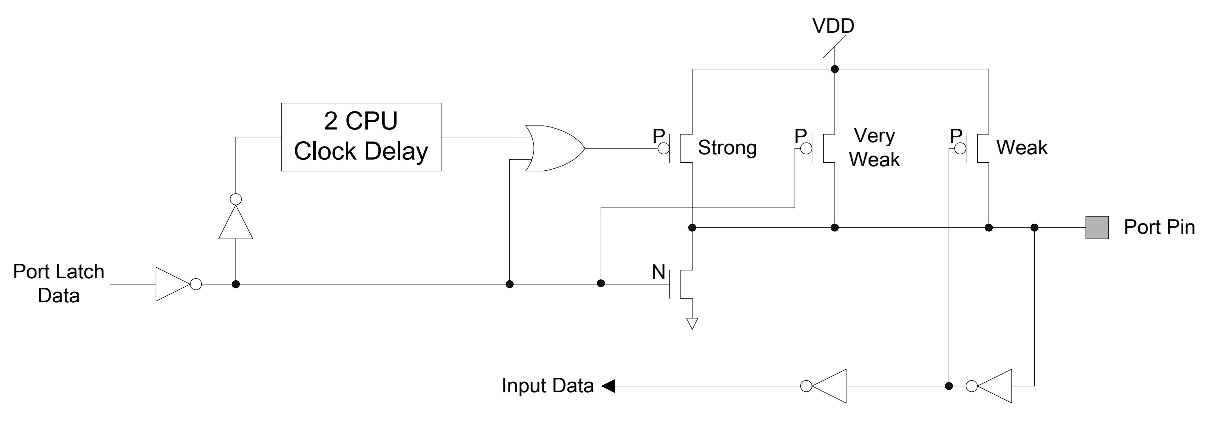


Figure 5-8 Quasi-bidirectional GPIO Mode

5.4.3 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 Port A Pin I/O Mode Control	0xFFFF_FFFF
PA_DINOFF	GPIO_BA+0x004	R/W	GPIO Port A Pin Input Disable	0x0000_0000
PA_DOUT	GPIO_BA+0x008	R/W	GPIO Port A Data Output Value	0x0000_FFFF
PA_DATMSK	GPIO_BA+0x00C	R/W	GPIO Port A Data Output Write Mask	0xXXXX_0000
PA_PIN	GPIO_BA+0x010	R	GPIO Port A Pin Value	0x0000_XXXX
PA_DBEN	GPIO_BA+0x014	R/W	GPIO Port A De-bounce Enable	0xXXXX_0000
PA_INTTYPE	GPIO_BA+0x018	R/W	GPIO Port A Interrupt Trigger Type	0xXXXX_0000
PA_INTEN	GPIO_BA+0x01C	R/W	GPIO Port A Interrupt Enable	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	GPIO Port A Interrupt Source Flag	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	GPIO Port B Pin I/O Mode Control	0xFFFF_FFFF
PB_DINOFF	GPIO_BA+0x044	R/W	GPIO Port B Pin Input Disable	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	GPIO Port B Data Output Value	0x0000_FFFF
PB_DATMSK	GPIO_BA+0x04C	R/W	GPIO Port B Data Output Write Mask	0xXXXX_0000
PB_PIN	GPIO_BA+0x050	R	GPIO Port B Pin Value	0x0000_XXXX
PB_DBEN	GPIO_BA+0x054	R/W	GPIO Port B De-bounce Enable	0xXXXX_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	GPIO Port B Interrupt Trigger Type	0xXXXX_0000
PB_INTEN	GPIO_BA+0x05C	R/W	GPIO Port B Interrupt Enable	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	GPIO Port B Interrupt Source Flag	0x0000_0000
DBCTL	GPIO_BA+0x180	R/W	Interrupt De-bounce Control	0x0000_0020

Note: In BSP register structure, GPIO is the structure name. Register is no prefix PA_ or PB_.

5.4.4 Register Description

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

Register	Offset	R/W	Description	Reset Value
PA_MODE	GPIO_BA+0x000	R/W	GPIO Port A Pin I/O Mode Control	0xFFFF_FFFF
PB_MODE	GPIO_BA+0x040	R/W	GPIO Port B Pin I/O Mode Control	0xFFFF_FFFF

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**	**Px I/O Pin[n] Mode Control** Determine each I/O type of GPIOx pins 00 = GPIO port [n] pin is in INPUT mode. 01 = GPIO port [n] pin is in OUTPUT mode. 10 = GPIO port [n] pin is in Open-Drain mode. 11 = GPIO port [n] pin is in Quasi-bidirectional mode.

GPIO Port [A/B] Input Disable (Px_DINOFF)

Register	Offset	R/W	Description	Reset Value
PA_DINOFF	GPIO_BA+0x004	R/W	GPIO Port A Pin Input Disable	0x0000_0000
PB_DINOFF	GPIO_BA+0x044	R/W	GPIO Port B Pin Input Disable	0x0000_0000

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

Bits	Description	
[31:16]	**DINOFF**	**GPIOx Pin[n] OFF Digital Input Path Enable** 0 = Enable IO digital input path (Default). 1 = Disable IO digital input path (low leakage mode).
[15:0]	**Reserved**	Reserved.

GPIO Port [A/B] Data Output Value (Px_DOUT)

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	GPIO Port A Data Output Value	0x0000_FFFF
PB_DOUT	GPIO_BA+0x048	R/W	GPIO Port B 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DOUT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DOUT**	**Px Pin[n] Output Value** Each of these bits controls the status of a GPIO pin when the GPIO pin is configured as output, open-drain or quasi-bidirectional mode. 1 = GPIO port [A/B] Pin[n] will drive High if the corresponding output mode bit is set. 0 = GPIO port [A/B] Pin[n] will drive Low if the corresponding output mode bit is set.

GPIO Port [A/B] Data Output Write Mask (Px_DATMSK)

Register	Offset	R/W	Description	Reset Value
PA_DATMSK	GPIO_BA+0x00C	R/W	GPIO Port A Data Output Write Mask	0xXXXX_0000
PB_DATMSK	GPIO_BA+0x04C	R/W	GPIO Port B Data Output Write Mask	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**
DATMSK							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATMSK							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DATMSK**	**Port [A/B] Data Output Write Mask** These bits are used to protect the corresponding register of Px_DOUT bit[n] . When set the DATMSK bit[n] to “1”, the corresponding DOUTn bit is writing protected. 0 = The corresponding Px_DOUT[n] bit can be updated. 1 = The corresponding Px_DOUT[n] bit is read only.

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

Register	Offset	R/W	Description	Reset Value
PA_PIN	GPIO_BA+0x010	R	GPIO Port A Pin Value	0x0000_XXXX
PB_PIN	GPIO_BA+0x050	R	GPIO Port B 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PIN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PIN**	**Port [A/B] Pin Values** The value read from each of these bit reflects the actual status of the respective GPIO pin

GPIO Port [A/B] De-bounce Enable (Px_DBEN)

Register	Offset	R/W	Description	Reset Value
PA_DBEN	GPIO_BA+0x014	R/W	GPIO Port A De-bounce Enable	0xXXXX_0000
PB_DBEN	GPIO_BA+0x054	R/W	GPIO Port B De-bounce Enable	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**
DBEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBEN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DBEN**	**Port [A/B] De-bounce Enable Control** DBEN[n]used to enable the de-bounce function for each corresponding bit. For an edge triggered interrupt to be generated, input signal must be valid for two consecutive de-bounce periods. The de-bounce time is controlled by the GPIO_DBCTL register. The DBEN[n] is used for “edge-trigger” interrupt only; it is ignored for “level trigger” interrupt 0 = The bit[n] de-bounce function is disabled. 1 = The bit[n] de-bounce function is enabled.

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

Register	Offset	R/W	Description	Reset Value
PA_INTTYPE	GPIO_BA+0x018	R/W	GPIO Port A Interrupt Trigger Type	0xXXXX_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	GPIO Port B Interrupt Trigger Type	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
TYPE							
7	6	5	4	3	2	1	0
TYPE							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TYPE**	**Port [A/B] Edge or Level Detection Interrupt Trigger Type** TYPE[n] used to control whether the interrupt mode is level triggered or edge triggered. If the interrupt mode is edge triggered, edge de-bounce is controlled by the DBEN register. If the interrupt mode is level triggered, the input source is sampled each clock to generate an interrupt 0 = Edge triggered interrupt. 1 = Level triggered interrupt. If level triggered interrupt is selected, then only one level can be selected in the Px_INTEN register. If both levels are set no interrupt will occur

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

Register	Offset	R/W	Description	Reset Value
PA_INTEN	GPIO_BA+0x01C	R/W	GPIO Port A Interrupt Enable	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	GPIO Port B Interrupt Enable	0x0000_0000

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

Bits	Description	
[31:16]	**RHIEN**	**Port [A/B] Interrupt Enable by Input Rising Edge or Input Level High** RHIEN[n] is used to enable the rising/high interrupt for each of the corresponding GPIO pins. It also enables the pin wakeup function. If the interrupt is configured in level trigger mode, a level “high” will generate an interrupt. If the interrupt is configured in edge trigger mode, a state change from “low-to-high” will generate an interrupt. GPB.0 and GPB.1 trigger individual IRQ vectors (IRQ2/IRQ3) while remaining GPIO trigger a single interrupt vector IRQ4. 0 = Disable GPIOx[n] for level-high or low-to-high interrupt. 1 = Enable GPIOx[n] for level-high or low-to-high interrupt.
[15:0]	**FLIEN**	**Port [A/B] Interrupt Enable by Input Falling Edge or Input Level Low** FLIEN[n] is used to enable the falling/low interrupt for each of the corresponding GPIO pins. It also enables the pin wakeup function. If the interrupt is configured in level trigger mode, a level “low” will generate an interrupt. If the interrupt is configured in edge trigger mode, a state change from “high-to-low” will generate an interrupt. GPB.0 and GPB.1 trigger individual IRQ vectors (IRQ2/IRQ3) while remaining GPIO trigger a single interrupt vector IRQ4. 0 = Disable GPIOx[n] for low-level or high-to-low interrupt. 1 = Enable GPIOx[n] for low-level or high-to-low interrupt.

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

Register	Offset	R/W	Description	Reset Value
PA_INTSRC	GPIO_BA+0x020	R/W	GPIO Port A Interrupt Source Flag	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	GPIO Port B 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTSRC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**INTSRC**	**Port [A/B] Interrupt Source Flag** Read : 1 = Indicates GPIOx[n] generated an interrupt. 0 = No interrupt from GPIOx[n]. Write : 1 = Clear the corresponding pending interrupt. 0 = No action.

Interrupt De-bounce Control (GPIO_DBCTL)

Register	Offset	R/W	Description	Reset Value
DBCTL	GPIO_BA+0x180	R/W	Interrupt De-bounce Control	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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**ICLKON**	**DBCLKSRC**	**DBCLKSEL**			

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**ICLKON**	**Interrupt Clock on Mode** Set this bit “0” will gate the clock to the interrupt generation circuit if the GPIOx[n] interrupt is disabled. 0 = disable the clock if the GPIOx[n] interrupt is disabled. 1 = Interrupt generation clock always active.
[4]	**DBCLKSRC**	**De-bounce Counter Clock Source Select** 0 = De-bounce counter clock source is HCLK. 1 = De-bounce counter clock source is the internal 16 kHz clock.
[3:0]	**DBCLKSEL**	**De-bounce Sampling Cycle Selection** For edge level interrupt GPIO state is sampled every 2^(DBCLKSEL) de-bounce clocks. For example if DBCLKSRC = 6, then interrupt is sampled every 2^6 = 64 de-bounce clocks. If DBCLKSRC is 10kHz oscillator this would be a 64ms de-bounce.

5.5 Brownout Detection and Temperature Alarm

The ISD91200 is equipped with a Brown-Out voltage detector. The Brown-Out detector features a configurable trigger level and can be configured by flash to be active upon reset. The Brown-Out detector also has a power saving mode where detection can be set up to be active for a configurable on and off time.

BOD operation require that the OSC10k low power oscillator is enabled (CLK_PWRCTL.LIRCDPDEN = 0).

5.5.1 Register Map

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

Register	Offset	R/W	Description	Reset Value
BOD Base Address: **BOD_BA = 0x4008_4000**				
BOD_BODSEL	BOD_BA+0x00	R/W	Brown Out Detector Select Register	0x0000_0000
BOD_BODCTL	BOD_BA+0x04	R/W	Brown Out Detector Enable Register	0x000X_000X
BOD_BODDTMR	BOD_BA+0x10	R/W	Brown Out Detector Timer Register	0x0003_03E3

5.5.2 Register Description

Brown-Out Detector Select Register (BOD_BODSEL)

This register is initialized by user flash configuration bits CONFIG0[22:18].

Register	Offset	R/W	Description	Reset Value
BOD_BODSEL	BOD_BA+0x00	R/W	Brown Out Detector 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			**BODHYS**	**BODVL**			

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**BODHYS**	**BOD Hysteresis** 0 = Hysteresis Disabled. 1 = Enable Hysteresis of BOD detection.
[3:0]	**BODVL**	**BOD Voltage Level** 1111b = 4.6V. 1110b = 4.2V. 1101b = 3.9V. 1100b = 3.7V. 1011b = 3.6V. 1010b = 3.4V. 1001b = 3.1V. 1000b = 3.0V. 0111b= 2.8V 0110b = 2.6V. 0101b = 2.4V. 0100b = 2.2V. 0011b = 2.1V. 0010b = 2.0V. 0001b = 1.9V. 0000b = 1.8V.

Brown-Out Detector Enable Register (BOD_BODCTL)

The effect of this is to generate a NMI interrupt (default NMI interrupt is BOD interrupt). The NMI ISR can be defined by the user to respond to this low voltage level.

Register	Offset	R/W	Description	Reset Value
BOD_BODCTL	BOD_BA+0x04	R/W	Brown Out Detector Enable Register	0x000X_000X

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

Bits	Description	
[31:5]	**Reserved**	Reserved.
[18:17]	**LVR_FILTER**	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]	**LVR_EN**	**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 config0[27]. 0 = Disable LVR function. 1 = Enable LVR function.
[15:6]	**Reserved**	Reserved.
[5]	**BODRST**	**BOD Reset** 1: Reset device when BOD is triggered.
[4]	**BODOUT**	**Output of BOD Detection Block** This signal can be monitored to determine the current state of the BOD comparator. BODOUT = 1 implies that VCC is less than BODVL.
[3]	**BODIF**	**Current Status of Interrupt** Latched whenever a BOD event occurs and IE=1. Write ‘1’ to clear.
[2]	**BODINTEN**	**BOD Interrupt Enable** 0= Disable BOD Interrupt. 1= Enable BOD Interrupt.

[1:0]	**BODEN**	**BOD Enable** 1xb = Enable continuous BOD detection. 01b = Enable time multiplexed BOD detection. See BOD_BODDTMR register. 00b = Disable BOD Detection.

Detection Time Multiplex Register (BOD_BODDTMR)

The BOD detector can be set up to take periodic samples of the supply voltage to minimize power consumption. The circuit can be configured and used in Standby Power Down (SPD) mode and can wake up the device if a BOD is event detected. The detection timer uses the OSC10k oscillator as time base so this oscillator must be active for timer operation. When active the BOD circuit requires ~165uA. With default timer settings, average current reduces to 500nA 165uA*DURTON/(DURTON+DURTOFF).

Register	Offset	R/W	Description	Reset Value
BOD_BODDTMR	BOD_BA+0x10	R/W	Brown Out Detector Timer Register	0x0003_03E3

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="4">Reserved</td><td colspan="4">DURTON</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="8">DURTOFF</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">DURTOFF</td></tr>
</table>

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**DURTON**	**Time BOD Detector Is Active** (DURTON+1) * 100us. Minimum value is 1. (default is 400us)
[15:0]	**DURTOFF**	**Time BOD Detector Is Off** (DURTOFF+1)*100us . Minimum value is 7. (default is 99.6ms)

5.6 I2C Serial Interface Controller (Master/Slave)

5.6.1 Introduction

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.

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).

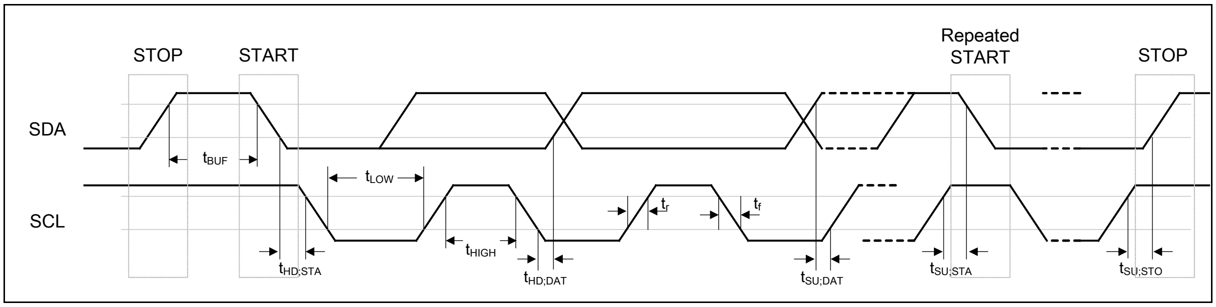


Figure 5-9 I2C Bus Timing

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 Table 5-3 for alternate GPIO pin functions. Pull up resistor is needed for these pins for I2C operation as these are open drain pins.

The I2C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of the bus are:

- 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 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 clocks allow versatile rate control
- Supports 7-bit addressing mode

- I2C-bus controllers support multiple address recognition (Four slave address with mask option)

5.6.1.1 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

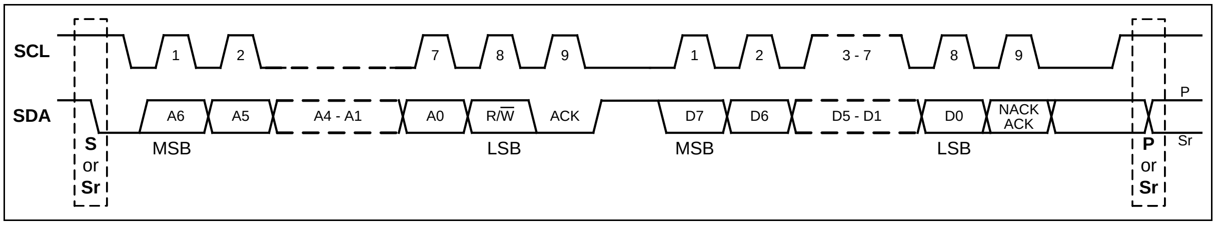


Figure 5-10 I2C Protocol

5.6.1.2 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.

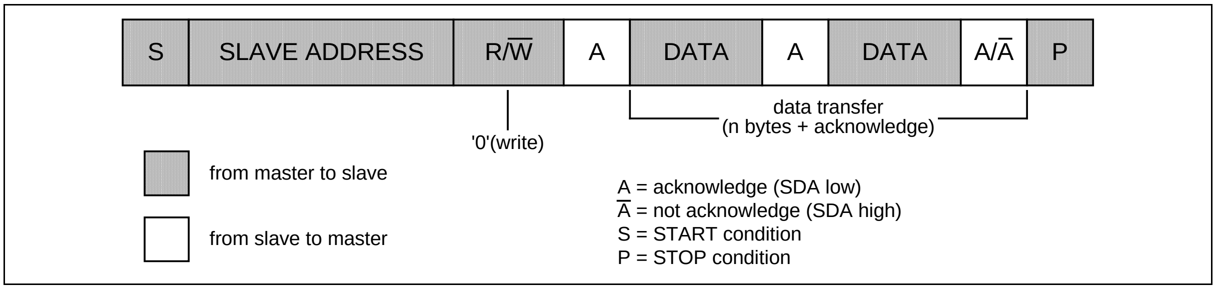


Figure 5-11 Master Transmits Data to Slave

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.

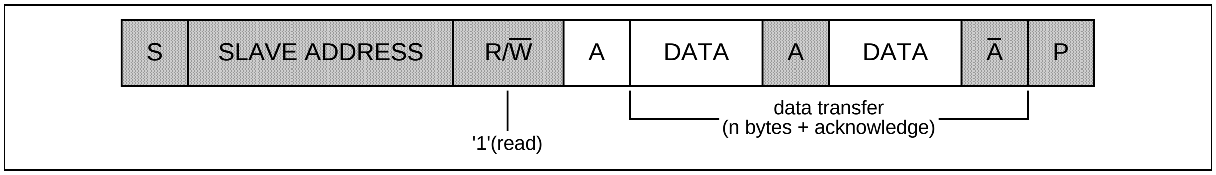


Figure 5-12 Master Reads Data from Slave

5.6.1.3 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.

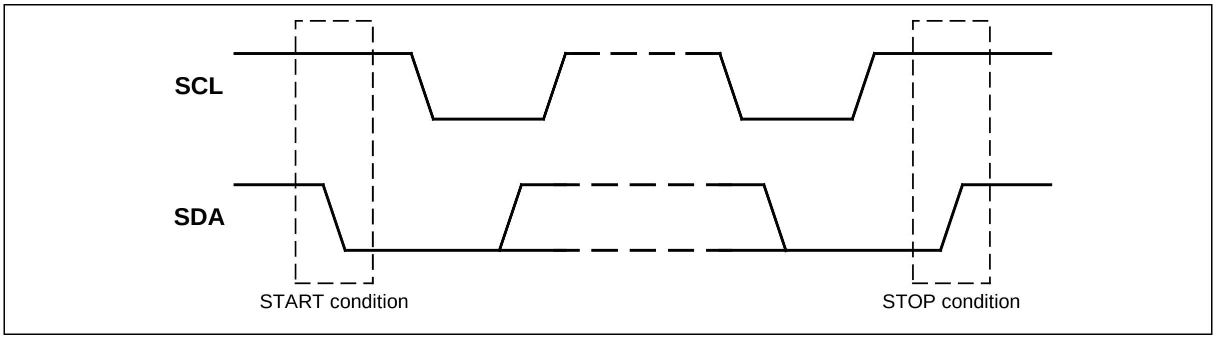


Figure 5-13 START and STOP condition

5.6.1.4 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.

5.6.1.5 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.

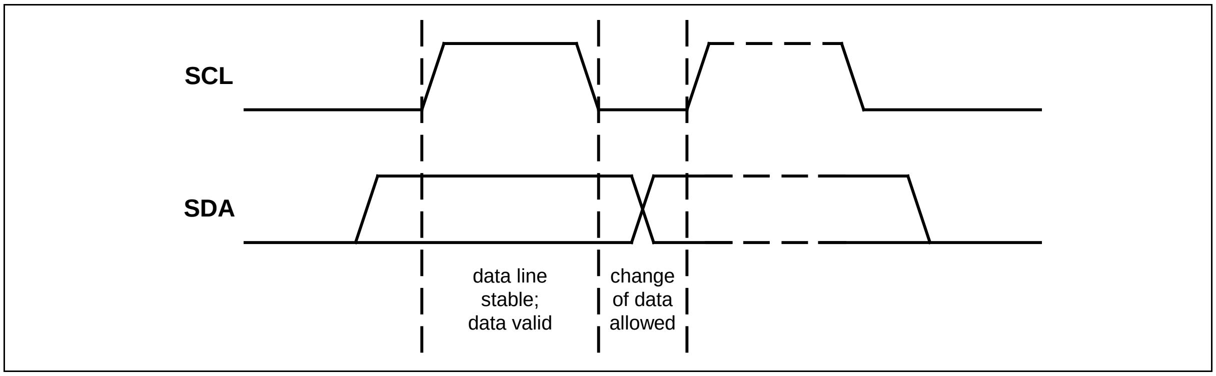


Figure 5-14 Bit Transfer on the I2C bus

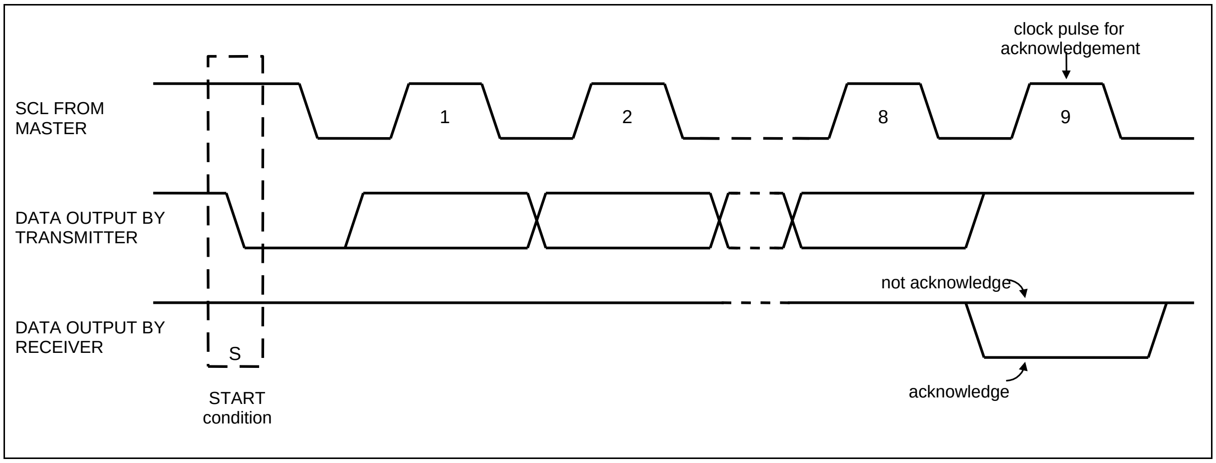


Figure 5-15 Acknowledge on the I2C bus

5.6.2 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.

5.6.2.1 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.

5.6.2.2 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.

5.6.2.3 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.

5.6.2.4 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.

5.6.3 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:

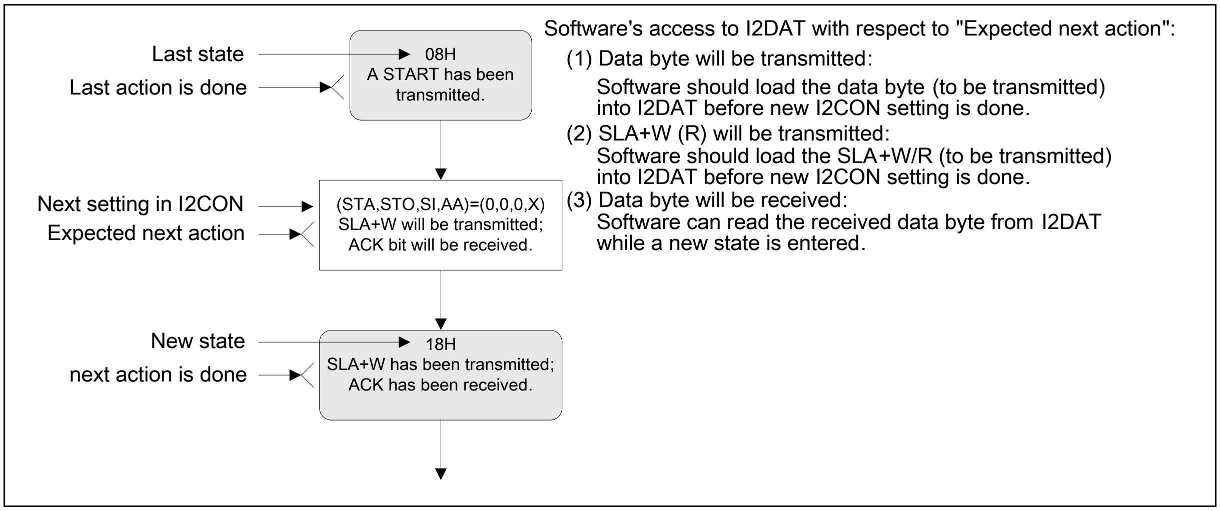

Last state
Last action is done
08H
A START has been transmitted.
Next setting in I2CON
Expected next action
(STA,STO,SI,AA)=(0,0,0,X)
SLA+W will be transmitted;
ACK bit will be received.
New state
next action is done
18H
SLA+W has been transmitted;
ACK has been received.
Software's access to I2DAT with respect to "Expected next action":
(1) Data byte will be transmitted:
Software should load the data byte (to be transmitted) into I2DAT before new I2CON setting is done.
(2) SLA+W (R) will be transmitted:
Software should load the SLA+W/R (to be transmitted) into I2DAT before new I2CON setting is done.
(3) Data byte will be received:
Software can read the received data byte from I2DAT while a new state is entered.

Figure 5-16 Legend for the following four figures

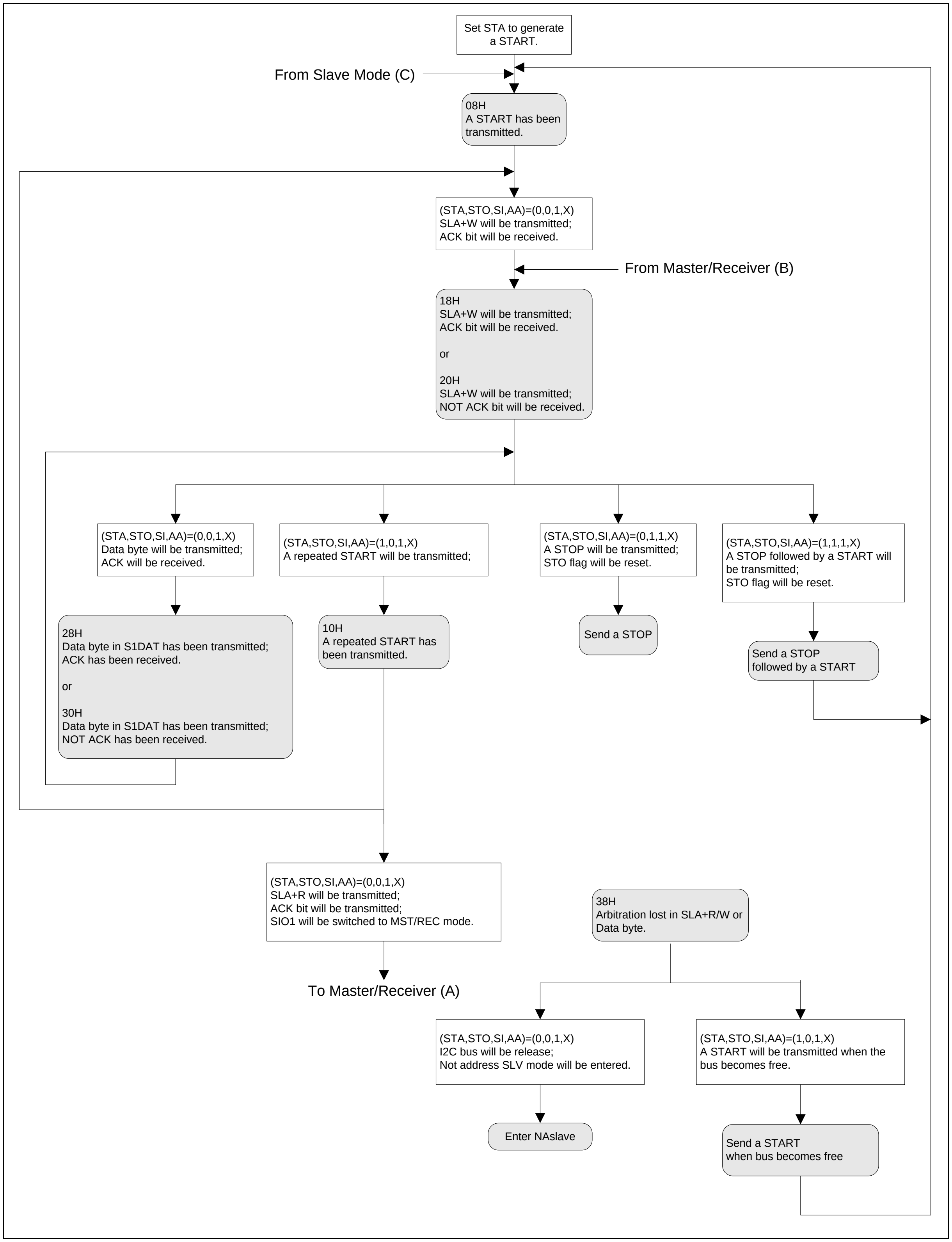

Set STA to generate a START.
From Slave Mode (C)
08H
A START has been transmitted.
(STA,STO,SI,AA)=(0,0,1,X)
SLA+W will be transmitted;
ACK bit will be received.
From Master/Receiver (B)
18H
SLA+W will be transmitted;
ACK bit will be received.
or
20H
SLA+W will be transmitted;
NOT ACK bit will be received.
(STA,STO,SI,AA)=(0,0,1,X)
Data byte will be transmitted;
ACK will be received.
(STA,STO,SI,AA)=(1,0,1,X)
A repeated START will be transmitted;
(STA,STO,SI,AA)=(0,1,1,X)
A STOP will be transmitted;
STO flag will be reset.
(STA,STO,SI,AA)=(1,1,1,X)
A STOP followed by a START will be transmitted;
STO flag will be reset.
28H
Data byte in S1DAT has been transmitted;
ACK has been received.
or
30H
Data byte in S1DAT has been transmitted;
NOT ACK has been received.
10H
A repeated START has been transmitted.
Send a STOP
Send a STOP followed by a START
(STA,STO,SI,AA)=(0,0,1,X)
SLA+R will be transmitted;
ACK bit will be transmitted;
SIO1 will be switched to MST/REC mode.
To Master/Receiver (A)
38H
Arbitration lost in SLA+R/W or Data byte.
(STA,STO,SI,AA)=(0,0,1,X)
I2C bus will be release;
Not address SLV mode will be entered.
(STA,STO,SI,AA)=(1,0,1,X)
A START will be transmitted when the bus becomes free.
Enter NAslave
Send a START when bus becomes free

Figure 5-17 Master Transmitter Mode

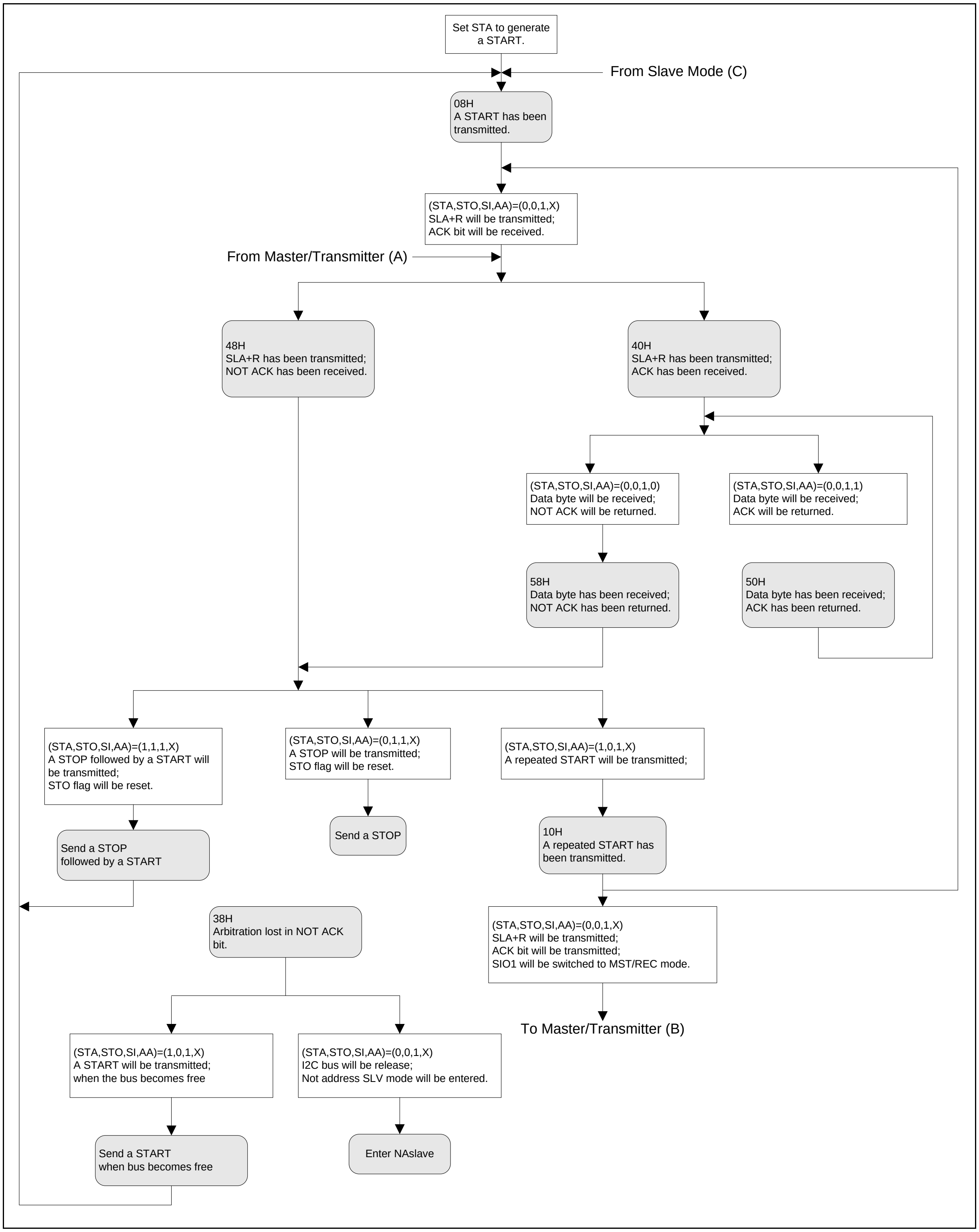

Set STA to generate a START.
From Slave Mode (C)
08H A START has been transmitted.
(STA,STO,SI,AA)=(0,0,1,X) SLA+R will be transmitted; ACK bit will be received.
From Master/Transmitter (A)
48H SLA+R has been transmitted; NOT ACK has been received.
40H SLA+R has been transmitted; ACK has been received.
(STA,STO,SI,AA)=(0,0,1,0) Data byte will be received; NOT ACK will be returned.
(STA,STO,SI,AA)=(0,0,1,1) Data byte will be received; ACK will be returned.
58H Data byte has been received; NOT ACK has been returned.
50H Data byte has been received; ACK has been returned.
(STA,STO,SI,AA)=(1,1,1,X) A STOP followed by a START will be transmitted; STO flag will be reset.
(STA,STO,SI,AA)=(0,1,1,X) A STOP will be transmitted; STO flag will be reset.
(STA,STO,SI,AA)=(1,0,1,X) A repeated START will be transmitted;
Send a STOP followed by a START
Send a STOP
10H A repeated START has been transmitted.
38H Arbitration lost in NOT ACK bit.
(STA,STO,SI,AA)=(0,0,1,X) SLA+R will be transmitted; ACK bit will be transmitted; SIO1 will be switched to MST/REC mode.
To Master/Transmitter (B)
(STA,STO,SI,AA)=(1,0,1,X) A START will be transmitted; when the bus becomes free
(STA,STO,SI,AA)=(0,0,1,X) I2C bus will be release; Not address SLV mode will be entered.
Send a START when bus becomes free
Enter NAslave

Figure 5-18 Master Receiver Mode

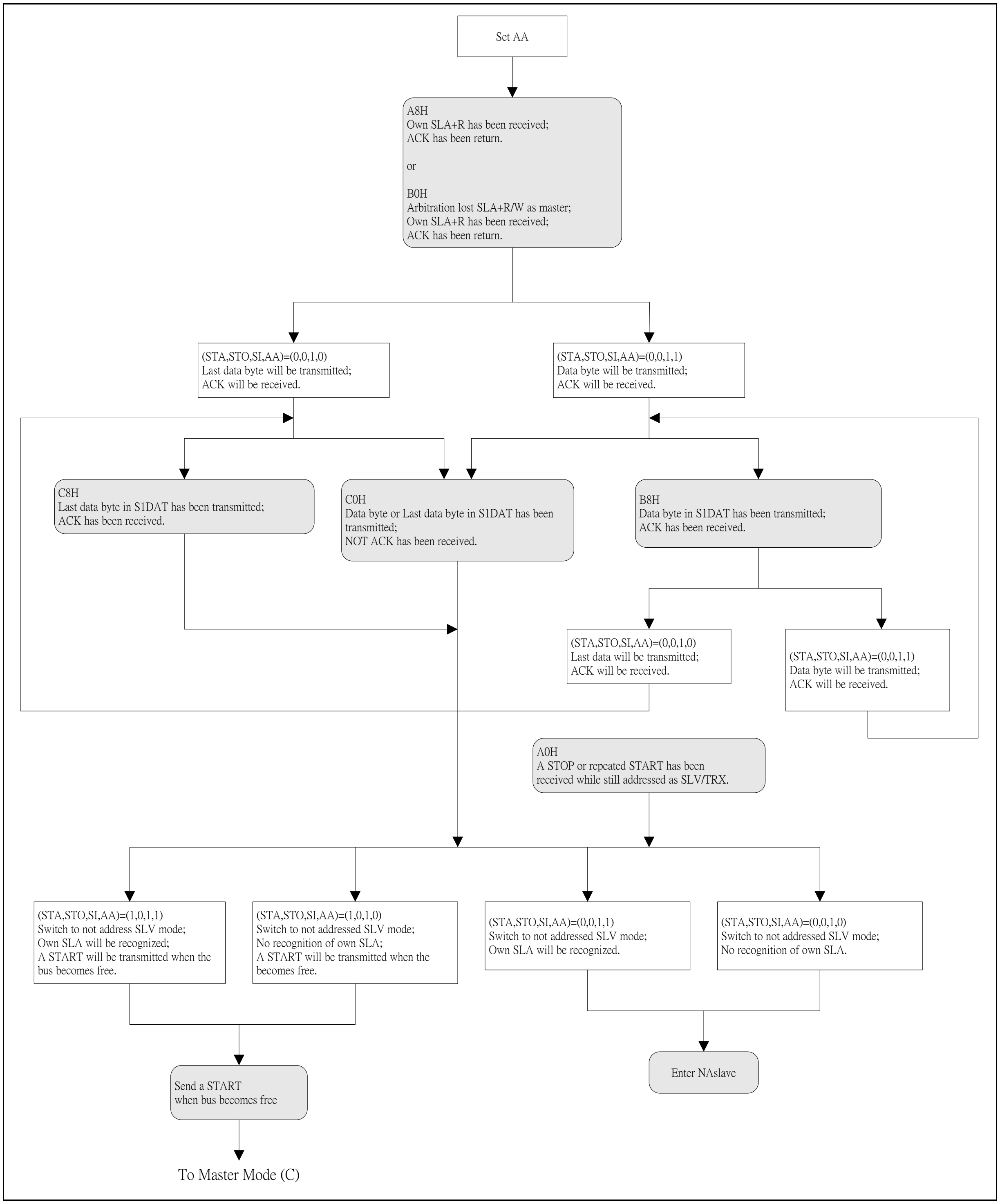

Set AA
A8H
Own SLA+R has been received;
ACK has been return.
or
B0H
Arbitration lost SLA+R/W as master;
Own SLA+R has been received;
ACK has been return.
(STA,STO,SI,AA)=(0,0,1,0)
Last data byte will be transmitted;
ACK will be received.
(STA,STO,SI,AA)=(0,0,1,1)
Data byte will be transmitted;
ACK will be received.
C8H
Last data byte in S1DAT has been transmitted;
ACK has been received.
C0H
Data byte or Last data byte in S1DAT has been transmitted;
NOT ACK has been received.
B8H
Data byte in S1DAT has been transmitted;
ACK has been received.
(STA,STO,SI,AA)=(0,0,1,0)
Last data will be transmitted;
ACK will be received.
(STA,STO,SI,AA)=(0,0,1,1)
Data byte will be transmitted;
ACK will be received.
A0H
A STOP or repeated START has been received while still addressed as SLV/TRX.
(STA,STO,SI,AA)=(1,0,1,1)
Switch to not address SLV mode;
Own SLA will be recognized;
A START will be transmitted when the bus becomes free.
(STA,STO,SI,AA)=(1,0,1,0)
Switch to not addressed SLV mode;
No recognition of own SLA;
A START will be transmitted when the becomes free.
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed SLV mode;
Own SLA will be recognized.
(STA,STO,SI,AA)=(0,0,1,0)
Switch to not addressed SLV mode;
No recognition of own SLA.
Send a START
when bus becomes free
Enter NAslave
To Master Mode (C)

Figure 5-19 Slave Transmitter Mode

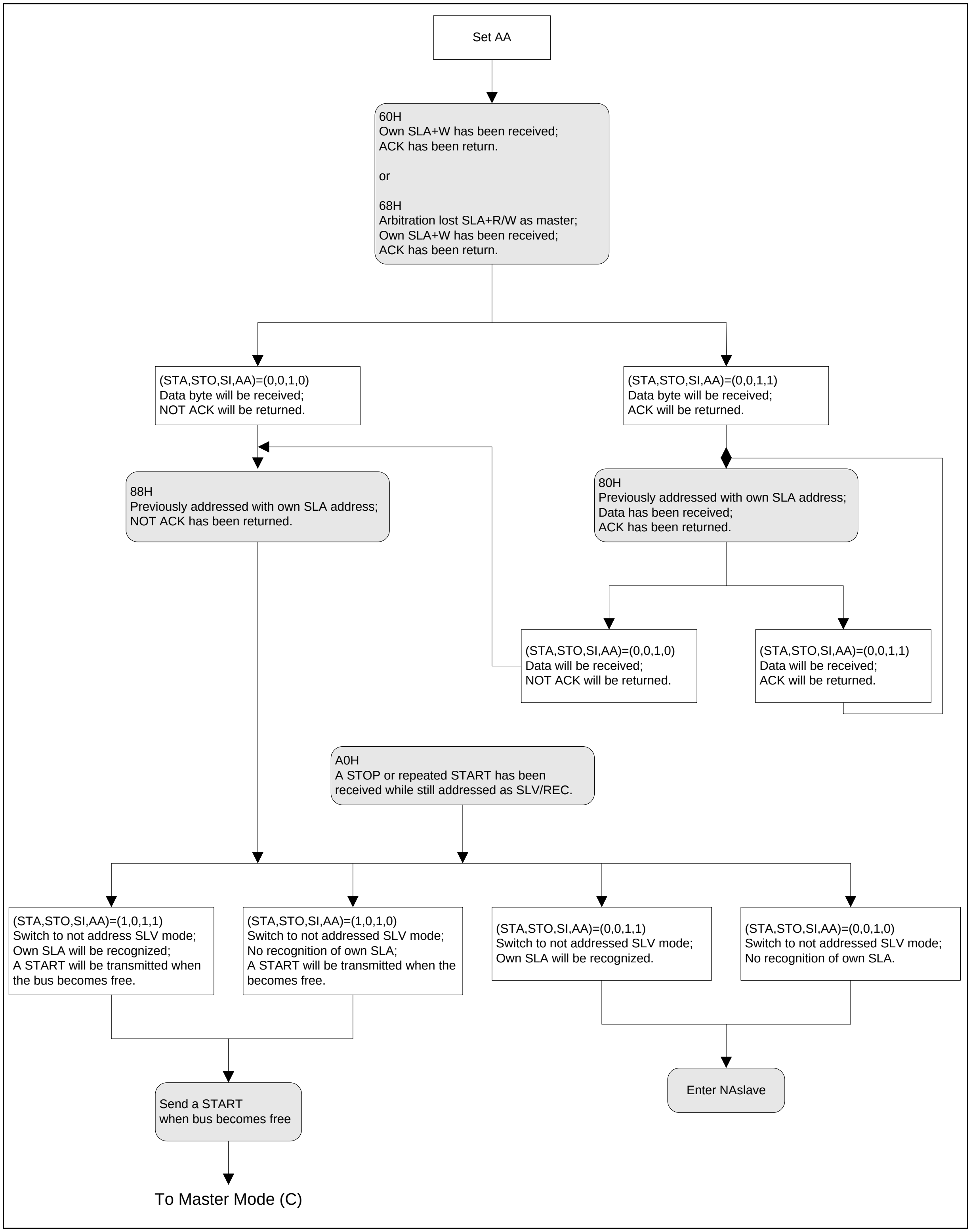

Set AA
60H
Own SLA+W has been received;
ACK has been return.
or
68H
Arbitration lost SLA+R/W as master;
Own SLA+W has been received;
ACK has been return.
(STA,STO,SI,AA)=(0,0,1,0)
Data byte will be received;
NOT ACK will be returned.
(STA,STO,SI,AA)=(0,0,1,1)
Data byte will be received;
ACK will be returned.
88H
Previously addressed with own SLA address;
NOT ACK has been returned.
80H
Previously addressed with own SLA address;
Data has been received;
ACK has been returned.
(STA,STO,SI,AA)=(0,0,1,0)
Data will be received;
NOT ACK will be returned.
(STA,STO,SI,AA)=(0,0,1,1)
Data will be received;
ACK will be returned.
A0H
A STOP or repeated START has been
received while still addressed as SLV/REC.
(STA,STO,SI,AA)=(1,0,1,1)
Switch to not address SLV mode;
Own SLA will be recognized;
A START will be transmitted when
the bus becomes free.
(STA,STO,SI,AA)=(1,0,1,0)
Switch to not addressed SLV mode;
No recognition of own SLA;
A START will be transmitted when the
becomes free.
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed SLV mode;
Own SLA will be recognized.
(STA,STO,SI,AA)=(0,0,1,0)
Switch to not addressed SLV mode;
No recognition of own SLA.
Send a START
when bus becomes free
Enter NAslave
To Master Mode (C)

Figure 5-20 Slave Receiver Mode

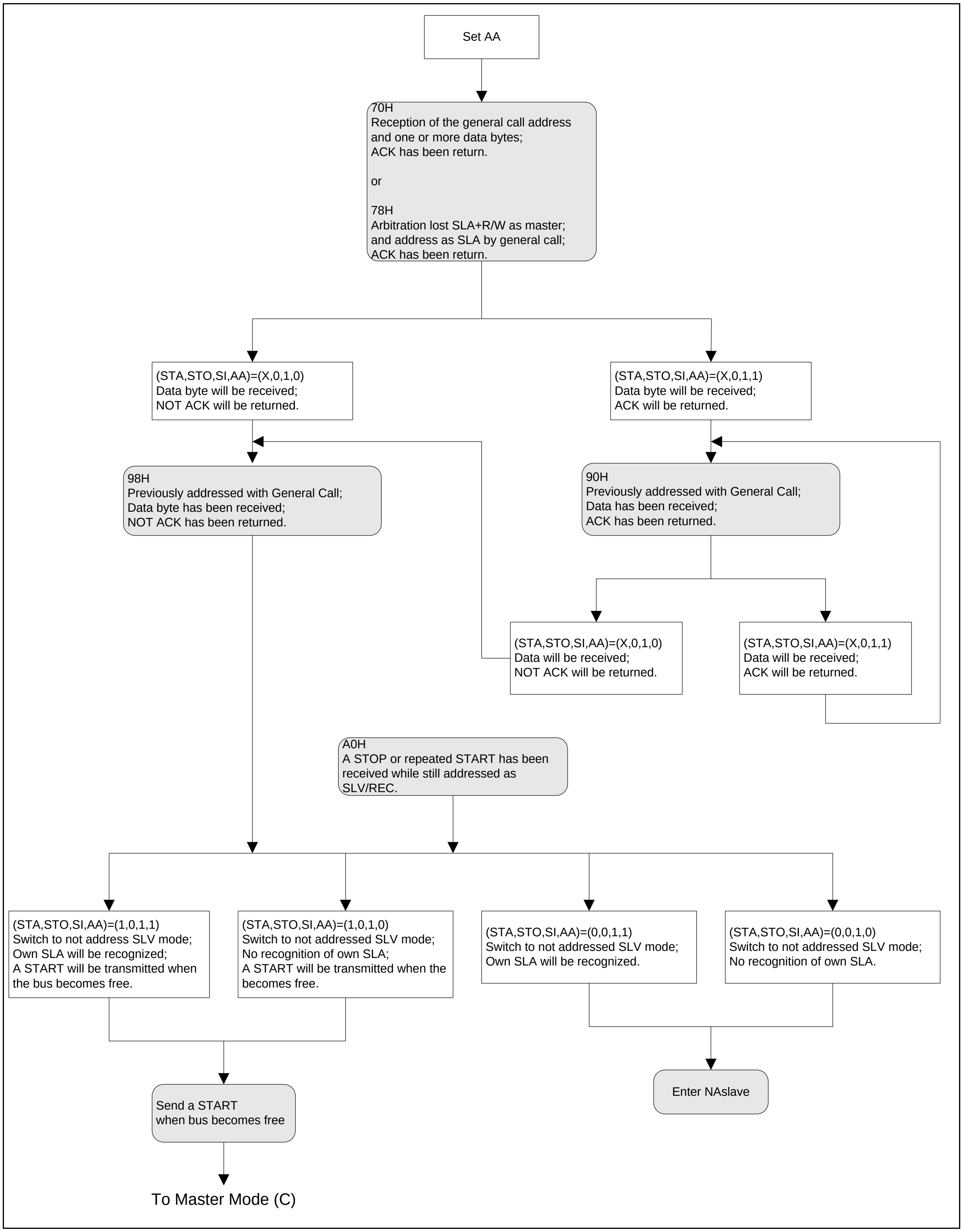

Set AA
70H
Reception of the general call address and one or more data bytes;
ACK has been return.
or
78H
Arbitration lost SLA+R/W as master;
and address as SLA by general call;
ACK has been return.
(STA,STO,SI,AA)=(X,0,1,0)
Data byte will be received;
NOT ACK will be returned.
(STA,STO,SI,AA)=(X,0,1,1)
Data byte will be received;
ACK will be returned.
98H
Previously addressed with General Call;
Data byte has been received;
NOT ACK has been returned.
90H
Previously addressed with General Call;
Data has been received;
ACK has been returned.
(STA,STO,SI,AA)=(X,0,1,0)
Data will be received;
NOT ACK will be returned.
(STA,STO,SI,AA)=(X,0,1,1)
Data will be received;
ACK will be returned.
A0H
A STOP or repeated START has been received while still addressed as SLV/REC.
(STA,STO,SI,AA)=(1,0,1,1)
Switch to not address SLV mode;
Own SLA will be recognized;
A START will be transmitted when the bus becomes free.
(STA,STO,SI,AA)=(1,0,1,0)
Switch to not addressed SLV mode;
No recognition of own SLA;
A START will be transmitted when the becomes free.
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed SLV mode;
Own SLA will be recognized.
(STA,STO,SI,AA)=(0,0,1,0)
Switch to not addressed SLV mode;
No recognition of own SLA.
Send a START when bus becomes free
Enter NAslave
To Master Mode (C)

Figure 5-21 GC Mode

5.6.4 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 INTEN (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 theISD91200 is 0x4002_0000.

5.6.4.1 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.

5.6.4.2 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.

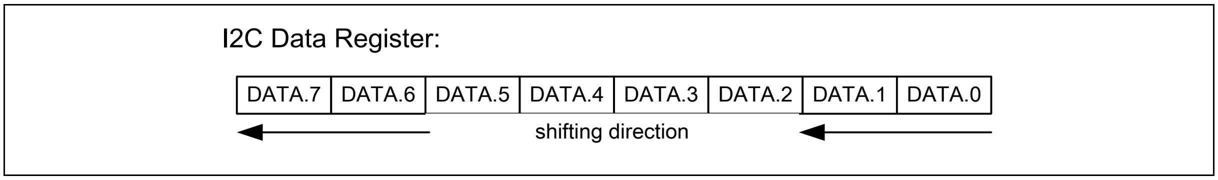


Figure 5-22 I2C Data Shift Direction

5.6.4.3 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.

5.6.4.4 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.

5.6.4.5 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.

5.6.4.6 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 5-23 I2C Time-out Count Block Diagram for the 14-bit time-out counter. User can clear TOIF by writing one to this bit.

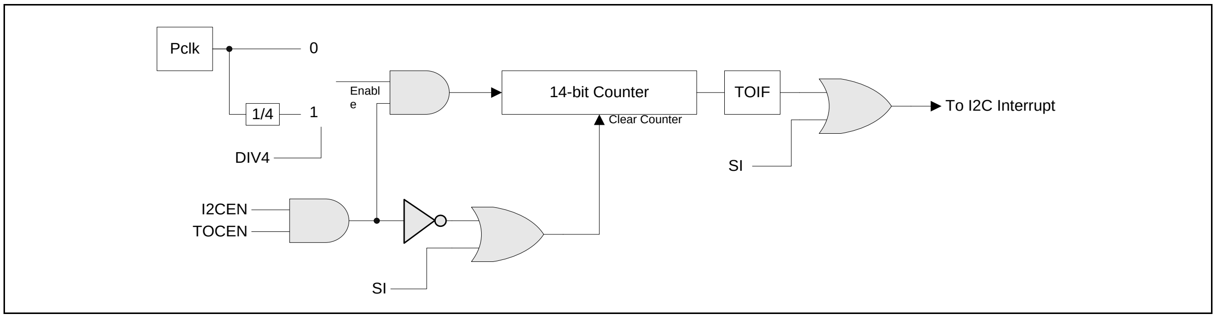


Figure 5-23 I2C Time-out Count Block Diagram

5.6.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
I2C Base Address: **I2C_BA = 0x4002_0000**				
I2C_CTL	I2C_BA+0x00	R/W	I2C Control Register	0x0000_0000
I2C_ADDR0	I2C_BA+0x04	R/W	I2C Slave address Register0	0x0000_0000
I2C_DAT	I2C_BA+0x08	R/W	I2C DATA Register	0x0000_0000
I2C_STATUS	I2C_BA+0x0C	R	I2C Status Register	0x0000_00F8
I2C_CLKDIV	I2C_BA+0x10	R/W	I2C clock divided Register	0x0000_0000
I2C_TOCTL	I2C_BA+0x14	R/W	I2C Time out control Register	0x0000_0000
I2C_ADDR1	I2C_BA+0x18	R/W	I2C Slave address Register1	0x0000_0000
I2C_ADDR2	I2C_BA+0x1C	R/W	I2C Slave address Register2	0x0000_0000
I2C_ADDR3	I2C_BA+0x20	R/W	I2C Slave address Register3	0x0000_0000
I2C_ADDRMSK0	I2C_BA+0x24	R/W	I2C Slave address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2C_BA+0x28	R/W	I2C Slave address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2C_BA+0x2C	R/W	I2C Slave address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2C_BA+0x30	R/W	I2C Slave address Mask Register3	0x0000_0000

5.6.6 Register Description

I2C CONTROL REGISTER (I2C_CTL)

Register	Offset	R/W	Description	Reset Value
I2C_CTL	I2C_BA+0x00	R/W	I2C 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**
INTEN	**I2CEN**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[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 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.

[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.

I2C DATA REGISTER (I2C_DAT)

Register	Offset	R/W	Description	Reset Value
I2C_DAT	I2C_BA+0x08	R/W	I2C 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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAT							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[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	I2C_BA+0x0C	R	I2C Status Register	0x0000_00F8

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**
STATUS							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[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	I2C_BA+0x10	R/W	I2C clock divided 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**
DIVIDER							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DIVIDER**	**I2C Clock Divided Register** The I2C clock rate bits: Data Baud Rate of I2C = PCLK /(4x(I2C_CLKDIV+1)).

I2C TIME-OUT COUNTER REGISTER (I2C_TOCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TOCTL	I2C_BA+0x14	R/W	I2C Time out 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					**TOCEN**	**TOCDIV4**	**TOIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[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 (I2CADDRx)

Register	Offset	R/W	Description	Reset Value
I2C_ADDR0	I2C_BA+0x04	R/W	I2C Slave address Register0	0x0000_0000
I2C_ADDR1	I2C_BA+0x18	R/W	I2C Slave address Register1	0x0000_0000
I2C_ADDR2	I2C_BA+0x1C	R/W	I2C Slave address Register2	0x0000_0000
I2C_ADDR3	I2C_BA+0x20	R/W	I2C Slave address Register3	0x0000_0000

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="7">ADDR[7:1]</td><td>GC</td></tr>
</table>

Bits	Description	
[31:8]	**Reserved**	Reserved.
[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 (I2CADMx)

Register	Offset	R/W	Description	Reset Value
I2C_ADDRMSK0	I2C_BA+0x24	R/W	I2C Slave address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2C_BA+0x28	R/W	I2C Slave address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2C_BA+0x2C	R/W	I2C Slave address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2C_BA+0x30	R/W	I2C Slave address Mask Register3	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**
ADDRMSK							**Reserved**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[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.

5.7 PWM Generator and Capture Timer

5.7.1 Introduction

The ISD91200 has two PWM generators which can be configured as 4 independent PWM outputs, PWM0CH0, PWM0CH1, PWM0CH2 and PWM0CH3, or as a complementary PWM pairs with programmable dead-zone generator. Each PWM Generator has an 8-bit pre-scaler, a clock divider providing 5 divided frequencies (1, 1/2, 1/4, 1/8, 1/16), two PWM Timers including two clock selectors, two 16-bit PWM down-counters for PWM period control, two 16-bit comparators for PWM duty control and one dead-zone generator. The PWM Generator provides PWM interrupt flags which are set by hardware when the corresponding PWM period down counter reaches zero. Each PWM interrupt source, with its corresponding enable bit, can generate a PWM interrupt request to the CPU. The PWM generator can be configured in one-shot mode to produce only one PWM cycle signal or continuous mode to output a periodic PWM waveform.

When PWM_CTL.DTEN01 is set, PWM0CH0 and PWM0CH1 perform complementary paired PWM function; the paired PWM timing, period, duty and dead-time are determined by PWM0 timer and Dead-zone generator0. Refer to Figure 5-25 PWM Generator Architecture Diagram for the architecture of PWM Timers.

To prevent PWM driving glitches to an output pin, the 16-bit period down-counter and 16-bit comparator are implemented with a double buffer. When user writes data to the counter/comparator registers, the updated value will not be load into the 16-bit down-counter/comparator until the down-counter reaches zero.

When the 16-bit period down-counter reaches zero, the interrupt request is generated. If PWM timer is configured in continuous mode, when the down counter reaches zero, it is reloaded with PWM Counter Register automatically and begins decrementing again. If the PWM timer is configured in one-shot mode, the down counter will stop and generate a single interrupt request when it reaches zero.

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

The alternate function of the PWM-timer is as a digital input capture timer. If Capture function is enabled the PWM output pin is switched as a capture input pin. The Capture0 and PWM0CH0 share one timer which is included in PWM0CH0; and the Capture1 and PWM0CH1 share PWM0CH1 timer. User must setup the PWM-timer before enabling the Capture feature. After the capture feature is enabled, the count is latched to the Capture Rising Latch Register (RCAPDAT) when input channel has a rising transition and latched to Capture Falling Latch Register (PWM_FCAPDATx) when input channel has a falling transition. Capture channel 0 interrupt is programmable by setting PWM_CAPCTL01.CRLIEN0[1] (Rising latch Interrupt enable) and PWM_CAPCTL01.CFLIEN0[2]] (Falling latch Interrupt enable) to determine the condition of interrupt occurrence. Capture channel 1 has the same feature by setting PWM_CAPCTL01.CRLIEN1[17] and PWM_CAPCTL01.CFLIEN1[18]. Whenever Capture issues interrupt, the PWM counter will also be reloaded.

5.7.2 Features

5.7.2.1 PWM function features:

- PWM Generator, incorporating an 8-bit pre-scaler, clock divider, two PWM-timers (down counters), a dead-zone generator and two PWM outputs.
- Up to 4 PWM channels or two paired PWM channel.
- 16 bits resolution.
- PWM Interrupt request synchronous with PWM period.
- Single-shot or Continuous mode PWM.

- Dead-Zone generator.

5.7.2.2 Capture Function Features:

- Timing control logic shared with PWM Generators.
- 2 Capture input channels shared with 2 PWM output channels.
- Each channel supports a rising latch register (RCAPDAT), a falling latch register (PWM_FCAPDATx) and Capture interrupt flag (CAPIFx)

5.7.3 PWM Generator Architecture

The following figures illustrate the architecture of the PWM.

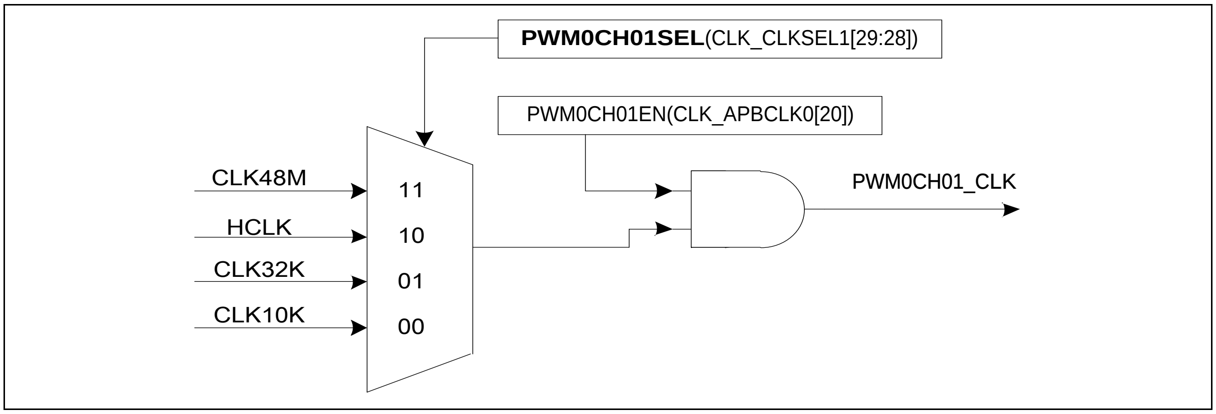


Figure 5-24 PWM Generator Clock Source Control

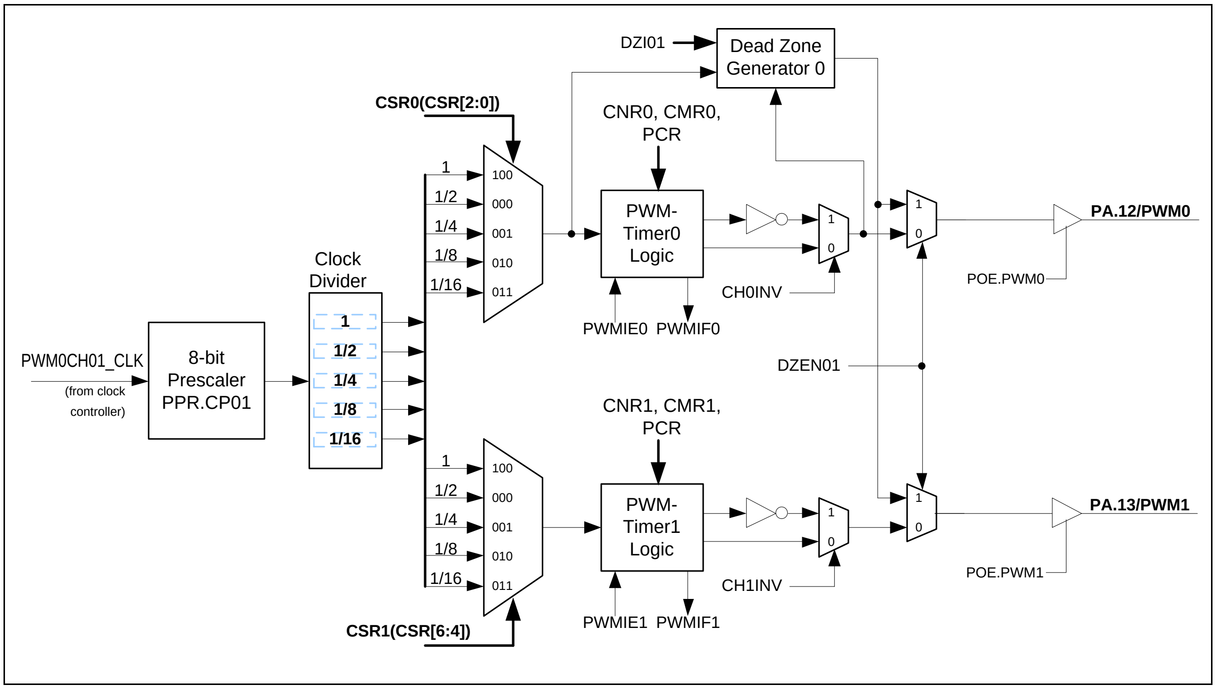


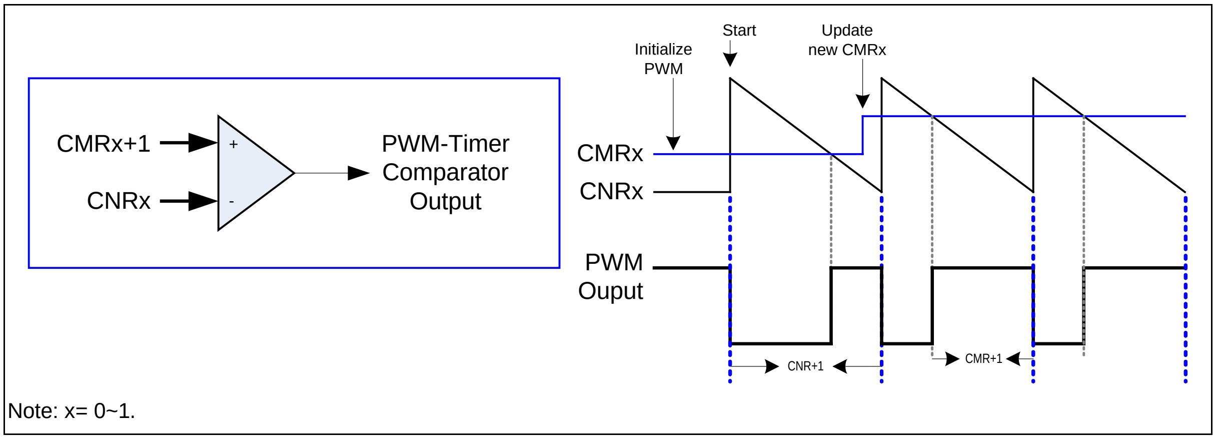
Figure 5-25 PWM Generator Architecture Diagram

5.7.4 PWM-Timer Operation

The PWM period and duty control are configured by the PWM down-counter register (PWMx_PERIODx) and PWM comparator register (PWMx_CMPDATx). Formulas for calculating the pulse width modulation are shown below and demonstrated in Figure 5-26 PWM Generation Timing. Note that the corresponding GPIO pins must be configured as the alternate function before PWM function is enabled.

- PWM frequency = PWM0CH01_CLK/(prescale+1)*(clock divider)/(PERIOD+1);
- Duty cycle = (CMP+1)/(PERIOD+1).
- CMP >= PERIOD: PWM output is always high.
- CMP < PERIOD: PWM low width= (PERIOD-CMP) unit[1]; PWM high width=(CMP+1) unit.
- CMP = 0: PWM low width = (PERIOD) unit; PWM high width=1 unit

Note: 1. Unit = one PWM clock cycle.



Note: x= 0~1.

Figure 5-26 PWM Generation Timing

The procedure to operate the PWM generator is shown in Figure 5-27 PWM-Timer Operation Timing. First initialize the PWM settings. At the same time ensure that GPIO are configured to PWM function. Next step is to enable PWM channel. After this, if PERIOD or CMP register is written by software, it is double buffered until the next counter reload, at which time the registers are updated to new values.

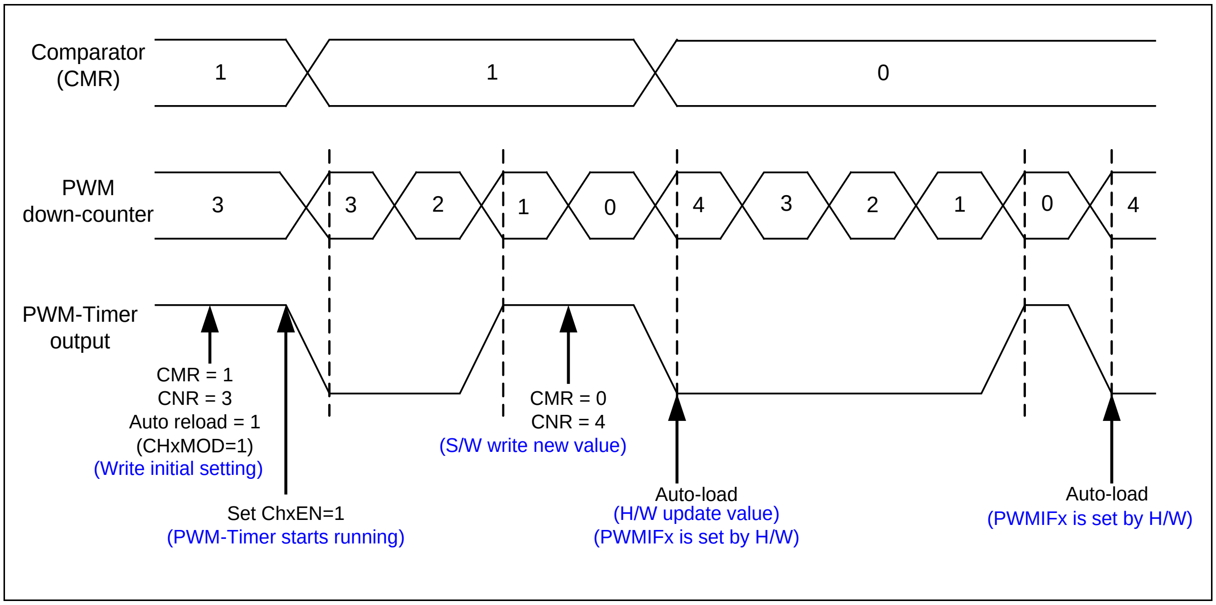


Figure 5-27 PWM-Timer Operation Timing

5.7.5 PWM Double Buffering, Auto-reload and One-shot Operation

The ISD91200 series PWM Timers are double buffered, the reload value is updated at the start of next period without affecting current timer operation. The PWM counter reset value can be written into PWM_PERIOD0~1 and current PWM counter value can be read from PWM_CNT0~1.

The bit CNTMODEx in PWM Control Register (PWM_CTL) determines whether PWMx operates in auto-reload or one-shot mode. If CNTMODEx is set to one, the auto-reload operation loads PERIODx to PWM counter when PWM counter reaches zero. If PERIODx is set to zero, PWM counter will halt when PWM counter counts to zero. If CNTMODEx is set as zero, counter will stop immediately.

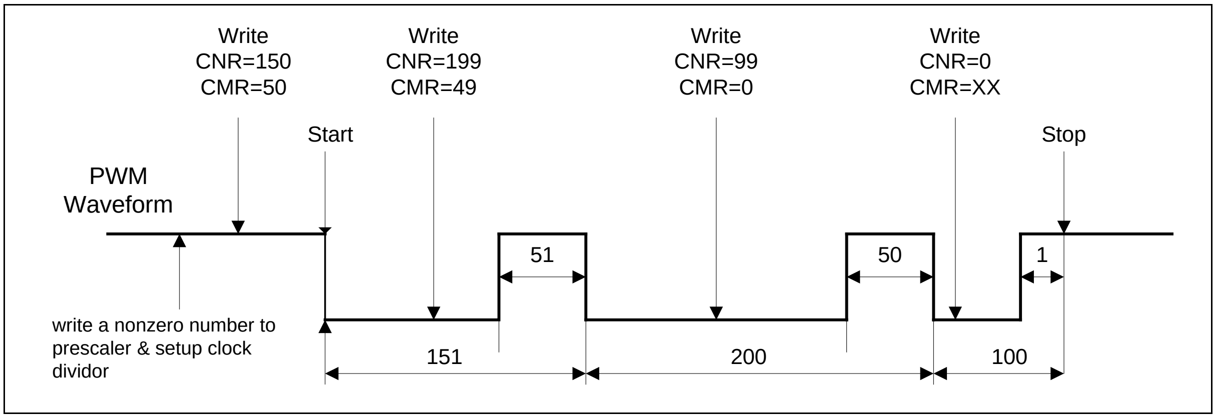


Figure 5-28 PWM Double Buffering.

5.7.6 Modulate Duty Cycle

The double buffering allows CMP to be written at any point in current cycle. The loaded value will take effect from next cycle. This is demonstrated in Figure 5-29 PWM Controller Duty Cycle Modulation (PERIOD = 150).

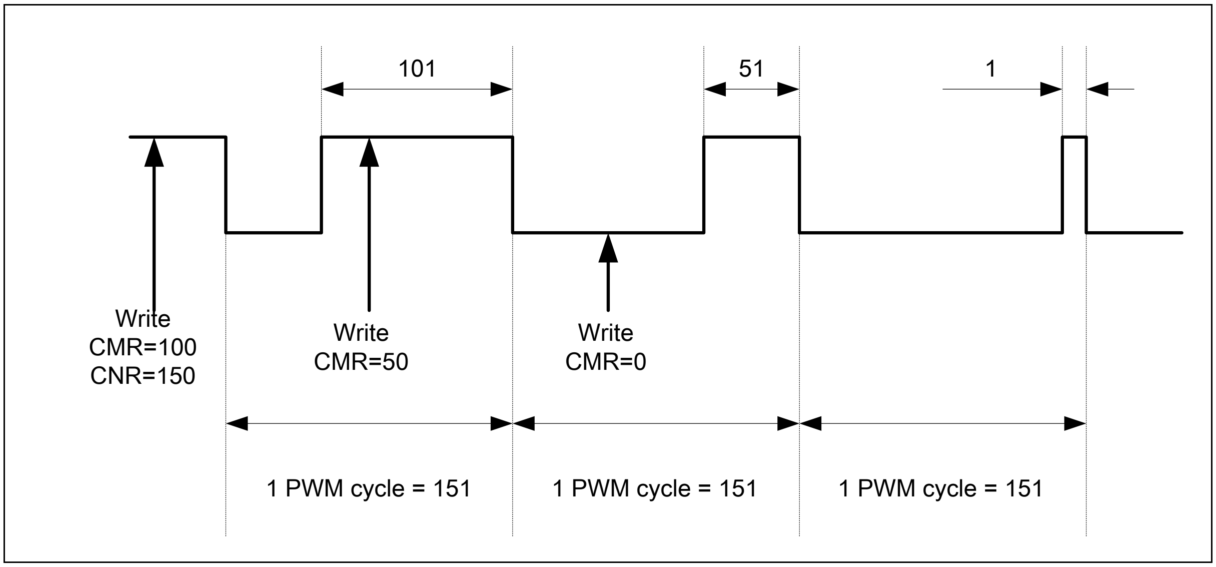


Figure 5-29 PWM Controller Duty Cycle Modulation (PERIOD = 150).

5.7.7 Dead-Zone Generator

The ISD91200 PWM generator includes a Dead Zone generator. This is used to ensure neither PWM output is active simultaneously for power device protection. The function generates a programmable time gap between rising PWM outputs. The user can program PPRx.DZI to determine the Dead Zone interval. The Dead Zone generator behavior is demonstrated in Figure 5-30 Paired-PWM Output with Dead Zone Generation Operation.

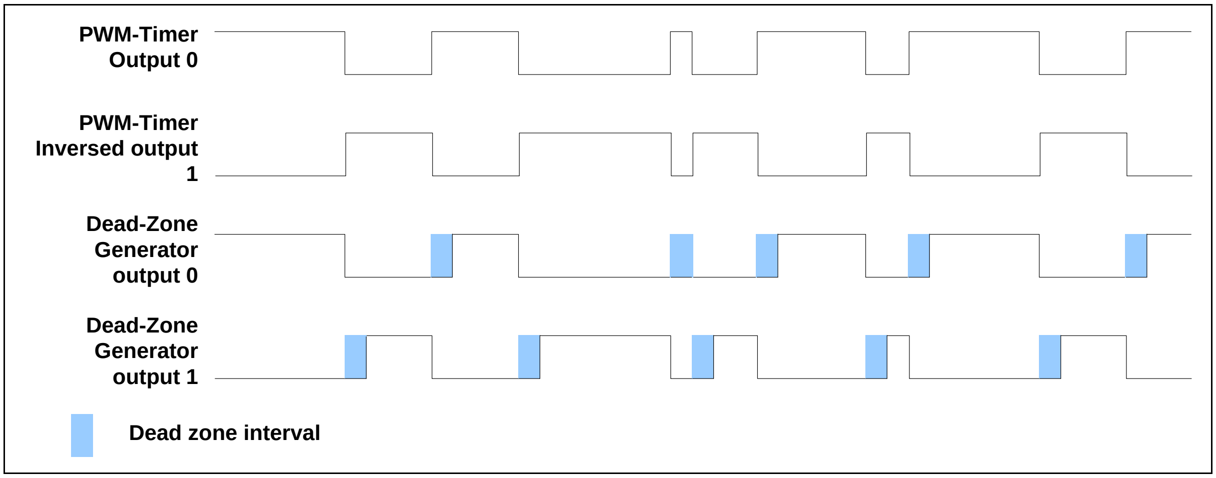

PWM-Timer Output 0
PWM-Timer Inversed output 1
Dead-Zone Generator output 0
Dead-Zone Generator output 1
Dead zone interval

Figure 5-30 Paired-PWM Output with Dead Zone Generation Operation

5.7.8 Capture Timer Operation

Instead of using the PWM generator to output a modulated signal, it can be configured as a capture timer to measure a modulated input. Capture channel 0 and PWM0CH0 share one timer and Capture channel 1 and PWM0CH1 share another timer. The capture timer latches PWM-counter to RCAPDAT when input channel has a rising transition and latches PWM-counter to PWM_FCAPDATx when input channel has a falling transition. Capture channel 0 interrupt is programmable by setting PWM_CAPCTL01[1] (Rising latch Interrupt enable) and PWM_CAPCTL01[2] (Falling latch Interrupt enable) to decide the condition of interrupt occurrence. Capture channel 1 has the same feature by setting PWM_CAPCTL01[17] and PWM_CAPCTL01[18]. Whenever the Capture module issues a capture interrupt, the corresponding PWM counter will be reloaded with PERIODx at this moment. Note that the corresponding GPIO pins must be configured as their alternate function before Capture function is enabled.

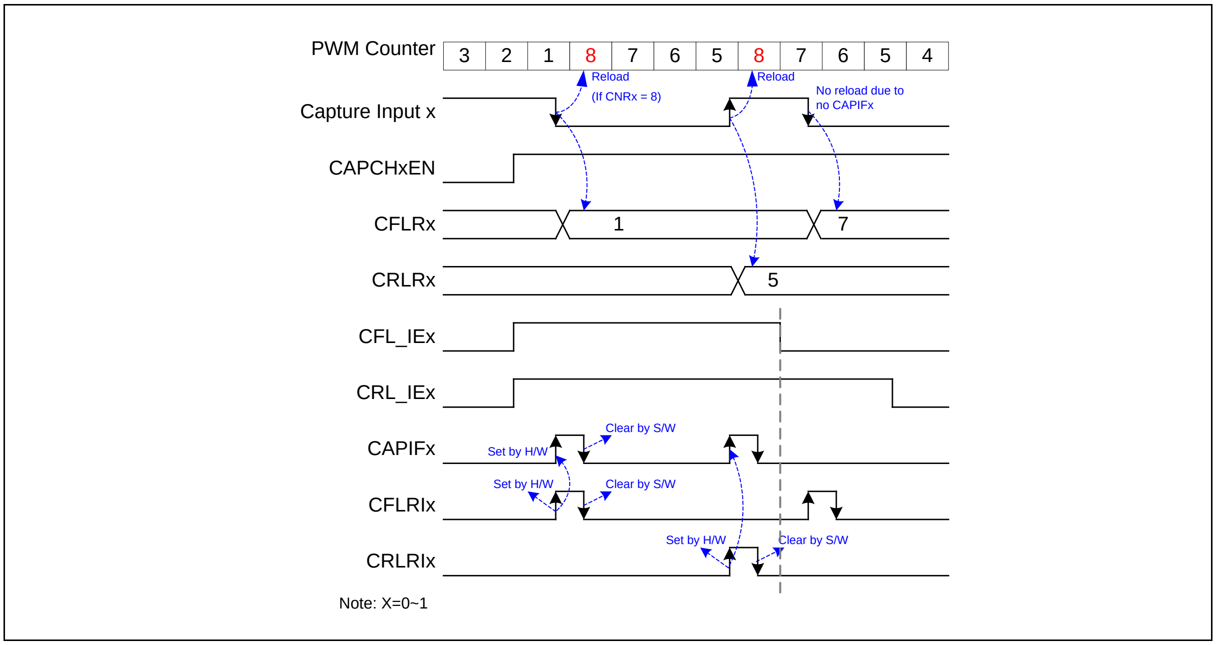


Figure 5-31 Capture Operation Timing

Figure 5-31 Capture Operation Timing demonstrates the case where PERIOD = 8:

1. The PWM counter will be reloaded with PERIODx=8 when a capture interrupt flag (CAPIFx) is set by a transition on the capture input.
2. The channel low pulse width is given by (PERIOD- RCAPDAT).
3. The channel high pulse width is given by (PERIOD - FCAPDAT).

5.7.9 PWM-Timer Interrupt Architecture

There are two PWM interrupts, PWM0_INT, PWM1_INT, which are multiplexed into PWM0_IRQ. PWM 0 and Capture 0 share one interrupt, PWM1 and Capture 1 share the same interrupt. Figure 5-32 PWM-Timer Interrupt Architecture Diagram demonstrates the architecture of PWM-Timer interrupts.

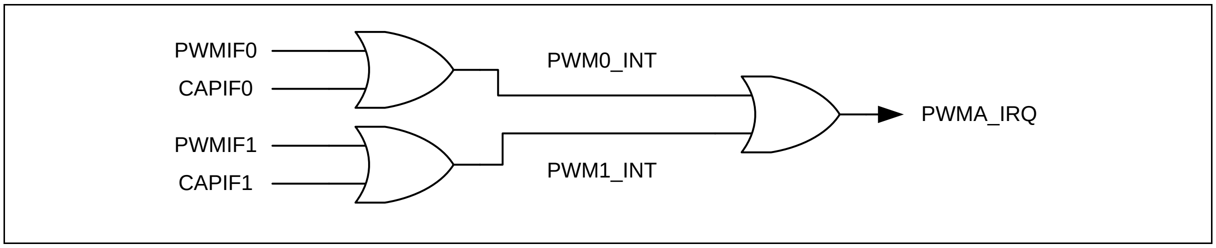


Figure 5-32 PWM-Timer Interrupt Architecture Diagram

5.7.10 PWM-Timer Initialization 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 Stop PWM-timer (PWM_CTL)
4. Setup comparator register (PWM_CMPDAT)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_POEN)
8. Setup the corresponding GPIO pins to PWM function (SYS_GPA_MFP)
9. Enable PWM timer start (Set CNTENx = 1 in PWM_CTL)

5.7.11 PWM-Timer Stop Procedure

- Method 1:

Set 16-bit down counter (PWMx_PERIODx) as 0, and monitor PWMx_CNTx (current value of 16-bit down-counter). When PWMx_CNTx reaches to 0, disable PWM-Timer (CNTENx in PWMx_CTL). ***(Recommended)***

- Method 2:

Set 16-bit down counter (PWMx_PERIODx) as 0. When interrupt request occurs, disable PWM-Timer (CNTENx in PWMx_CTL). ***(Recommended)***

- Method 3:

Disable PWM-Timer directly (CNTENx in PWMx_CTL). ***(Not recommended)***

5.7.12 Capture Start Procedure

1. Setup clock selector (PWM_CLKDIV)
2. Setup prescaler (PWM_CLKPSC)
3. Setup channel enable, rising/falling interrupt enable and input signal inverter on/off (PWM_CAPCTL01, PWM_CAPCTL23)
4. Setup PWM down-counter (PERIOD)
5. Set Capture Input Enable Register (PWM_CAPINEN)
6. Setup the corresponding GPIO pins to PWM function (SYS_GPA_MFP)
7. Enable PWM timer start running (Set CNTENx = 1 in PWM_CTL)

5.7.13 Register Map

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

Register	Offset	R/W	Description	Reset Value
PWM Base Address: **PWM_BA = 0x4004_0000**				
PWM_CLKPSC	PWM_BA+0x000	R/W	PWM Prescaler Register	0x0000_0000
PWM_CLKDIV	PWM_BA+0x004	R/W	PWM Clock Select Register	0x0000_0000
PWM_CTL	PWM_BA+0x008	R/W	PWM Control Register	0x0000_0000
PWM_PERIOD0	PWM_BA+0x00C	R/W	PWM Counter Register 0	0x0000_0000
PWM_CMPDAT0	PWM_BA+0x010	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CNT0	PWM_BA+0x014	R	PWM Data Register 0	0x0000_0000
PWM_PERIOD1	PWM_BA+0x018	R/W	PWM Counter Register 1	0x0000_0000
PWM_CMPDAT1	PWM_BA+0x01C	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CNT1	PWM_BA+0x020	R	PWM Data Register 1	0x0000_0000
PWM_PERIOD2	PWM_BA+0x024	R/W	PWM Counter Register 2	0x0000_0000
PWM_CMPDAT2	PWM_BA+0x028	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CNT2	PWM_BA+0x02C	R	PWM Data Register 2	0x0000_0000
PWM_PERIOD3	PWM_BA+0x030	R/W	PWM Counter Register 3	0x0000_0000
PWM_CMPDAT3	PWM_BA+0x034	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CNT3	PWM_BA+0x038	R	PWM Data Register 3	0x0000_0000
PWM_INTEN	PWM_BA+0x040	R/W	PWM Interrupt Enable Register	0x0000_0000
PWM_INTSTS	PWM_BA+0x044	R/W	PWM Interrupt Flag Register	0x0000_0000
PWM_CAPCTL01	PWM_BA+0x050	R/W	Capture Control Register For Pair Of PWM0CH0 And PWM0CH1	0x0000_0000
PWM_CAPCTL23	PWM_BA+0x054	R/W	Capture Control Register For Pair Of PWM0CH2 And PWM0CH3	0x0000_0000
PWM_RCAPDAT0	PWM_BA+0x058	R	Capture Rising Latch Register (Channel 0)	0x0000_0000
PWM_FCAPDAT0	PWM_BA+0x05C	R	Capture Falling Latch Register (Channel 0)	0x0000_0000
PWM_RCAPDAT1	PWM_BA+0x060	R	Capture Rising Latch Register (Channel 1)	0x0000_0000
PWM_FCAPDAT1	PWM_BA+0x064	R	Capture Falling Latch Register (Channel 1)	0x0000_0000
PWM_RCAPDAT2	PWM_BA+0x068	R	Capture Rising Latch Register (Channel 2)	0x0000_0000
PWM_FCAPDAT2	PWM_BA+0x06C	R	Capture Falling Latch Register (Channel 2)	0x0000_0000

PWM_RCAPDAT3	PWM_BA+0x070	R	Capture Rising Latch Register (Channel 3)	0x0000_0000
PWM_FCAPDAT3	PWM_BA+0x074	R	Capture Falling Latch Register (Channel 3)	0x0000_0000
PWM_CAPINEN	PWM_BA+0x078	R/W	Capture Input Enable Register	0x0000_0000
PWM_POEN	PWM_BA+0x07C	R/W	PWM0 Output Enable Register for CH0~CH3	0x0000_0000

5.7.14 Register Description

PWM Pre-Scale Register (PWM_CLKPSC)

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

31	30	29	28	27	26	25	24
DTCNT23							
23	22	21	20	19	18	17	16
DTCNT01							
15	14	13	12	11	10	9	8
CLKPSC23							
7	6	5	4	3	2	1	0
CLKPSC01							

Bits	Description	
[31:24]	**DTCNT23**	**Dead Zone Interval Register for Pair of PWM0CH2 and PWM0CH3** These 8 bits determine dead zone length. The unit time of dead zone length is that from clock selector 0.
[23:16]	**DTCNT01**	**Dead Zone Interval Register for Pair of PWM0CH0 and PWM0CH1** These 8 bits determine dead zone length. The unit time of dead zone length is that from clock selector 0.
[15:8]	**CLKPSC23**	**Clock Pre-scaler for Pair of PWM0CH2 and PWM0CH3** Clock input is divided by (CLKPSC23 + 1) If CLKPSC23 = 0, then the pre-scaler output clock will be stopped. This implies PWM counter 2 and 3 will also be stopped.
[7:0]	**CLKPSC01**	**Clock Pre-scaler Pair of PWM0CH0 and PWM0CH1** Clock input is divided by (CLKPSC01 + 1) If CLKPSC01 = 0, then the pre-scaler output clock will be stopped. This implies PWM counter 0 and 1 will also be stopped.

PWM Clock Select Register (PWM_CLKDIV)

Register	Offset	R/W	Description	Reset Value
PWM_CLKDIV	PWM_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	CLKDIV3			Reserved	CLKDIV2		
7	6	5	4	3	2	1	0
Reserved	CLKDIV1			Reserved	CLKDIV0		

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**CLKDIV3**	**Timer 3 Clock Source Selection** (Table is as CLKDIV0)
[11]	**Reserved**	Reserved.
[10:8]	**CLKDIV2**	**Timer 2 Clock Source Selection** (Table is as CLKDIV0)
[7]	**Reserved**	Reserved.
[6:4]	**CLKDIV1**	**Timer 1 Clock Source Selection** (Table is as CLKDIV0)
[3]	**Reserved**	Reserved.
[2:0]	**CLKDIV0**	**Timer 0 Clock Source Selection** 0 = Input clock divided by 2 1 = Input clock divided by 4 2 = Input clock divided by 8 3 = Input clock divided by 16 4 = Input clock divided by 1

PWM Control Register (PWM_CTL)

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

31	30	29	28	27	26	25	24
Reserved				CNTMODE3	PINV3	Reserved	CNTEN3
23	22	21	20	19	18	17	16
Reserved				CNTMODE2	PINV2	Reserved	CNTEN2
15	14	13	12	11	10	9	8
Reserved				CNTMODE1	PINV1	Reserved	CNTEN1
7	6	5	4	3	2	1	0
Reserved		DTEN23	DTEN01	CNTMODE0	PINV0	Reserved	CNTEN0

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**CNTMODE3**	**PWM-timer 3 Auto-reload/One-shot Mode** 0 = One-Shot Mode. 1 = Auto-load Mode. **Note:** A rising transition of this bit will cause PWM_PERIOD3 and PWM_CMPDAT3 to be cleared.
[26]	**PINV3**	**PWM-timer 3 Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.
[25]	**Reserved**	Reserved.
[24]	**CNTEN3**	**PWM-timer 3 Enable/Disable Start Run** 0 = Stop PWM-Timer 3. 1 = Enable PWM-Timer 3 Start/Run.
[23:20]	**Reserved**	Reserved.
[19]	**CNTMODE2**	**PWM-timer 2 Auto-reload/One-shot Mode** 0 = One-Shot Mode. 1 = Auto-load Mode. **Note:** A rising transition of this bit will cause PWM_PERIOD2 and PWM_CMPDAT2 to be cleared.
[18]	**PINV2**	**PWM-timer 2 Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.

[16]	**CNTEN2**	**PWM-timer 2 Enable/Disable Start Run** 0 = Stop PWM-Timer 2. 1 = Enable PWM-Timer 2 Start/Run.
[15:12]	**Reserved**	Reserved.
[11]	**CNTMODE1**	**PWM-timer 1 Auto-reload/One-shot Mode** 0 = One-Shot Mode. 1 = Auto-load Mode. **Note:** A rising transition of this bit will cause PWM_PERIOD1 and PWM_CMPDAT1 to be cleared.
[10]	**PINV1**	**PWM-timer 1 Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.
[9]	**Reserved**	Reserved.
[8]	**CNTEN1**	**PWM-timer 1 Enable/Disable Start Run** 0 = Stop PWM-Timer 1. 1 = Enable PWM-Timer 1 Start/Run.
[7:6]	**Reserved**	Reserved.
[5]	**DTEN23**	**Dead-zone 23 Generator Enable/Disable Pair of PWM0CH2 and PWM0CH3** 0 = Disable. 1 = Enable. **Note:** When Dead-Zone Generator is enabled, the pair of PWM0CH2 and PWM0CH3 become a complementary pair.
[4]	**DTEN01**	**Dead-zone 01 Generator Enable/Disable Pair of PWM0CH0 and PWM0CH1** 0 = Disable. 1 = Enable. **Note:** When Dead-Zone Generator is enabled, the pair of PWM0CH0 and PWM0CH1 become a complementary pair.
[3]	**CNTMODE0**	**PWM-timer 0 Auto-reload/One-shot Mode** 0 = One-Shot Mode. 1 = Auto-reload Mode. **Note:** A rising transition of this bit will cause PWM_PERIOD0 and PWM_CMPDAT0 to be cleared.
[2]	**PINV0**	**PWM-timer 0 Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.
[1]	**Reserved**	Reserved.
[0]	**CNTEN0**	**PWM-timer 0 Enable/Disable Start Run** 0 = Stop PWM-Timer 0 Running. 1 = Enable PWM-Timer 0 Start/Run.

PWM Counter Register 1-0 (PWM_PERIOD1-0)

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD0	PWM_BA+0x00C	R/W	PWM Counter Register 0	0x0000_0000
PWM_PERIOD1	PWM_BA+0x018	R/W	PWM Counter Register 1	0x0000_0000
PWM_PERIOD2	PWM_BA+0x024	R/W	PWM Counter Register 2	0x0000_0000
PWM_PERIOD3	PWM_BA+0x030	R/W	PWM Counter 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**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**PWM Counter/Timer Reload Value** PERIOD determines the PWM period. PWM frequency = PWM0CHx_CLK/(prescale+1)*(clock divider)/(PERIOD+1);. Duty ratio = (CMP+1)/(PERIOD+1). 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. (Unit = one PWM clock cycle). **Note:** Any write to PERIOD will take effect in next PWM cycle.

PWM Comparator Register 1-0 (PWM_CMPDAT)

Register	Offset	R/W	Description	Reset Value
PWM_CMPDAT0	PWM_BA+0x010	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1	PWM_BA+0x01C	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2	PWM_BA+0x028	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3	PWM_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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**PWM Comparator Register** CMP determines the PWM duty cycle. PWM frequency = PWM0CHx_CLK/(prescale+1)*(clock divider)/(PERIOD+1);. Duty Cycle = (CMP+1)/(PERIOD+1). 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. (Unit = one PWM clock cycle). Note: Any write to CMP will take effect in next PWM cycle.

PWM Data Register 1-0 (PWM_CNT)

Register	Offset	R/W	Description	Reset Value
PWM_CNT0	PWM_BA+0x014	R	PWM Data Register 0	0x0000_0000
PWM_CNT1	PWM_BA+0x020	R	PWM Data Register 1	0x0000_0000
PWM_CNT2	PWM_BA+0x02C	R	PWM Data Register 2	0x0000_0000
PWM_CNT3	PWM_BA+0x038	R	PWM Data 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**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNT**	**PWM Data 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	PWM_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				**PIEN3**	**PIEN2**	**PIEN1**	**PIEN0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PIEN3**	**PWM Timer 3 Interrupt Enable** 0 = Disable. 1 = Enable.
[2]	**PIEN2**	**PWM Timer 2 Interrupt Enable** 0 = Disable. 1 = Enable.
[1]	**PIEN1**	**PWM Timer 1 Interrupt Enable** 0 = Disable. 1 = Enable.
[0]	**PIEN0**	**PWM Timer 0 Interrupt Enable** 0 = Disable. 1 = Enable.

PWM Interrupt Flag Register (PWM_INTSTS)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS	PWM_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				**PIF3**	**PIF2**	**PIF1**	**PIF0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PIF3**	**PWM Timer 3 Interrupt Flag** Flag is set by hardware when PWM0CH3 down counter reaches zero, software can clear this bit by writing ‘1’ to it.
[2]	**PIF2**	**PWM Timer 2 Interrupt Flag** Flag is set by hardware when PWM0CH2 down counter reaches zero, software can clear this bit by writing ‘1’ to it.
[1]	**PIF1**	**PWM Timer 1 Interrupt Flag** Flag is set by hardware when PWM0CH1 down counter reaches zero, software can clear this bit by writing ‘1’ to it.
[0]	**PIF0**	**PWM Timer 0 Interrupt Flag** Flag is set by hardware when PWM0CH0 down counter reaches zero, software can clear this bit by writing ‘1’ to it.

Note: User can clear each interrupt flag by writing a one to corresponding bit in PWM_INTSTS.

Capture Control Register (PWM_CAPCTL01)

Register	Offset	R/W	Description	Reset Value
PWM_CAPCTL01	PWM_BA+0x050	R/W	Capture Control Register For Pair Of PWM0CH0 And PWM0CH1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CFLIF1	CRLIF1	Reserved	CAPIF1	CAPEN1	CFLIEN1	CRLIEN1	CAPINV1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CFLIF0	CRLIF0	Reserved	CAPIF0	CAPEN0	CFLIEN0	CRLIEN0	CAPINV0

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CFLIF1**	**PWM_FCAPDAT1 Latched Indicator Bit** When input channel 1 has a **falling** transition, PWM_FCAPDAT1 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.
[22]	**CRLIF1**	**PWM_RCAPDAT1 Latched Indicator Bit** When input channel 1 has a **rising** transition, PWM_RCAPDAT1 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.
[20]	**CAPIF1**	**Capture1 Interrupt Indication Flag** If channel 1 rising latch interrupt is enabled (CRLIEN1 = 1), a rising transition at input channel 1 will result in CAPIF1 to high; Similarly, a falling transition will cause CAPIF1 to be set high if channel 1 falling latch interrupt is enabled (CFLIEN1 = 1). This flag is cleared by software writing a ‘1’ to it.
[19]	**CAPEN1**	**Capture Channel 1 Transition Enable/Disable** 0 = Disable capture function on channel 1. 1 = Enable capture function on channel 1. 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.
[18]	**CFLIEN1**	**Channel 1 Falling Latch Interrupt Enable** 0 = Disable falling edge latch interrupt. 1 = Enable falling edge latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.

[17]	**CRLIEN1**	**Channel 1 Rising Latch Interrupt Enable** 0 = Disable rising edge latch interrupt. 1 = Enable rising edge latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.
[16]	**CAPINV1**	**Channel 1 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer
[15:8]	**Reserved**	Reserved.
[7]	**CFLIF0**	**PWM_FCAPDAT0 Latched Indicator Bit** When input channel 0 has a **falling** transition, PWM_FCAPDAT0 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]	**CRLIF0**	**PWM_RCAPDAT0 Latched Indicator Bit** When input channel 0 has a **rising** transition, PWM_RCAPDAT0 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.
[4]	**CAPIF0**	**Capture0 Interrupt Indication Flag** If channel 0 rising latch interrupt is enabled (CRLIEN0 = 1), a rising transition at input channel 0 will result in CAPIF0 to high; Similarly, a falling transition will cause CAPIF0 to be set high if channel 0 falling latch interrupt is enabled (CFLIEN0 = 1). This flag is cleared by software writing a ‘1’ to it.
[3]	**CAPEN0**	**Capture Channel 0 Transition Enable/Disable** 0 = Disable capture function on channel 0. 1 = Enable capture function on channel 0. 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]	**CFLIEN0**	**Channel 0 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]	**CRLIEN0**	**Channel 0 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]	**CAPINV0**	**Channel 0 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer

Capture Control Register (PWM_CAPCTL23)

Register	Offset	R/W	Description	Reset Value
PWM_CAPCTL23	PWM_BA+0x054	R/W	Capture Control Register For Pair Of PWM0CH2 And PWM0CH3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CFLIF3	CRLIF3	Reserved	CAPIF3	CAPEN3	CFLIEN3	CRLIEN3	CAPINV3
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CFLIF2	CRLIF2	Reserved	CAPIF2	CAPEN2	CFLIEN2	CRLIEN2	CAPINV2

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CFLIF3**	**PWM_FCAPDAT3 Latched Indicator Bit** When input channel 3 has a **falling** transition, PWM_FCAPDAT3 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.
[22]	**CRLIF3**	**PWM_RCAPDAT3 Latched Indicator Bit** When input channel 3 has a **rising** transition, PWM_RCAPDAT3 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.
[21]	**Reserved**	Reserved.
[20]	**CAPIF3**	**Capture3 Interrupt Indication Flag** If channel 3 rising latch interrupt is enabled (CRLIEN3 = 1), a rising transition at input channel 3 will result in CAPIF3 to high; Similarly, a falling transition will cause CAPIF3 to be set high if channel 3 falling latch interrupt is enabled (CFLIEN3 = 1). This flag is cleared by software writing a '1' to it.
[19]	**CAPEN3**	**Capture Channel 3 Transition Enable/Disable** 0 = Disable capture function on channel 1. 1 = Enable capture function on channel 1. 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.
[18]	**CFLIEN3**	**Channel 3 Falling Latch Interrupt Enable** 0 = Disable falling edge latch interrupt. 1 = Enable falling edge latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.

[17]	**CRLIEN3**	**Channel 3 Rising Latch Interrupt Enable** 0 = Disable rising edge latch interrupt. 1 = Enable rising edge latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.
[16]	**CAPINV3**	**Channel 3 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer
[15:8]	**Reserved**	Reserved.
[7]	**CFLIF2**	**PWM_FCAPDAT2 Latched Indicator Bit** When input channel 2 has a **falling** transition, PWM_FCAPDAT2 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]	**CRLIF2**	**PWM_RCAPDAT2 Latched Indicator Bit** When input channel 2 has a **rising** transition, PWM_RCAPDAT2 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.
[4]	**CAPIF2**	**Capture2 Interrupt Indication Flag** If channel 2 rising latch interrupt is enabled (CRLIEN2 = 1), a rising transition at input channel 2 will result in CAPIF2 to high; Similarly, a falling transition will cause CAPIF2 to be set high if channel 2 falling latch interrupt is enabled (CFLIEN2 = 1). This flag is cleared by software writing a ‘1’ to it.
[3]	**CAPEN2**	**Capture Channel 2 Transition Enable/Disable** 0 = Disable capture function on channel 0. 1 = Enable capture function on channel 0. 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]	**CFLIEN2**	**Channel 2 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]	**CRLIEN2**	**Channel 2 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]	**CAPINV2**	**Channel 2 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer

Capture Rising Latch Register n (PWM_RCAPDATn)

Register	Offset	R/W	Description	Reset Value
PWM_RCAPDAT0	PWM_BA+0x058	R	Capture Rising Latch Register (Channel 0)	0x0000_0000
PWM_RCAPDAT1	PWM_BA+0x060	R	Capture Rising Latch Register (Channel 1)	0x0000_0000
PWM_RCAPDAT2	PWM_BA+0x068	R	Capture Rising Latch Register (Channel 2)	0x0000_0000
PWM_RCAPDAT3	PWM_BA+0x070	R	Capture Rising Latch Register (Channel 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**
RCAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RCAPDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[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 n(PWM_FCAPDATn)

Register	Offset	R/W	Description	Reset Value
PWM_FCAPDAT0	PWM_BA+0x05C	R	Capture Falling Latch Register (Channel 0)	0x0000_0000
PWM_FCAPDAT1	PWM_BA+0x064	R	Capture Falling Latch Register (Channel 1)	0x0000_0000
PWM_FCAPDAT2	PWM_BA+0x06C	R	Capture Falling Latch Register (Channel 2)	0x0000_0000
PWM_FCAPDAT3	PWM_BA+0x074	R	Capture Falling Latch Register (Channel 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**
FCAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FCAPDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[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.

Capture Input Enable Register (PWM_CAPINEN)

Register	Offset	R/W	Description	Reset Value
PWM_CAPINEN	PWM_BA+0x078	R/W	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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**CAPINEN**			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**CAPINEN**	**Capture Input Enable Register** 0 : OFF (GPA[13:12], GPB[15:14] pin input disconnected from Capture block) 1 : ON (GPA[13:12] , GPB[15:14] pin, if in PWM alternative function, will be configured as an input and fed to capture function) CAPINEN[3:0] Bit [3][2][1][0] Bit xxx1 : Capture channel 0 is from GPA [12] Bit xx1x : Capture channel 1 is from GPA [13] Bit x1xx : Capture channel 2 is from GPB [14] Bit 1xxx : Capture channel 3 is from GPB [15]

PWM Output Enable Register (PWM_POEN)

Register	Offset	R/W	Description	Reset Value
PWM_POEN	PWM_BA+0x07C	R/W	PWM0 Output Enable Register for CH0~CH3	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				**POEN3**	**POEN2**	**POEN1**	**POEN0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**POEN3**	**PWM0CH3 Output Enable Register** 0 = Disable PWM0CH3 output to pin. 1 = Enable PWM0CH3 output to pin. Note: The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)
[2]	**POEN2**	**PWM0CH2 Output Enable Register** 0 = Disable PWM0CH2 output to pin. 1 = Enable PWM0CH2 output to pin. Note: The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)
[1]	**POEN1**	**PWM0CH1 Output Enable Register** 0 = Disable PWM0CH1 output to pin. 1 = Enable PWM0CH1 output to pin. Note: The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)
[0]	**POEN0**	**PWM0CH0 Output Enable Register** 0 = Disable PWM0CH0 output to pin. 1 = Enable PWM0CH 0 output to pin. Note: The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)

5.8 Real Time Clock (RTC)

5.8.1 Overview

Real Time Clock (RTC) unit provides real time clock, calendar and alarm functions. The clock source of the RTC is an external 32.768 kHz crystal connected at pins XI32K and XO32K or from an external 32.768 kHz oscillator output fed to pin XI32K. The RTC unit provides the time (second, minute, hour) in Time Load Register (RTC_TIME) as well as calendar (day, month, year) in Calendar Load Register (RTC_CAL). The data is expressed in BCD (Binary Coded Decimal) format. The unit offers an alarm function whereby the user can preset the alarm time in the Time Alarm Register (RTC_TALM) and alarm calendar in Calendar Alarm Register (RTC_CALM).

The RTC unit supports periodic Time-Tick and Alarm-Match interrupts. The periodic interrupt has 8 period options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second which are selected by RTC_TICK.TTR. When RTC counter in RTC_TIME and RTC_CAL is equal to alarm setting registers RTC_TALM and RTC_CALM, the alarm interrupt flag (RTC_INTSTS.ALMIF) is set and the alarm interrupt is requested if the alarm interrupt is enabled (RTC_INTEN.ALMIEN=1). The RTC Time Tick and Alarm Match can wake the CPU from sleep mode or Standby Power-Down (SPD) mode if the Wakeup CPU function is enabled (RTC_TICK.TWKEN=1).

5.8.2 RTC Features

- Consists of a time counter (second, minute, hour) and calendar counter (day, month, year).
- Alarm register (second, minute, hour, day, month, year).
- 12-hour or 24-hour mode is selectable.
- Automatic leap year compensation.
- Day of week counter.
- Frequency compensate register (FCR).
- All time and calendar registers are expressed in BCD code.
- Support periodic time tick interrupt with 8 period options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second.
- Support RTC Time-Tick and Alarm-Match interrupt
- Support CPU wakeup from sleep or standby power-down mode.

5.8.3 RTC Block Diagram

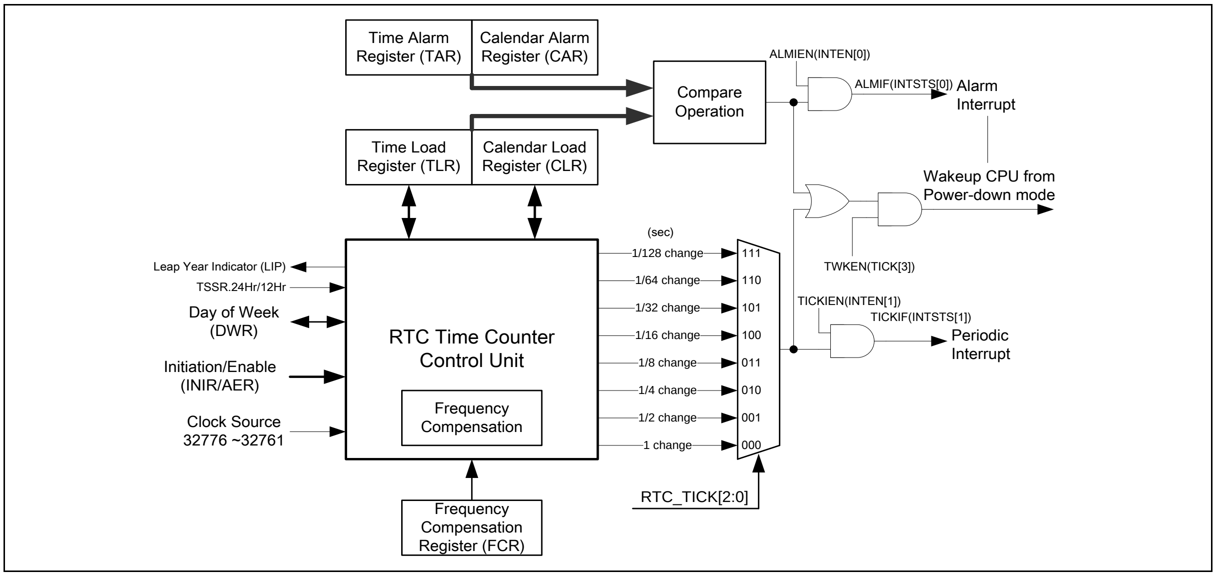


Figure 5-33 RTC Block Diagram

5.8.4 Functional Description

5.8.4.1 Access to RTC register

Due to clock frequency difference between RTC clock and system clock, when the user writes new data to any one of the RTC registers, the register will not be updated until 2 RTC clock periods later (60us). The programmer should take this into consideration for determining access sequence between RTC_CLKFMT, RTC_TALM and RTC_TIME.

In addition, the RTC block does not check whether written data is out of bounds for a valid BCD time or calendar load. RTC does not check validity of RTC_WEEKDAY and RTC_CAL write either.

5.8.4.2 RTC Initiation

When RTC block is powered on, programmer must write 0xA5EB1357 to RTC_INIT register to reset all logic. RTC_INIT acts as a hardware reset circuit. Once RTC_INIT has been set to0xA5EB1357, internal reset operation begins. When reset operation is finished, RTC_INIT[0] is set by hardware and RTC is ready for operation.

5.8.4.3 RTC Read/Write Enable

Register RTC_RWEN[15:0] serves as the RTC read/write password to protect RTC registers. RTC_RWEN[15:0] have to be set to 0xA965 to enable access. Once set, it will take effect 512 RTC clocks later (about 15ms). Programmer can read RTC enabled status flag in RTC_RWEN.ENF to check whether RTC is access enabled. Access is automatically cleared after 200ms.

5.8.4.4 Frequency Compensation

The RTC Frequency Compensation Register (RTC_FREQADJ)allows software to configure digital compensation to the 32768Hz clock input. The RTC_FREQADJ allows compensation of a clock input in the range from 32761Hz to 32776Hz. If desired, RTC clock can be measured during manufacture from a GPIO pin and compensation value calculated and stored in flash memory for retrieval when the product is first powered on. Following are compensation examples for a higher or lower measured frequency clock input.

Table 5-7 RTC Frequency Compensation Example

Example 1: Frequency Counter Measurement : 32773.65Hz (> 32768 Hz) Integer Part: 32773 = 0x8005 RTC_FREQADJ. INTEGER= (32773 – 32761) = 12 = 0x0C Fractional Part: 0.65 X 60 = 39 = 0x27 RTC_FREQADJ. FRACTION= 0x27
Example 2 Frequency counter measurement : 32765.27Hz (< 32768 Hz) Integer part: 32765 = 0x7ffd RTC_FREQADJ.INTEGER = (32765 – 32761) = 4 = 0x04 Fractional part: 0.27 x 60 = 16.2 = 0x10 RTC_FREQADJ.FRACTION = 0x10

5.8.4.5 Time and Calendar counter

RTC_TIME and RTC_CAL are used to load the time and calendar. RTC_TALM and RTC_CALM are used to set the alarm. They are all represented by a BCD format, see register descriptions for digit assignments.

5.8.4.6 12/24 hour Time Scale Selection

RTC can be selected to report time in either a 12 or 24hour time scale. If 12 hour mode is selected then AM/PM indication is provided by the hour digit being >=2, see register description RTC_CLKFMT for details. The 12/24 hour time scale selection depends on RTC_CLKFMT.24HEN.

5.8.4.7 Day of the week counter

The RTC unit provides day of week in Day of the Week Register (RTC_WEEKDAY). The value is defined from 0 to 6 to represent Sunday to Saturday respectively.

5.8.4.8 Periodic Time Tick Interrupt

The periodic interrupt has 8 period option 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second which are selected by RTC_TICK.TICKSEL[2:0]. When periodic time tick interrupt is enabled by setting RTC_INTEN.TICKEN to 1, the Periodic Time Tick Interrupt is requested as selected by RTC_TICK register.

5.8.4.9 Alarm Time Interrupt

When RTC counter in RTC_TIME and RTC_CAL is equal to alarm setting in RTC_TALM and RTC_CALM the alarm interrupt flag (RTC_INTSTS.ALMIF) is set. If alarm interrupt is enabled (RTC_INTEN.ALMIEN=1) the alarm interrupt is also requested.

5.8.4.10 Additional Notes

1. RTC_TALM, RTC_CALM, RTC_TIME and RTC_CAL registers are all BCD counter.
2. Programmer has to make sure that values loaded are reasonable. For example, some invalid RTC_CAL values would be 201a (year), 13 (month), 00 (day).
3. Reset state :

Register	**Reset State**
RTC_RWEN	0
RTC_CAL	05/1/1 (year/month/day)

RTC_TIME	00:00:00 (hour : minute : second)
RTC_CALM	00/00/00 (year/month/day)
RTC_TALM	00:00:00 (hour : minute : second)
RTC_CLKFMT	1 (24 hr. mode)
RTC_WEEKDAY	6 (Saturday)
RTC_INTEN	0
RTC_INTSTS	0
RTC_LEAPYEAR	0
RTC_TICK	0
PWRTOUT	5555

4. In RTC_TIME and RTC_TALM, only 2 BCD digits are used to express “year”. It is assumed that 2 BCD digits of xY denote 20xY, but not 19xY or 21xY.

5.8.5 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
RTC Base Address: **RTC_BA = 0x4000_8000**				
RTC_INIT	RTC_BA+0x000	R/W	RTC Initialization Register	0x0000_0000
RTC_RWEN	RTC_BA+0x004	R/W	RTC Access Enable Register	0x0000_0000
RTC_FREQADJ	RTC_BA+0x008	R/W	RTC Frequency Compensation Register	0x0000_0700
RTC_TIME	RTC_BA+0x00C	R/W	Time Load Register	0x0000_0000
RTC_CAL	RTC_BA+0x010	R/W	Calendar Load Register	0x0000_0000
RTC_CLKFMT	RTC_BA+0x014	R/W	Time Scale Selection Register	0x0000_0001
RTC_WEEKDAY	RTC_BA+0x018	R/W	Day of the Week Register	0x0000_0000
RTC_TALM	RTC_BA+0x01C	R/W	Time Alarm Register	0x0000_0000
RTC_CALM	RTC_BA+0x020	R/W	Calendar Alarm Register	0x0000_0000
RTC_LEAPYEAR	RTC_BA+0x024	R	Leap year Indicator Register	0x0000_0000
RTC_INTEN	RTC_BA+0x028	R/W	RTC Interrupt Enable Register	0x0000_0000
RTC_INTSTS	RTC_BA+0x02C	R/W	RTC Interrupt Indicator Register	0x0000_0000
RTC_TICK	RTC_BA+0x030	R/W	RTC Time Tick Register	0x0000_0000

5.8.6 Register Description

RTC Initiation Register (RTC_INIT)

Register	Offset	R/W	Description	Reset Value
RTC_INIT	RTC_BA+0x000	R/W	RTC Initialization Register	0x0000_0000

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

Bits	Description	
[31:1]	**INIT**	**RTC Initialization** After a power-on reset (POR) RTC block should be initialized by writing 0xA5EB1357 to INIT. This will force a hardware reset then release all logic and counters.
[0]	**ATVSTS**	**RTC Active Status (Read Only)** 0: RTC is in reset state 1: RTC is in normal active state.

RTC Access Enable Register (RTC_RWEN)

Register	Offset	R/W	Description	Reset Value
RTC_RWEN	RTC_BA+0x004	R/W	RTC Access Enable Register	0x0000_0000

23	22	21	20	19	18	17	16
Reserved							**RWENF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RWEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RWEN							

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**RWENF**	**RTC Register Access Enable Flag (Read Only)** 1 = RTC register read/write enable. 0 = RTC register read/write disable. This bit will be set after RWEN[15:0] register is set to 0xA965, it will clear automatically in 512 RTC clock cycles or RWEN[15:0] ! = 0xA965. The effect of RTC_RWEN.RWENF is as the below. Table 5-8 RTC_RWEN.RWENF Register Access Effect. Register : RWENF = 1 : RWENF = 0. RTC_INIT : R/W : R/W RTC_FREQADJ : R/W : - RTC_TIME : R/W : R RTC_CAL : R/W : R RTC_CLKFMT : R/W : R/W RTC_WEEKDAY : R/W : R RTC_TALM : R/W : - RTC_CALM : R/W : - RTC_LEAPYEAR : R : R RTC_INTEN : R/W : R/W RTC_INTSTS : R/W : R/W RTC_TICK : R/W : -
[15:0]	**RWEN**	**RTC Register Access Enable Password (Write Only)** 0xA965 = Enable RTC acces..s Others = Disable RTC acces..s

RTC Frequency Compensation Register (RTC_FREQADJ)

Register	Offset	R/W	Description	Reset Value
RTC_FREQADJ	RTC_BA+0x008	R/W	RTC Frequency Compensation 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				**INTEGER**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**FRACTION**					

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**INTEGER**	**Integer Part** Register should contain the value (INT(F_{actual}) – 32761) Ex: Integer part of detected value = 32772,. RTC_FREQADJ.INTEGER = 32772-32761 = 11 (..1011b) The range between 32761 and 32776
[7:6]	**Reserved**	Reserved.
[5:0]	**FRACTION**	**Fractional Part** Formula = (fraction part of detected value) x 60. Refer to Table 5-7 RTC Frequency Compensation Example for the examples.

Note: This register can be read back after the RTC enable is active.

RTC Time Load Register (RTC_TIME)

This register is Read Only until access enable password is written to **RTC_RWEN** register. The register returns the current time.

Register	Offset	R/W	Description	Reset Value
RTC_TIME	RTC_BA+0x00C	R/W	Time Load Register	0x0000_0000

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

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21:20]	**TENHR**	10 Hour Time Digit (0~3)
[19:16]	**HR**	1 Hour Time Digit (0~9)
[15]	**Reserved**	Reserved.
[14:12]	**TENMIN**	10 Min Time Digit (0~5)
[11:8]	**MIN**	1 Min Time Digit (0~9)
[7]	**Reserved**	Reserved.
[6:4]	**TENSEC**	10 Sec Time Digit (0~5)
[3:0]	**SEC**	1 Sec Time Digit (0~9)

Note:

1. RTC_TIME is a BCD counter and RTC will not check loaded data for validity.
2. Valid range is listed in the parenthesis.

RTC Calendar Load Register (RTC_CAL)

This register is Read Only until access enable password is written to **RTC_RWEN** register. The register returns the current date.

Register	Offset	R/W	Description	Reset Value
RTC_CAL	RTC_BA+0x010	R/W	Calendar Load Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**TENYEAR**	10-Year Calendar Digit (0~9)
[19:16]	**YEAR**	1-Year Calendar Digit (0~9)
[15:13]	**Reserved**	Reserved.
[12]	**TENMON**	10-Month Calendar Digit (0~1)
[11:8]	**MON**	1-Month Calendar Digit (0~9)
[7:6]	**Reserved**	Reserved.
[5:4]	**TENDAY**	10-Day Calendar Digit (0~3)
[3:0]	**DAY**	1-Day Calendar Digit (0~9)

RTC Time Scale Selection Register (RTC_CLKFMT)

Register	Offset	R/W	Description	Reset Value
RTC_CLKFMT	RTC_BA+0x014	R/W	Time Scale Selection 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							**24HEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**24HEN**	**24-hour / 12-hour Mode Selection** Determines whether RTC_TIME and RTC_TALM are in 24-hour mode or 12-hour mode 1 = select 24-hour time scale. 0 = select 12-hour time scale with AM and PM indication. The range of 24-hour time scale is between 0 and 23. 12-hour time scale: 01(AM01), 02(AM02), 03(AM03), 04(AM04), 05(AM05), 06(AM06) 07(AM07), 08(AM08), 09(AM09), 10(AM10), 11(AM11), 12(AM12) 21(PM01), 22(PM02), 23(PM03), 24(PM04), 25(PM05), 26(PM06) 27(PM07), 28(PM08), 29(PM09), 30(PM10), 31(PM11), 32(PM12)

RTC Day of the Week Register (RTC_WEEKDAY)

Register	Offset	R/W	Description	Reset Value
RTC_WEEKDAY	RTC_BA+0x018	R/W	Day of the Week 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					WEEKDAY		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**WEEKDAY**	**Day of the Week Register** 0 (Sunday), 1 (Monday), 2 (Tuesday), 3 (Wednesday) 4 (Thursday), 5 (Friday), 6 (Saturday)

RTC Time Alarm Register (RTC_TALM)

Register	Offset	R/W	Description	Reset Value
RTC_TALM	RTC_BA+0x01C	R/W	Time Alarm Register	0x0000_0000

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

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21:20]	**TENHR**	10 Hour Time Digit of Alarm Setting (0~3)[2]
[19:16]	**HR**	1 Hour Time Digit of Alarm Setting (0~9)
[15]	**Reserved**	Reserved.
[14:12]	**TENMIN**	10 Min Time Digit of Alarm Setting (0~5)
[11:8]	**MIN**	1 Min Time Digit of Alarm Setting (0~9)
[7]	**Reserved**	Reserved.
[6:4]	**TENSEC**	10 Sec Time Digit of Alarm Setting (0~5)
[3:0]	**SEC**	1 Sec Time Digit of Alarm Setting (0~9)

Note:

1. RTC_TALM is a BCD digit counter and RTC will not check validity of loaded data. Valid range is listed in the parenthesis.
2. This register can be read back after the RTC unit is active.

RTC Calendar Alarm Register (RTC_CALM)

Register	Offset	R/W	Description	Reset Value
RTC_CALM	RTC_BA+0x020	R/W	Calendar Alarm Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**TENYEAR**	10-Year Calendar Digit of Alarm Setting (0~9)
[19:16]	**YEAR**	1-Year Calendar Digit of Alarm Setting (0~9)
[15:13]	**Reserved**	Reserved.
[12]	**TENMON**	10-Month Calendar Digit of Alarm Setting (0~1)
[11:8]	**MON**	1-Month Calendar Digit of Alarm Setting (0~9)
[7]	**Reserved**	Reserved.
[5:4]	**TENDAY**	10-Day Calendar Digit of Alarm Setting (0~3)
[3:0]	**DAY**	1-Day Calendar Digit of Alarm Setting (0~9)

Note:

1. RTC_TIME is a BCD digit counter and RTC will not check validity loaded data, valid range is listed in the parenthesis.
2. This register can be read back after the RTC unit is active.

RTC Leap year Indication Register (RTC_LEAPYEAR)

Register	Offset	R/W	Description	Reset Value
RTC_LEAPYEAR	RTC_BA+0x024	R	Leap year Indicator 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							**LEAPYEAR**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**LEAPYEAR**	**Leap Year Indication Register (Read Only)** 0 = Current year is not a leap year. 1 = Current year is leap year.

RTC Interrupt Enable Register (RTC_INTEN)

Register	Offset	R/W	Description	Reset Value
RTC_INTEN	RTC_BA+0x028	R/W	RTC 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						**TICKIEN**	**ALMIEN**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TICKIEN**	**Time-tick Interrupt and Wakeup-by-tick Enable** 1 = RTC Time-Tick Interrupt is enabled. 0 = RTC Time-Tick Interrupt is disabled.
[0]	**ALMIEN**	**Alarm Interrupt Enable** 1 = RTC Alarm Interrupt is enabled. 0 = RTC Alarm Interrupt is disabled.

RTC Interrupt Indication Register (RTC_INTSTS)

Register	Offset	R/W	Description	Reset Value
RTC_INTSTS	RTC_BA+0x02C	R/W	RTC Interrupt Indicator 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						**TICKIF**	**ALMIF**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TICKIF**	**RTC Time-tick Interrupt Flag** When RTC Time-Tick Interrupt is enabled (RTC_INTEN.TICKIF=1), RTC unit will set TIF high at the rate selected by RTC_TICK[2:0]. This bit cleared/acknowledged by writing 1 to it. 0= Indicates no Time-Tick Interrupt condition. 1= Indicates RTC Time-Tick Interrupt generated.
[0]	**ALMIF**	**RTC Alarm Interrupt Flag** When RTC Alarm Interrupt is enabled (RTC_INTEN.ALMIF=1), RTC unit will set ALMIF to high once the RTC real time counters RTC_TIME and RTC_CAL reach the alarm setting time registers RTC_TALM and RTC_CALM. This bit cleared/acknowledged by writing 1 to it. 0= Indicates no Alarm Interrupt condition. 1= Indicates RTC Alarm Interrupt generated.

RTC Time-Tick Register (RTC_TICK)

Register	Offset	R/W	Description	Reset Value
RTC_TICK	RTC_BA+0x030	R/W	RTC Time Tick 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				**TWKEN**	**TICKSEL**		

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**TWKEN**	**RTC Timer Wakeup CPU Function Enable Bit** If TWKE is set before CPU is in power-down mode, when a RTC Time-Tick or Alarm Match occurs, CPU will wake up. 0= Disable Wakeup CPU function. 1= Enable the Wakeup function.
[2:0]	**TICKSEL**	**Time Tick Period Select** The RTC time tick period for Periodic Time-Tick Interrupt request. Time Tick (second) : 1 / (2^TTR) Note: This register can be read back after the RTC is active.

5.9 Serial 0 Peripheral Interface (SPI0) Controller

5.9.1 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 ISD91200 series contains an 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 SPI0 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 SPI0 interface supports Dual and Quad IO as is common on serial flash memories, where data is transferred 2 or 4 bits per clock period.

5.9.2 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.
- Byte or word Sleep Suspend Mode.
- PDMA access support.

5.9.3 SPI0 Block Diagram

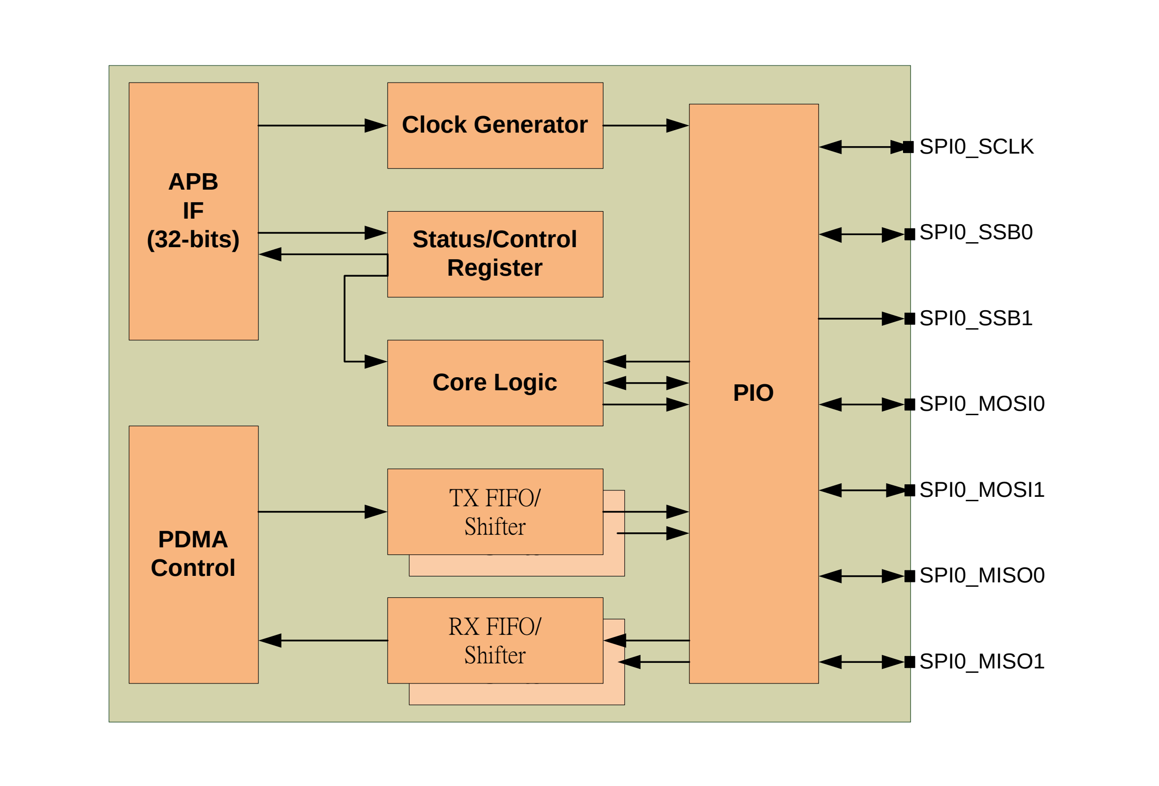


Figure 5-34 SPI0 Block Diagram

5.9.4 SPI0 Function Descriptions

5.9.4.1 SPI Engine Clock and SPI Serial Clock

The SPI controller derives its clock source from the system HCLK as determined by the CLKSEL1 register. The frequency of the SPI master clock is determined by the divisor ratio SPI0_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.

5.9.4.2 Master/Slave Mode

This SPI0 controller can be configured as in master or slave mode by setting the SLAVE bit (SPI0_CTL.SLAVE). In master mode the ISD91200 generates SCLK and SSB signals to access one or more slave devices. In slave mode the ISD91200 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.

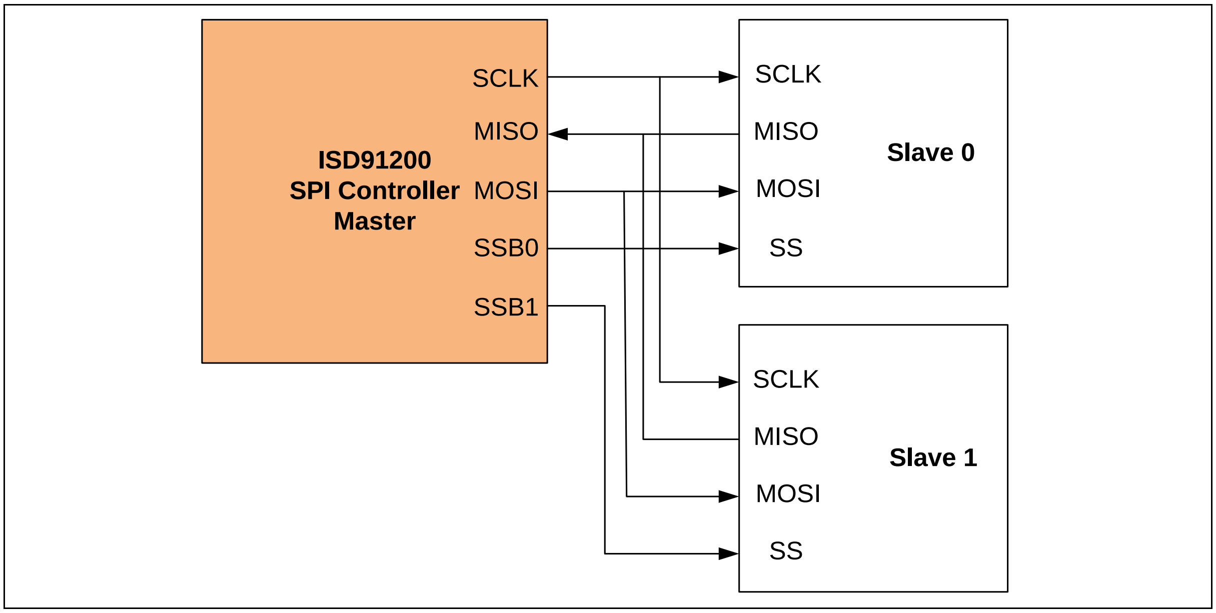


Figure 5-35 SPI Master Mode Application Block Diagram

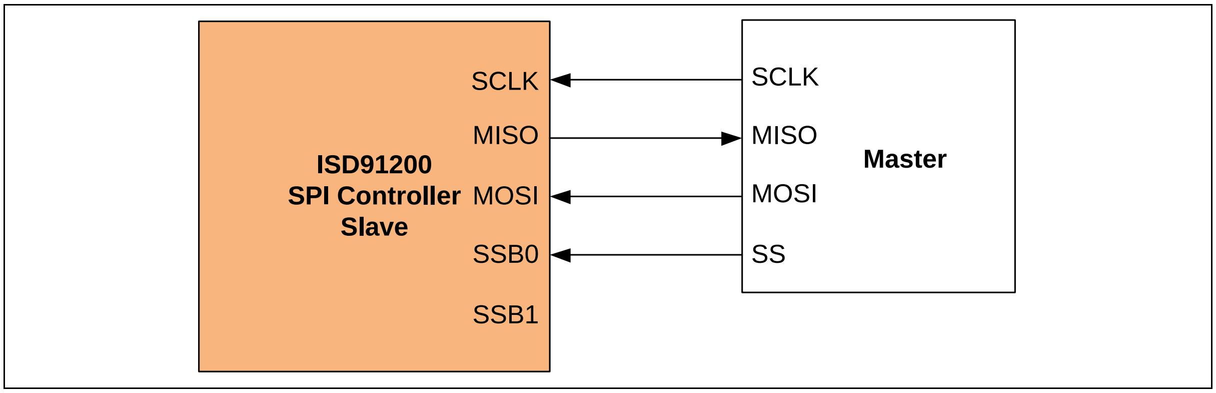


Figure 5-36 SPI Slave Mode Application Block Diagram

5.9.4.3 Slave Select

In master mode, the SPI controller can address up to two off-chip slave devices through the slave select output pins SPI0_SSB0 and SPI0_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 SPI0_SSB0 to address the SPI controller. The slave select signal can be programmed to be active low or active high via the SPI0_SSCTL.SSACTPOL bit. In addition the SPI0_SSCTL.SSLTRIG 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.

5.9.4.4 Automatic Slave Select

In master mode, if the bit SPI0_SSCTL.ASS is set, the slave select signals will be generated automatically and output to SPI0_SSB0 and SPI0_SSB1 pins according to registers SPI0_SSCTL.SS[0] and SPI0_SSCTL.SS[1]. In this mode, SPI0 controller will assert SSB when transaction is triggered and de-assert when data transfer is finished. If the SPI0_SSCTL.ASS bit is cleared, the slave select output signals are asserted and de-asserted by manual setting and clearing the related bits in the SPI0_SSCTL.SS[1:0] register. The active level of the slave select output signals is specified by the SPI0_SSCTL.SSACTPOL 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.

5.9.4.5 Serial Clock

In master mode, writing a divisor into the SPI0_CLKDIV.DIVIDER register will program the output frequency of serial clock to the SPI0_SCLK output port. In slave mode, the off-chip master device drives the serial clock through the SPI0_SCLK.

5.9.4.6 Clock Polarity

The SPI0_CTL.CLKPOL bit defines the serial clock idle state in master mode. If CLKPOL = 1, the output SPI0_SCLK is high in idle state. If CLKPOL=0,it is low in idle state.

5.9.4.7 Transmit/Receive Bit Length

The bit length of a transfer word is defined in SPI0_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.

5.9.4.8 LSB First

The SPI0_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.

5.9.4.9 Transmit Edge

The SPI0_CTL.TXNEG bit determines whether transmit data is changed on the positive or negative edge of the SPI0_SCLK serial clock. If TXNEG=0 then transmitted data will change state on the rising edge of SPI0_SCLK. If TXNEG=1 then transmitted data will change state on the falling edge of SPI0_SCLK.

5.9.4.10 Receive Edge

The SPI0_CTL.RXNEG bit determines whether data is received at either the negative edge or positive edge of serial clock SPI0_SCLK. If RXNEG=1 then data is clocked in on the falling edge of SPI0_SCLK. If RXNEG=0 data is clocked in on the rising edge of SPI0_SCLK. Note that RXNEG should be the inverse of TXNEG for standard SPI operation.

5.9.4.11 Word Suspend

The four bit field SUSPITV (SPI0_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).

.

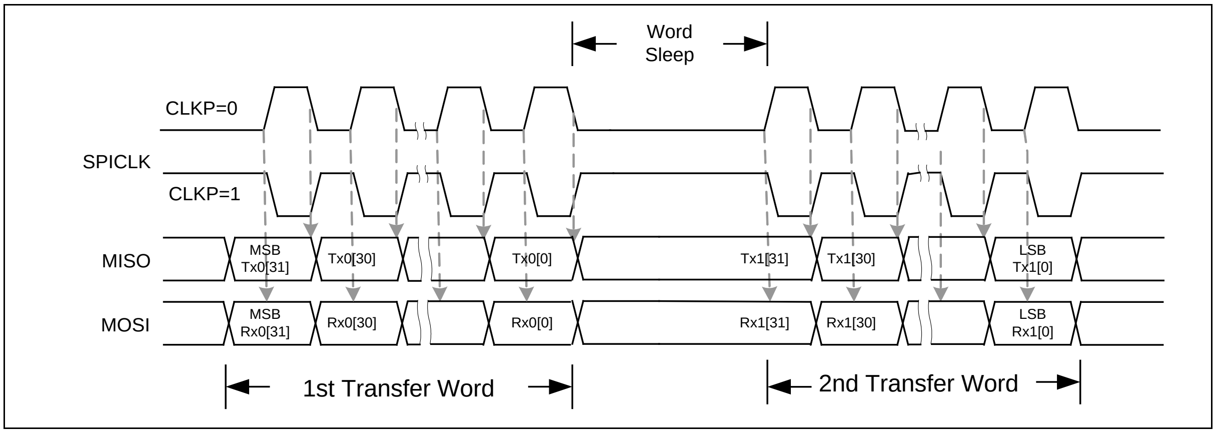


Figure 5-37 Word Sleep Suspend Mode

5.9.4.12 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 (SPI0_CTL.LSB = 0) and the SPI0_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.

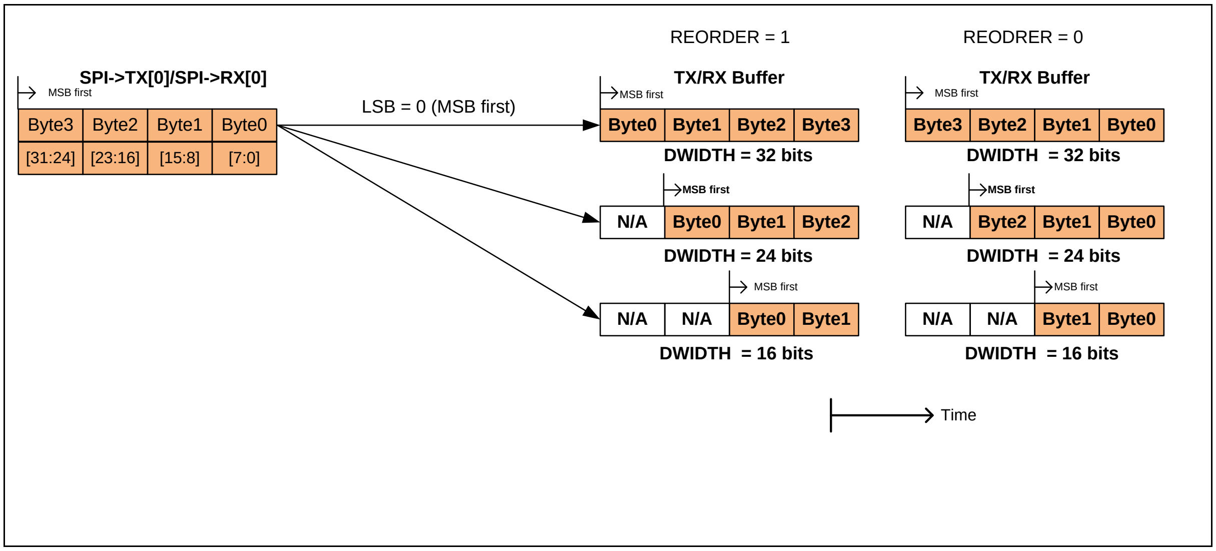
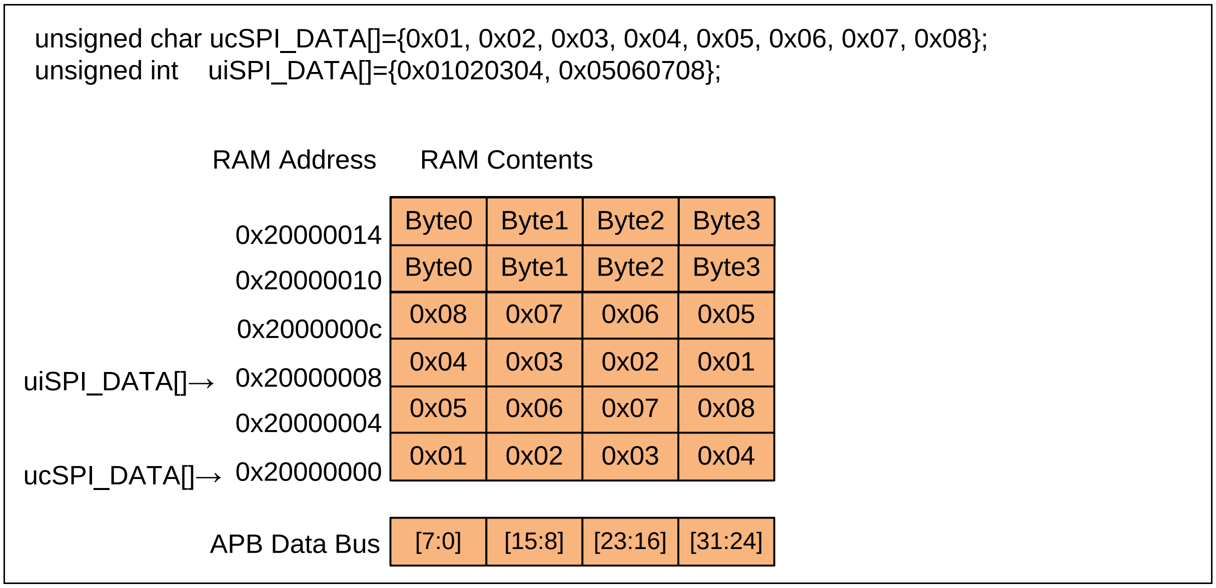


Figure 5-38 Byte Re-Ordering Transfer

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};



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

	RAM Address	RAM Contents			
	0x20000014	Byte0	Byte1	Byte2	Byte3
	0x20000010	Byte0	Byte1	Byte2	Byte3
	0x2000000c	0x08	0x07	0x06	0x05
uiSPI_DATA[]→	0x20000008	0x04	0x03	0x02	0x01
	0x20000004	0x05	0x06	0x07	0x08
ucSPI_DATA[]→	0x20000000	0x01	0x02	0x03	0x04
	APB Data Bus	[7:0]	[15:8]	[23:16]	[31:24]

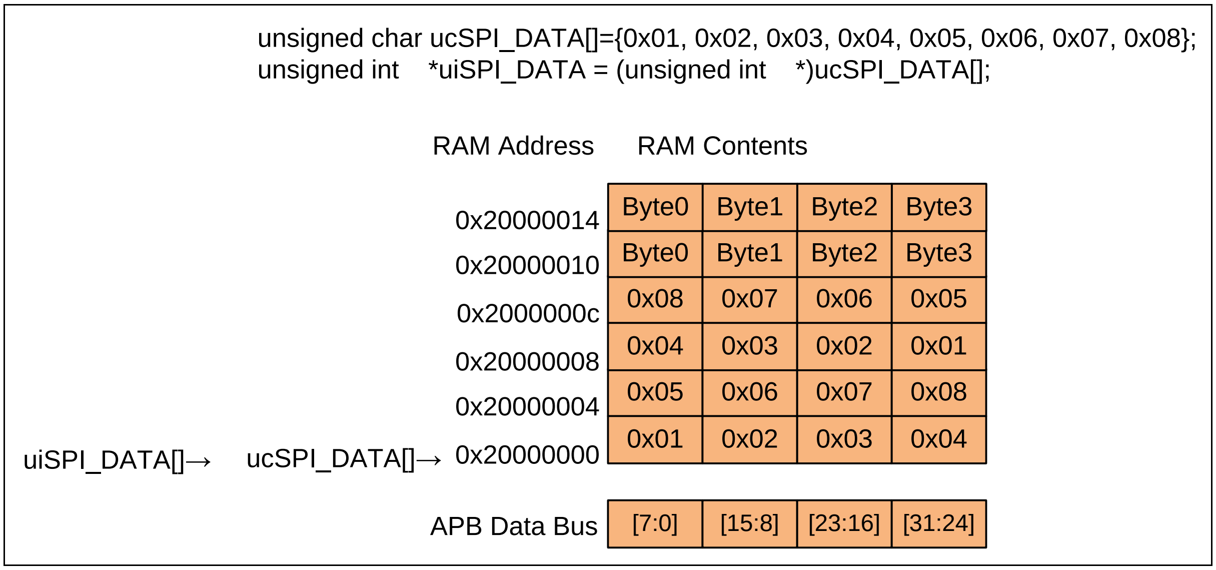
Figure 5-39 Byte Order in Memory

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 SPI0_TX = ucSPI_DATA[i++]
2. Set DWIDTH=32 and send word-by-word SPI0_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.



```
unsigned char ucSPI_DATA[]={0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08};
unsigned int    *uiSPI_DATA = (unsigned int    *)ucSPI_DATA[];
```

		RAM Address	RAM Contents			
		0x20000014	Byte0	Byte1	Byte2	Byte3
		0x20000010	Byte0	Byte1	Byte2	Byte3
		0x2000000c	0x08	0x07	0x06	0x05
		0x20000008	0x04	0x03	0x02	0x01
		0x20000004	0x05	0x06	0x07	0x08
uiSPI_DATA[]→	ucSPI_DATA[]→	0x20000000	0x01	0x02	0x03	0x04
		APB Data Bus	[7:0]	[15:8]	[23:16]	[31:24]

Figure 5-40 Byte Order in Memory

Now if we set DWIDTH=32 and sent word-by-word SPI0_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}.

5.9.4.13 Interrupt

- SPI unit transfer interrupt

As the SPI controller finishes a unit transfer, the unit transfer interrupt flag UNITIF (SPI0_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 UNITIEN (SPI0_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, SPI0_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, SPI0_STATUS[5], will be set to 1. The SPI controller will issue an interrupt if the SLVTOIEN, SPI0_SSCR[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, SPI0_STATUS[6], will be set to 1. The SPI controller will issue an interrupt if the SLVBCEIEN, SPI0_SSCR[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 SPI0_TX register, the under-run event, TXUFIF (SPI0_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, SPI0_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, SPI0_SSCR[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 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 the RXOVIF will be set to 1 and follow-up data will be dropped. The SPI controller will issue an interrupt if the RXOVIEN, SPI0_FIFOCTL[5], 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, SPI0_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, SPI0_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, SPI0_FIFOCTL[2], is set to 1.

5.9.4.14 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, SPI0_MISO, and SPI0_MOSI, are required to communicate with a SPI master. The SPISS 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.

5.9.4.15 2-Bit Mode

The SPI controller supports 2-bit Transfer mode when setting the TWOBIT bit (SPI0_CTL[16]) to 1. In 2-bit mode, the SPI controller performs full duplex data transfer. In other words, the 2-bit serial data can be transmitted and received simultaneously.

For example, in Master mode, the first data written to the TX FIFO will be transmitted through SPI0_MOSI0 and the second data written to the TX FIFO will be transmitted through the SPI0_MOSI1 pin. After transmission, the first read of RX FIFO will result in the data received from SPI0_MISO0 pin and the second read the data received from SPI0_MISO1 pin.

In Slave mode, the first two data stored in the TX FIFO will be transmitted through the SPI0_MISO0 and SPI0_MISO1 pin respectively. Concurrently, the RX FIFO will store the data received from the SPI0_MOSI0 and SPI0_MOSI1 pin, same as Master mode.

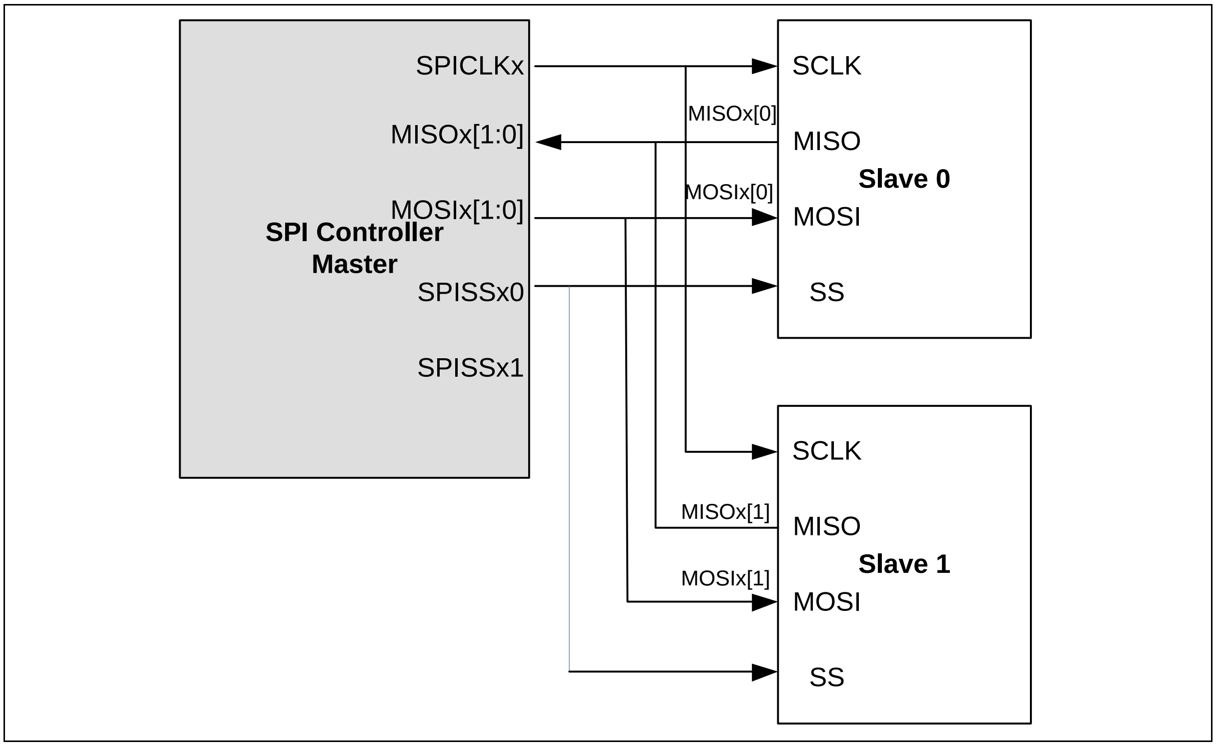


Figure 5-41 2-Bit Mode System Architecture

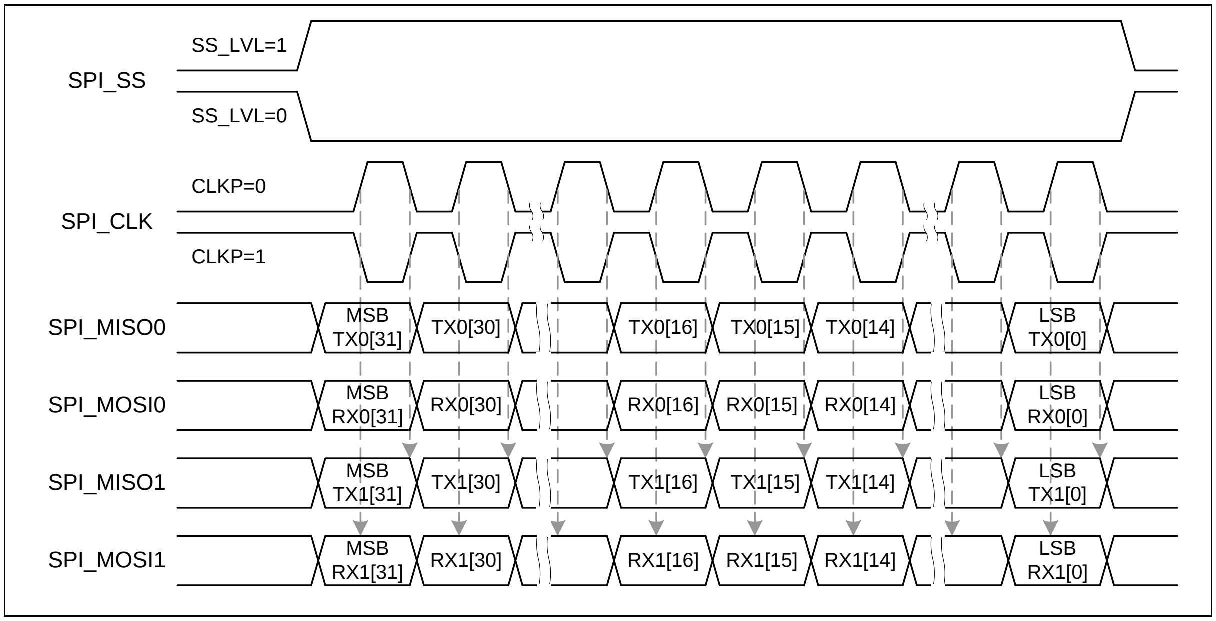


Figure 5-42 2-Bit Mode (Slave Mode)

5.9.4.16 Dual/Quad I/O Mode

The SPI controller supports dual and quad I/O transfer when setting the DUALIOEN bit or the QUADIOEN bit (SPI0_CTL[21], SPI0_CTL[22]) to 1. Many SPI Serial Flash devices support Dual/ Quad I/O transfer. The QDIODIR bit (SPI0_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 SPI0_MOSI0 is the even bit data output and the SPI0_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 SPI0_MISO0 and SPI0_MOSI0 will be set as data input ports.

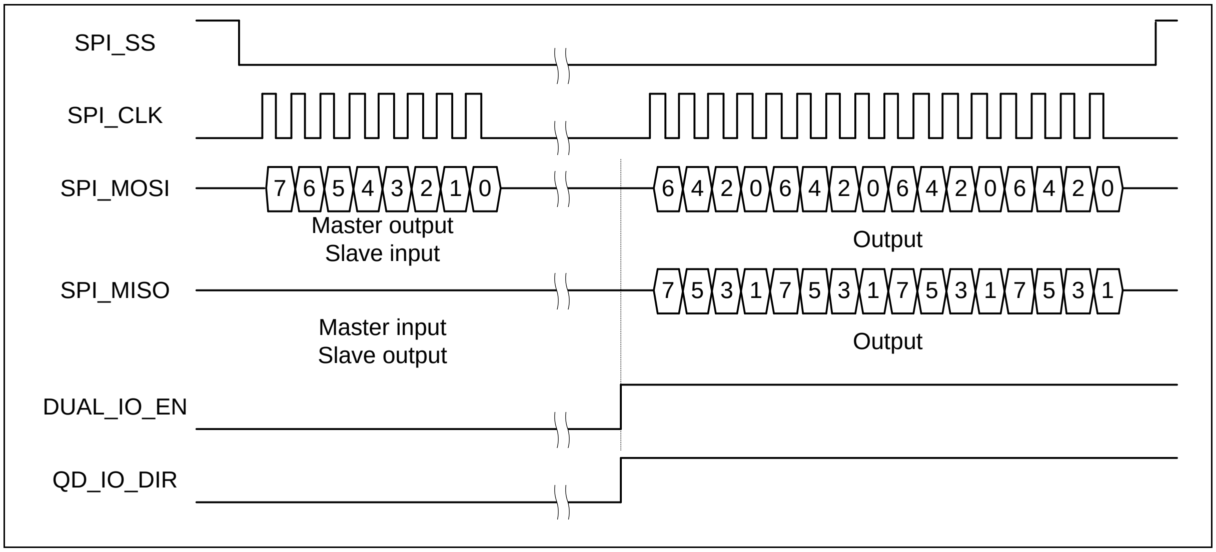


Figure 5-43 Bit Sequence of Dual Output Mode

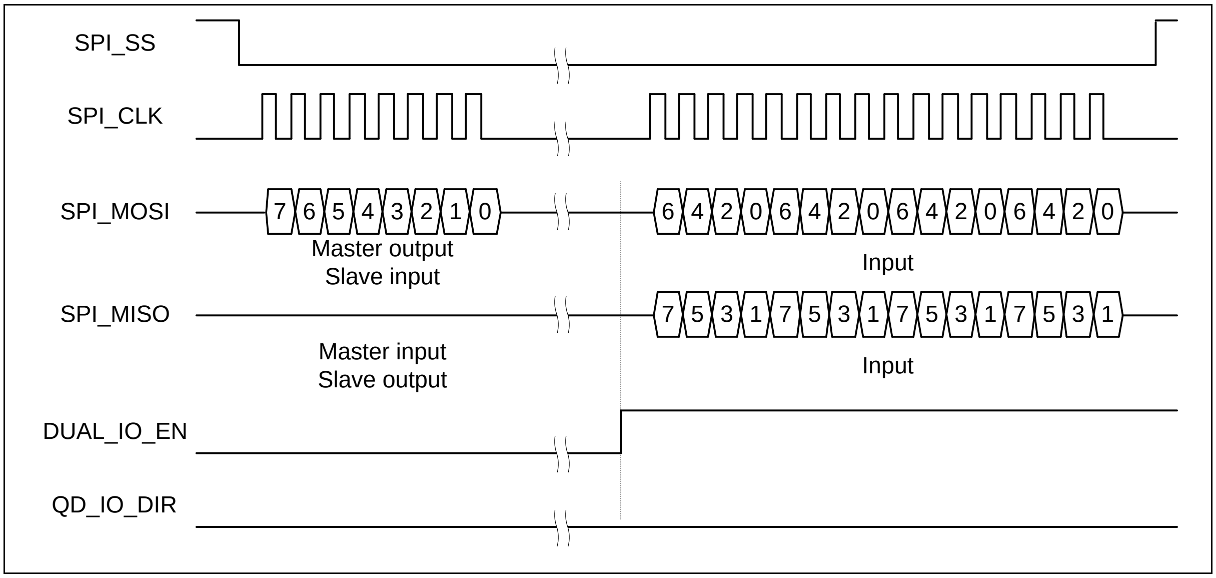


Figure 5-44 Bit Sequence of Dual Input Mode

For Quad I/O mode, if both the QUADIOEN and QDIODIR bits are set as 1, the SPI0_MOSI0 and SPI0_MOSI1 are the even bit data output and the SPI0_MISO0 and SPI0_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 SPI0_MISO0, SPI0_MISO1, SPI0_MOSI0 and SPI0_MOSI1 will be set as data input ports.

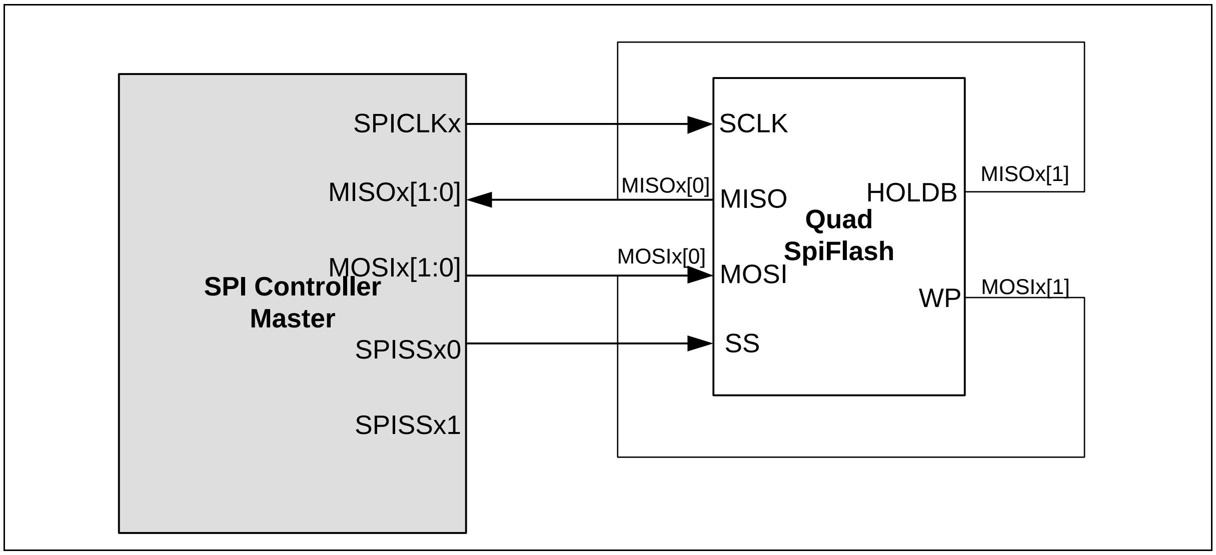


Figure 5-45 Quad Mode System Architecture

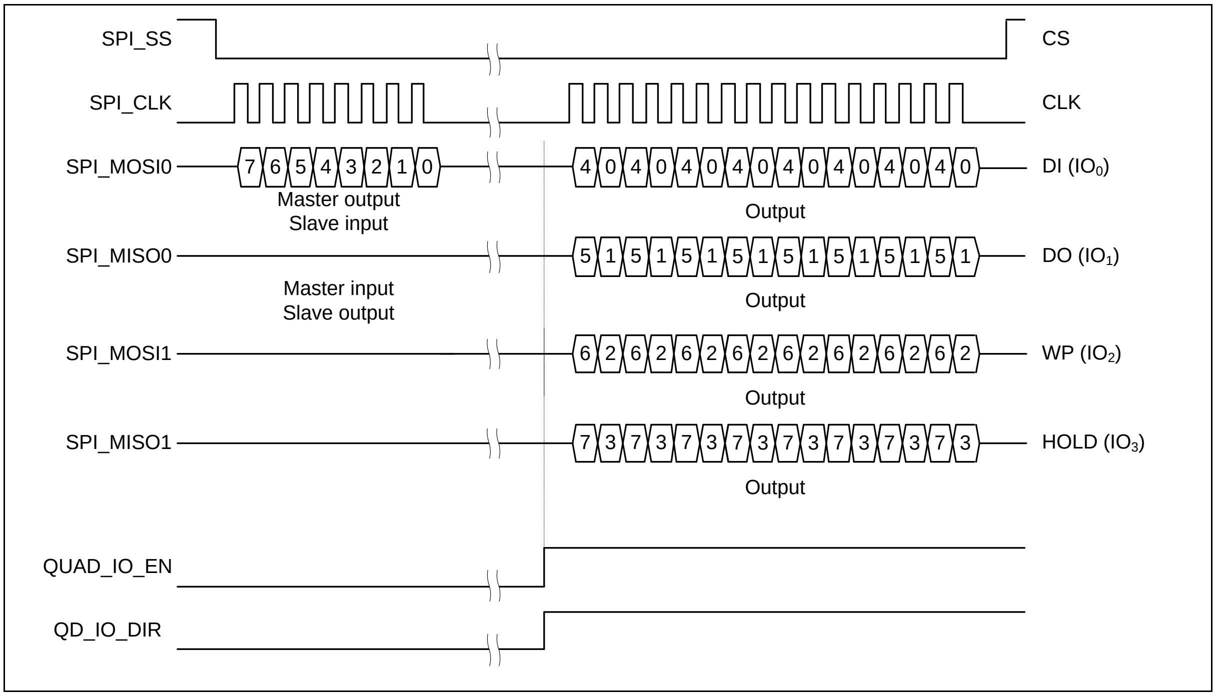


Figure 5-46 Bit Sequence of Quad Output Mode

5.9.4.17 4-Level FIFO Buffer

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

The transmit FIFO buffer is an 4-level depth, 32-bit wide, first-in, first-out register buffer. 4 words of data can be written to the transmit FIFO buffer in advance through software by writing the SPI0_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 4-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 (SPI0_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 4-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 SPI0_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, SPI0_FIFOCTL[29:28], and RXTH, SPI0_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, SPI0_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, SPI0_STATUS[10], bit will be set to 1.

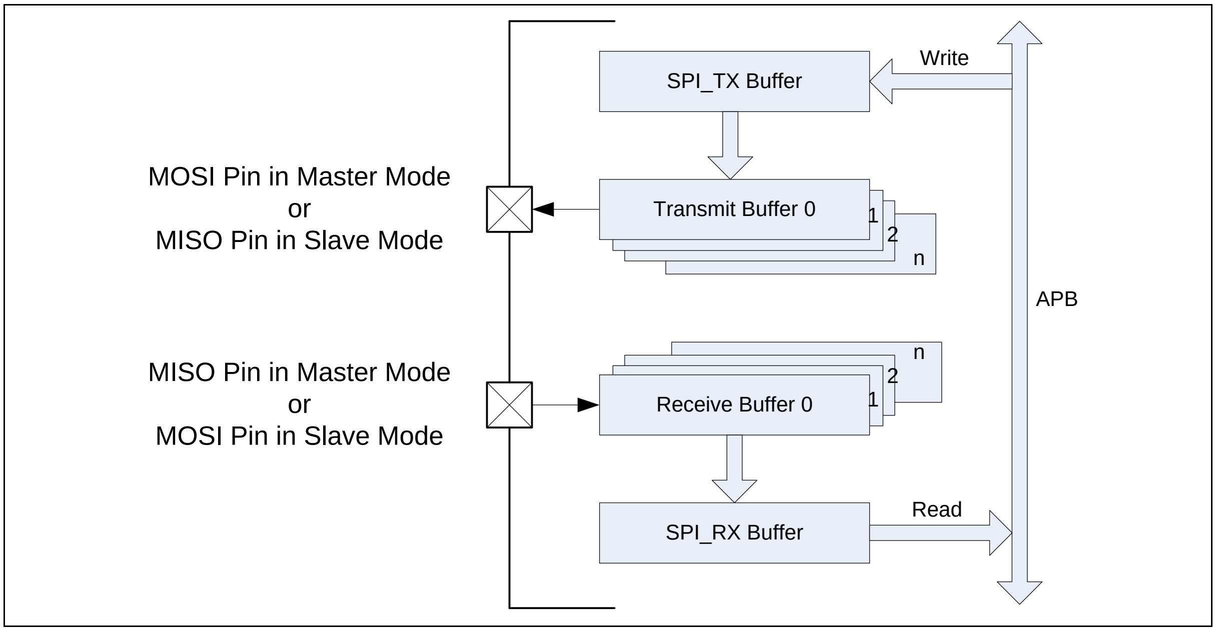


Figure 5-47 FIFO Mode Block Diagram

In Master mode, the first datum is written to the SPI0_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 SPI0_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 (SPI0_CTL [7:4]). User can write data into SPI0_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 SPI0_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 SPI0_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 SPI0_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 SPI0_RX register is read by software.

In Slave mode, during transmission operation, when data is written to the SPI0_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 SPI0_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 SPI0_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 SPI0_TX register, the under-run event, TXUFIF (SPI0_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, SPI0_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 SPI0_MOSI0/1 pin and stored to SPI0_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, SPI0_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, SPI0_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 SLVTOIF be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

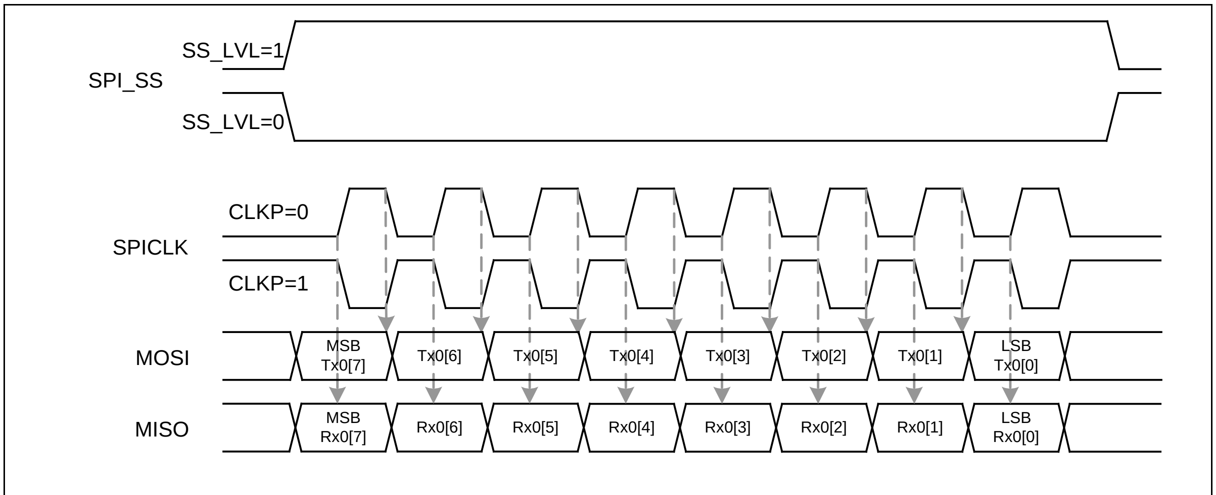
5.9.4.18 DMA Receive Mode

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 SPI0_RXTSNCNT holds the number of SPI transactions (total number of bytes is determined by SPI0_CTL.DWIDTH value).

5.9.5 SPI Timing Diagram

In master/slave mode, the device address/slave select (SPI0_SSB0/1) signal can be configured as active low or active high by the SPI0_SSCTL.SSACTPOL bit.

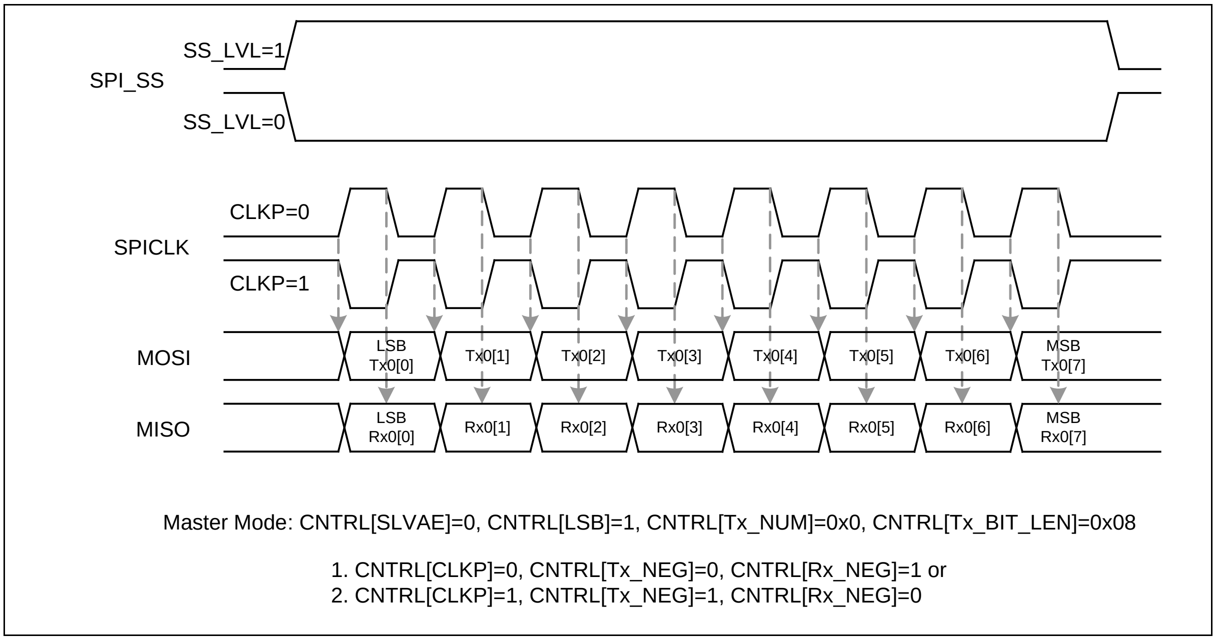
The serial clock phase and polarity is controlled by CLKPOL, RXNEG and TXNEG bits. The bit length of a transfer word is configured by the DWIDTH parameter. Whether data transmission is MSB first or LSB first is controlled by the SPI0_CTL.LSB bit. Four examples of SPI timing diagrams for master/slave operations and the related settings are shown as below.



Master Mode: CNTRL[SLVAE]=0, CNTRL[LSB]=0, CNTRL[Tx_NUM]=0x0, CNTRL[Tx_BIT_LEN]=0x08

1. CNTRL[CLKP]=0, CNTRL[Tx_NEG]=1, CNTRL[Rx_NEG]=0 or
2. CNTRL[CLKP]=1, CNTRL[Tx_NEG]=0, CNTRL[Rx_NEG]=1

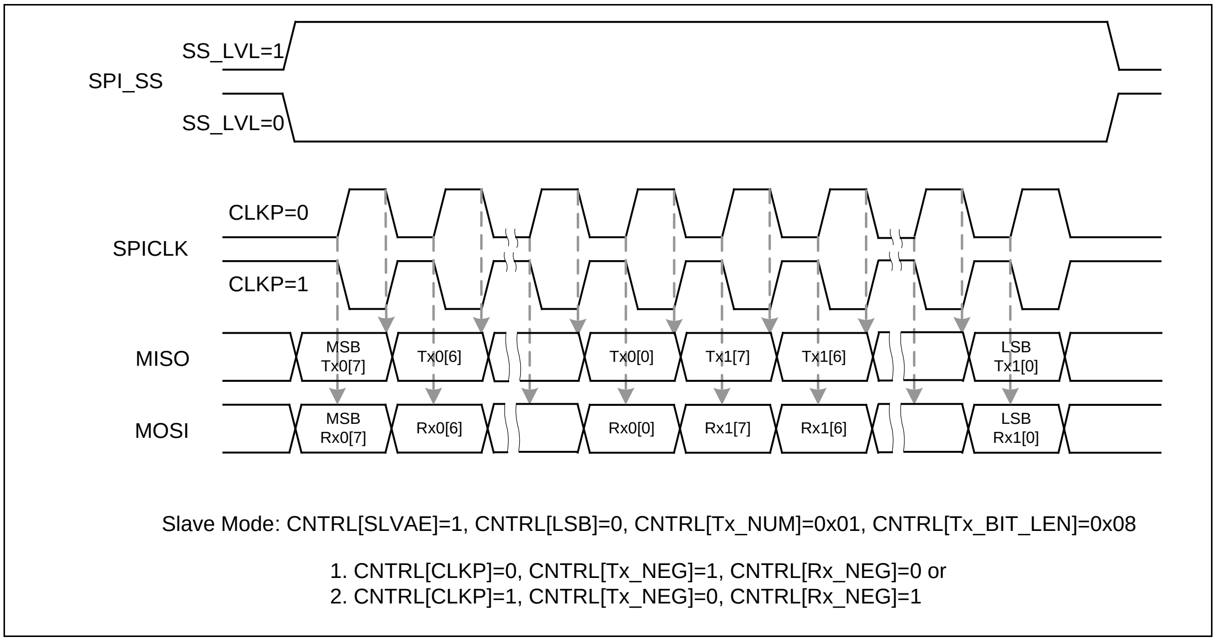
Figure 5-48 SPI Timing in Master Mode



Master Mode: CNTRL[SLVAE]=0, CNTRL[LSB]=1, CNTRL[Tx_NUM]=0x0, CNTRL[Tx_BIT_LEN]=0x08

1. CNTRL[CLKP]=0, CNTRL[Tx_NEG]=0, CNTRL[Rx_NEG]=1 or
2. CNTRL[CLKP]=1, CNTRL[Tx_NEG]=1, CNTRL[Rx_NEG]=0

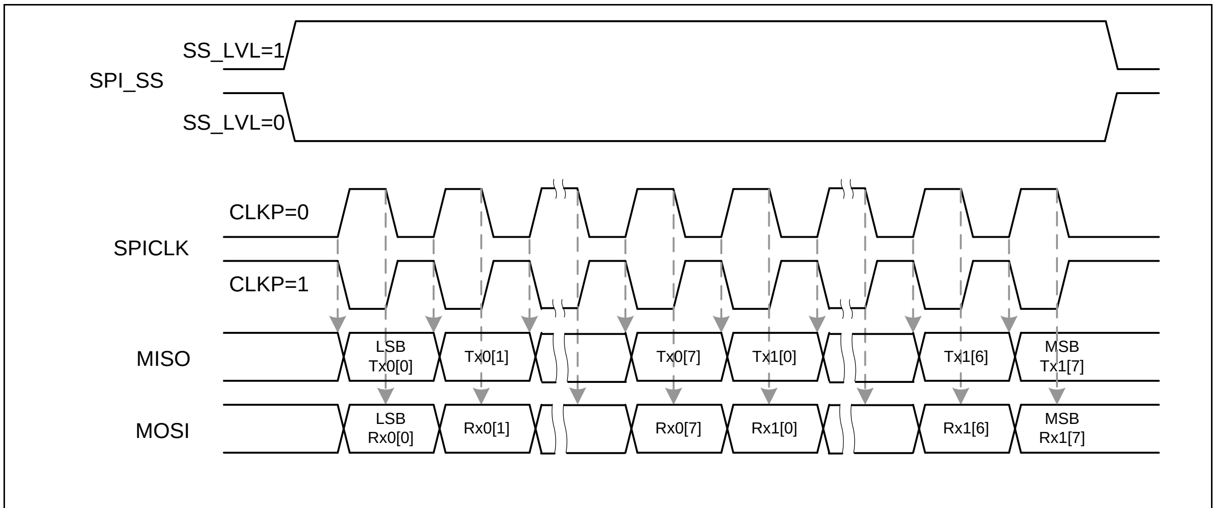
Figure 5-49 SPI Timing in Master Mode (Alternate Phase of SPICLK)



Slave Mode: CNTRL[SLVAE]=1, CNTRL[LSB]=0, CNTRL[Tx_NUM]=0x01, CNTRL[Tx_BIT_LEN]=0x08

1. CNTRL[CLKP]=0, CNTRL[Tx_NEG]=1, CNTRL[Rx_NEG]=0 or
2. CNTRL[CLKP]=1, CNTRL[Tx_NEG]=0, CNTRL[Rx_NEG]=1

Figure 5-50 SPI Timing in Slave Mode



Slave Mode: CNTRL[SLVAE]=1, CNTRL[LSB]=1, CNTRL[Tx_NUM]=0x01, CNTRL[Tx_BIT_LEN]=0x08

1. CNTRL[CLKP]=0, CNTRL[Tx_NEG]=0, CNTRL[Rx_NEG]=1 or
2. CNTRL[CLKP]=1, CNTRL[Tx_NEG]=1, CNTRL[Rx_NEG]=0

Figure 5-51 SPI Timing in Slave Mode (Alternate Phase of SPICLK)

5.9.6 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 SPI0 interface to the above specifications perform the following steps:

1) Write a divisor into the SPI0_CLKDIV register to determine the output frequency of serial clock.
2) Configure the SPI0_SSCTL register to address device. For example to manually address, set SPI0_SSCTL.ASS=0, SPI0_SSCTL.SSACTPOL =0 for active low SS. When software wishes to address device it will set SPI0_SSCTL.SS=1 to output an active SS on SPI0_SSB0 pin.
3) Configure the SPI0_CTL register. Set SPI0_CTL.SLAVE=0 for master mode, set SPI0_CTL.CLKPOL=0 for SCLK polarity normally low, set SPI0_CTL.TXNEG=1 so that data changes on falling edge of SCLK, set SPI0_CTL.RXNEG=0 so that data is latched into device on positive edge of SCLK, set SPI0_CTL.DWIDTH=4 and SPI0_CTL.TX_NUM=0 for a single byte transfer and finally set SPI0_CTL.LSB=0 for MSB first transfer.
4) If manually selecting slave device set SPI0_SSCTL.SS=1.
5) To transmit one byte of data, write data to SPI0_TX register. If only doing a receive, write a dummy byte to SPI0_TX register.
6) Enable the SPI0_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 SPI0_CTL.UNITIEN bit is set) or by polling the UNITIF bit which will be cleared to 0 by hardware automatically at end of transmission.
8) Read out the received one byte data from SPI0_RX
9) Go to 5) to continue another data transfer or set SPI0_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 SPI0_SSCTL register. SPI0_SSCTL.SSACTPOL=1 for active high slave select, SPI0_SSCTL.SS_LTRIG=1 for level sensitive trigger.

2) Configure the SPI0_CTL register. Set SPI0_CTL.SLAVE=1 for slave mode, set SPI0_CTL.CLKPOL=1 for SCLK polarity idle high, set SPI0_CTL.TXNEG=1 so that data changes on falling edge of SCLK, set SPI0_CTL.RXNEG=0 so that data is latched into device on positive edge of SCLK, set SPI0_CTL.DWIDTH=4 and SPI0_CTL.TX_NUM=0 for a single byte transfer and finally set SPI0_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 SPI0_CTL.UNITIEN bit is set) or by polling the UNITIF 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 GO_BUSY bit to stop data transfer.

5.9.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPI0_BA = 0x4003_0000**				
SPI0_CTL	SPI0_BA + 0x00	R/W	Control and Status Register	0x0000_0034
SPI0_CLKDIV	SPI0_BA + 0x04	R/W	Clock Divider Register (Master Only)	0x0000_0000
SPI0_SSCTL	SPI0_BA + 0x08	R/W	Slave Select Register	0x0000_0000
SPI0_PDMACTL	SPI0_BA + 0x0C	R/W	SPI PDMA Control Register	0x0000_0000
SPI0_FIFOCTL	SPI0_BA + 0x10	R/W	FIFO Control/Status Register	0x4400_0000
SPI0_STATUS	SPI0_BA + 0x14	R/W	Status Register	0x0005_0110
SPI0_RXTSNCNT	SPI0_BA + 0x18	R/W	Receive Transaction Count Register	0x0000_0000
SPI0_TX	SPI0_BA + 0x20	W	FIFO Data Transmit Register	0x0000_0000
SPI0_RX	SPI0_BA + 0x30	R	FIFO Data Receive Register	0x0000_0000
SPI0_VERNUM	SPI0_BA + 0x50	R	IP Version Number Register	0x0201_0001

5.9.8 Register Description

SPI Control and Status Register (SPI0_CTL)

Register	Offset	R/W	Description	Reset Value
SPI0_CTL	SPI0_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	UNITIEN	TWOBIT
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		LSB	DWIDTH				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

Bits	Description	
[31:25]	**Reserved**	Reserved.
[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.
[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 SPI0_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. For DUAL and QUAD transactions with REORDER, SUSPITV must be set to 0.
[18]	**SLAVE**	**Master Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**UNITIEN**	**Unit Transfer Interrupt Enable** 0 = Disable SPI Unit Transfer Interrupt. 1 = Enable SPI Unit Transfer Interrupt to CPU.
[16]	**TWOBIT**	**Two Bits Transfer Mode** 0 = Disable two-bit transfer mode. 1 = Enable two-bit transfer mode. When 2-bit mode is enabled, the first serial transmitted bit data is from the first FIFO buffer data, and the 2nd serial transmitted bit data is from the second FIFO buffer data. As the same as transmitted function, the first received bit data is stored into the first FIFO buffer and the 2nd received bit data is stored into the second FIFO buffer at the same time.
[15:14]	**Reserved**	Reserved.
[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]	**CLKPOL**	**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 (SPI0_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SPI0_CLKDIV	SPI0_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.
[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.

SPI Slave Select Register (SPI0_SSCTL)

Register	Offset	R/W	Description	Reset Value
SPI0_SSCTL	SPI0_BA + 0x08	R/W	Slave Select Register	0x0000_0000

31	30	29	28	27	26	25	24
SLVTOCNT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SLVTOCNT							
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	SSACTPOL	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.
[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.
[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.
[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 SPI0_CLK, SPI0_MISO, and SPI0_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 SPI0_SSCTL[1:0]. 1 = If this bit is set, SPI0_SS0/1 signals will be generated automatically. It means that device/slave select signal, which is set in SPI0_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]	**SSACTPOL**	**Slave Select Active Level** This bit defines the active status of slave select signal (SPI0_SS0/1). 0 = The slave select signal SPI0_SS0/1 is active on low-level/falling-edge. 1 = The slave select signal SPI0_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 SPI0_SS0/1 line at inactive state; writing 1 to any bit location of this field will select appropriate SPI0_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 SPI0_SS0/1 is specified in SSACTPOL. Note: SPI0_SS0 is defined as the slave select input in Slave mode.

SPI DMA Control Register (SPI0_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPI0_PDMACTL	SPI0_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.
[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 (SPI0_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
SPI0_FIFOCTL	SPI0_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	
[31:30]	**Reserved**	Reserved.
[29:28]	**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. 00: 1 word will transmit 01: 2 word will transmit 10: 3 word will transmit 11: 4 word will transmit
[27:26]	**Reserved**	Reserved.
[25:24]	**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. 00: 1 word will transmit 01: 2 word will transmit 10: 3 word will transmit 11: 4 word will transmit
[23:8]	**Reserved**	Reserved.
[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 SPI0_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 SPI0_SSCTL.SLVTORST, is enabled.

SPI Status Register (SPI0_STATUS)

Register	Offset	R/W	Description	Reset Value
SPI0_STATUS	SPI0_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.
[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.
[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 SPI0_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 SPI0_SSCTL.SSACTPOL 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 (SPI0_RXTSNCNT)

Register	Offset	R/W	Description	Reset Value
SPI0_RXTSNCNT	SPI0_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.
[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 SPI0_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 (SPI0_TX)

Register	Offset	R/W	Description	Reset Value
SPI0_TX	SPI0_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 SPI0_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 (SPI0_RX)

Register	Offset	R/W	Description	Reset Value
SPI0_RX	SPI0_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 SPI0_STATUS. RXEMPTY bit is not set to 1. This is a read-only register.

5.10 Serial 1 Peripheral Interface (SPI1) Controller

5.10.1 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 ISD91200 contains an 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 SPI1 controller can be set as a master; also can be set as a slave controlled by an off-chip master device.

5.10.2 Features

- Supports master or slave mode operation.
- Configurable word length of up to 32 bits. Up to two words can be transmitted per a transaction, giving a maximum of 64 bits for each data transaction.
- Provide burst mode operation.
- MSB or LSB first transfer.
- 2 device/slave select lines in master mode, single device/slave select line in slave mode.
- Byte or word Sleep Suspend Mode .
- Support dual FIFO mode.
- PDMA access support.

5.10.3 SPI1 Block Diagram

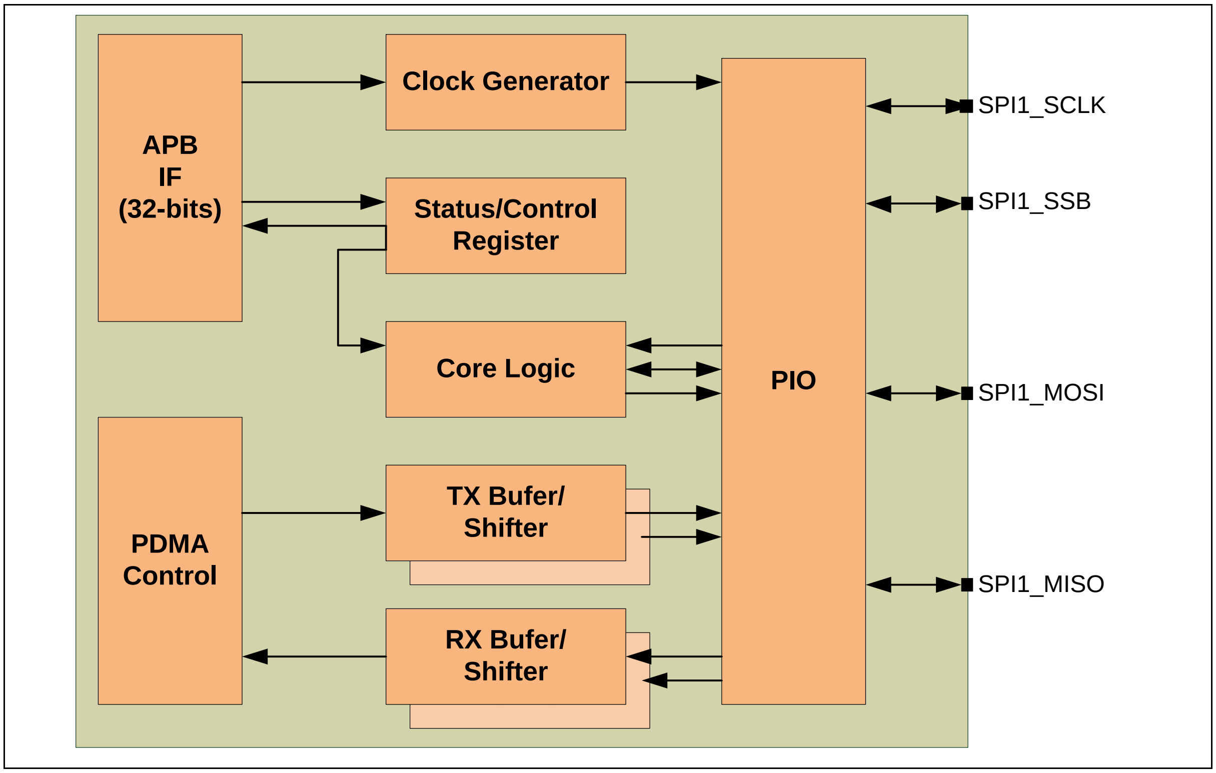


Figure 5-52 SPI1 Block Diagram

5.10.4 SPI1 Function Descriptions

5.10.4.1 SPI Engine Clock and SPI Serial Clock

The SPI controller derives its clock source from the system HCLK as determined by the CLKSEL1 register. The frequency of the SPI master clock is determined by the divisor ratio SPI1_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.

5.10.4.2 Master/Slave Mode

This SPI1 controller can be configured as in master or slave mode by setting the SLAVE bit (SPI1_CTL.SLAVE). In master mode the ISD91200 generates SCLK and SSB signals to access one or more slave devices. In slave mode the ISD91200 monitors SCLK and SSB signals to respond to data transactions from an off-chip master. The signal directions are summarized in the below application block diagrams.

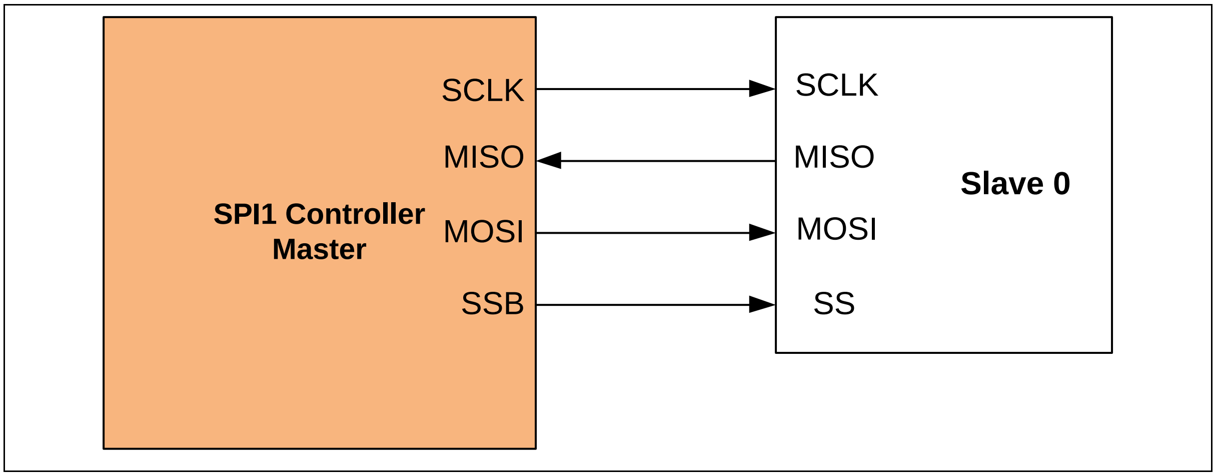


Figure 5-53 SPI1 Master Mode Application Block Diagram

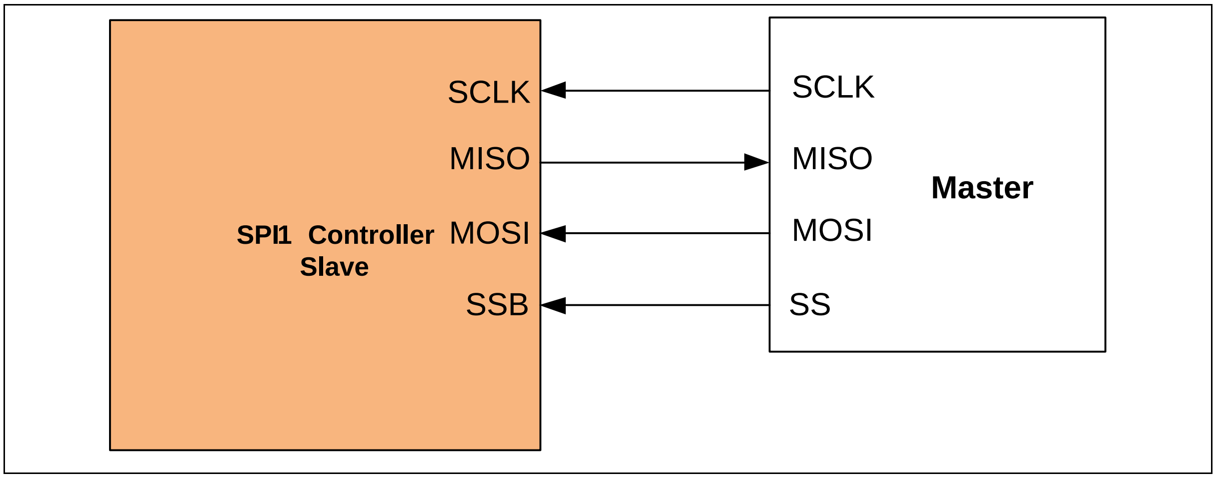


Figure 5-54 SPI1 Slave Mode Application Block Diagram

5.10.4.3 Slave Select

In master mode, the SPI controller can address up to one off-chip slave devices through the slave select output pins SPI1_SSB. 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_SSB to address the SPI1 controller. The slave select signal can be programmed to be active low or active high via the SPI1_SSCTL.SSLVL bit. In addition the SPI1_SSCTL.SSLTRIG 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.

5.10.4.4 Automatic Slave Select

In master mode, if the bit SPI1_SSCTL.AUTOSS is set, the slave select signals will be generated automatically and output to SPI1_SSB pins according to registers SPI1_SSCTL.SS. In this mode, SPI controller will assert SSB when transaction is triggered and de-assert when data transfer is finished. If the SPI1_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 SPI1_SSCTL.SS register. The active level of the slave select output signals is specified by the SPI1_SSCTL.SSLVL bit.

5.10.4.5 Serial Clock

In master mode, writing a divisor into the SPI1_CLKDIV register will program the output frequency of serial clock to the SPI1_SCLK output port. In slave mode, the off-chip master device drives the serial clock through the SPI1_SCLK.

5.10.4.6 Clock Polarity

The SPI1_CTL.CLKP bit defines the serial clock idle state in master mode. If CLKP = 1, the output SPI1_SCLK is high in idle state. If CLKP=0, it is low in idle state.

5.10.4.7 Transmit/Receive Bit Length

The bit length of a transfer word is defined in SPI1_CTL.TXBITLEN bit field. It is set to define the length of a transfer word and can be up to 32 bits in length. TXBITLEN=0x0 enables 32bit word length.

5.10.4.8 Burst Mode

The SPI controller has a burst mode controlled by the SPI1_CTL.TXNUM field. If set to 0x01, SPI controller will burst two transactions from the SPI1_TX0 and SPI1_TX1 registers as shown in the waveform below:

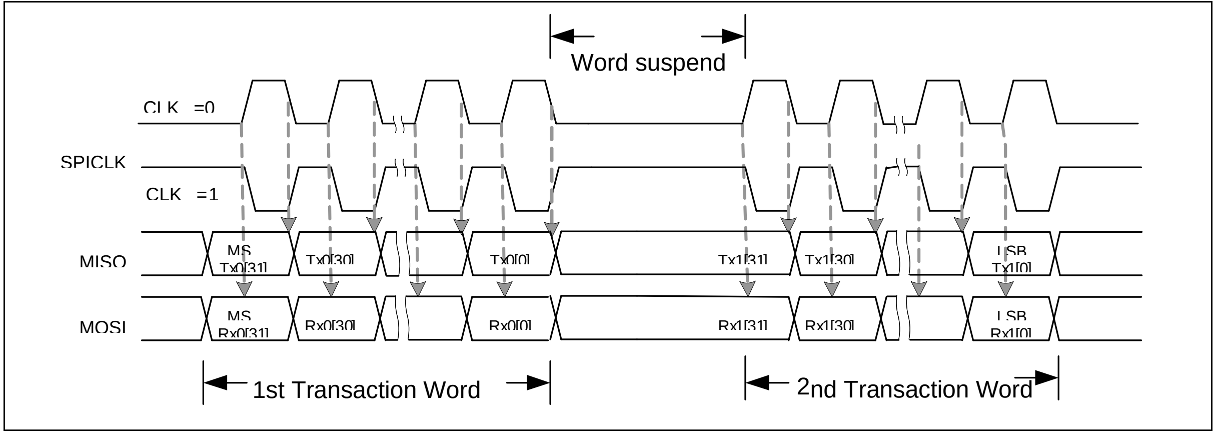


Figure 5-55 Two Transactions in One Transfer (Burst Mode)

5.10.4.9 LSB First

The SPI1_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.

5.10.4.10 Transmit Edge

The SPI1_CTL.TXNEG bit determines whether transmit data is changed on the positive or negative edge of the SPI1_SCLK serial clock. If TXNEG=0 then transmitted data will change state on the rising edge of SPI1_SCLK. If TXNEG=1 then transmitted data will change state on the falling edge of SPI1_SCLK.

5.10.4.11 Receive Edge

The SPI_CTL.RXNEG bit determines whether data is received at either the negative edge or positive edge of serial clock SPI1_SCLK. If RXNEG=1 then data is clocked in on the falling edge of SPI1_SCLK. If RXNEG=0 data is clocked in on the rising edge of SPI1_SCLK. Note that RXNEG should be the inverse of TXNEG for standard SPI operation.

5.10.4.12 Word Sleep Suspend

The bit field SPI1_CTL.SLEEP provides a configurable suspend interval of SLEEP+2 serial clock periods between successive word transfers in master mode. The suspend interval is from the last falling clock edge of the preceding transfer word to the first rising clock edge of the following transfer word if CLKP = 0. If CLKP = 1, the interval is from the rising clock edge of the preceding transfer word to the falling clock edge of the following transfer word. The default value of SLEEP is 0x0 (2 serial clock cycles). Word Sleep only occurs when TXNUM=1. For TXNUM=0, this parameter will determine a Byte Sleep condition if the BYTESLEEP bit is set.

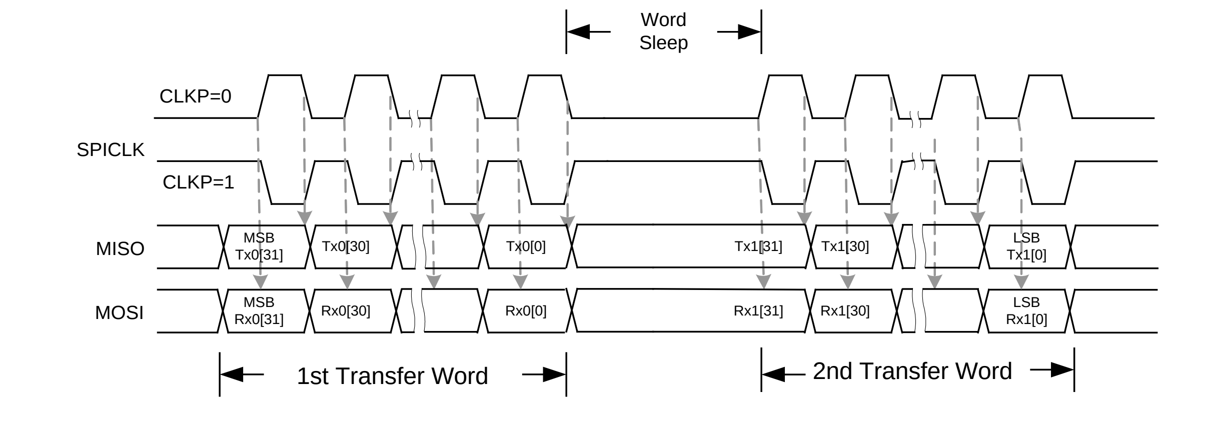


Figure 5-56 Word Sleep Suspend Mode

5.10.4.13 Byte Endian

APB access to the SPI controller is via the 32bit wide TX and RX registers. When the transfer is set as MSB first (SP1I_CTL.LSB = 0) and the SPI1_CTL.BYTEENDIAN 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 TXBITLEN =0 (32 bits transfer), the sequence of transmitted/received data will be BYTE0, BYTE1, BYTE2, and then BYTE3. If TXBITLEN is set to 24-bits, the sequence will be BYTE0, BYTE1, and BYTE2. The rule of 16-bits mode is the same as above.

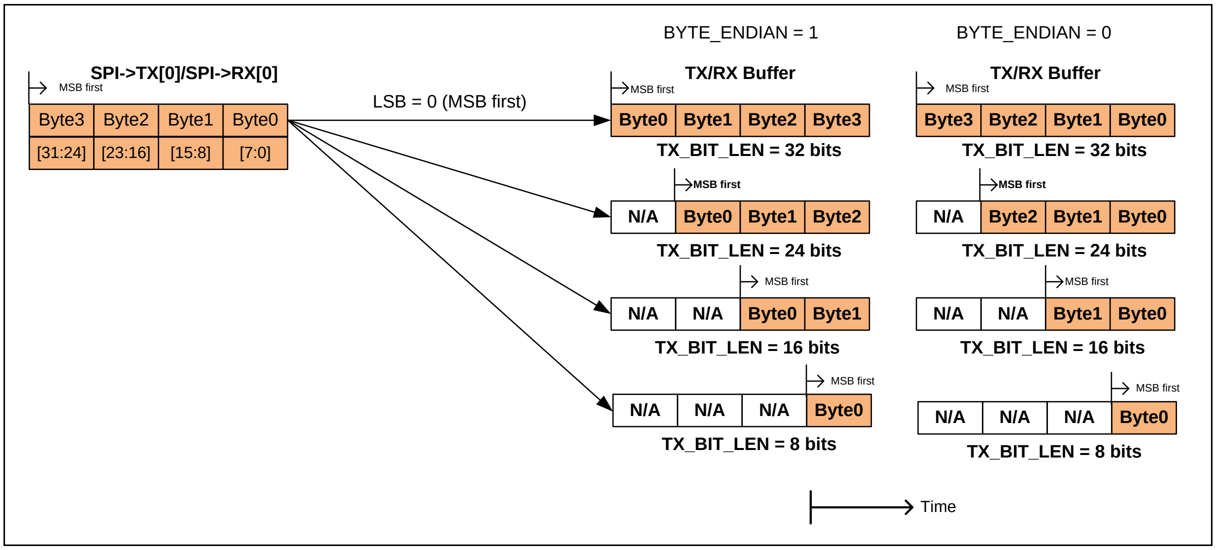
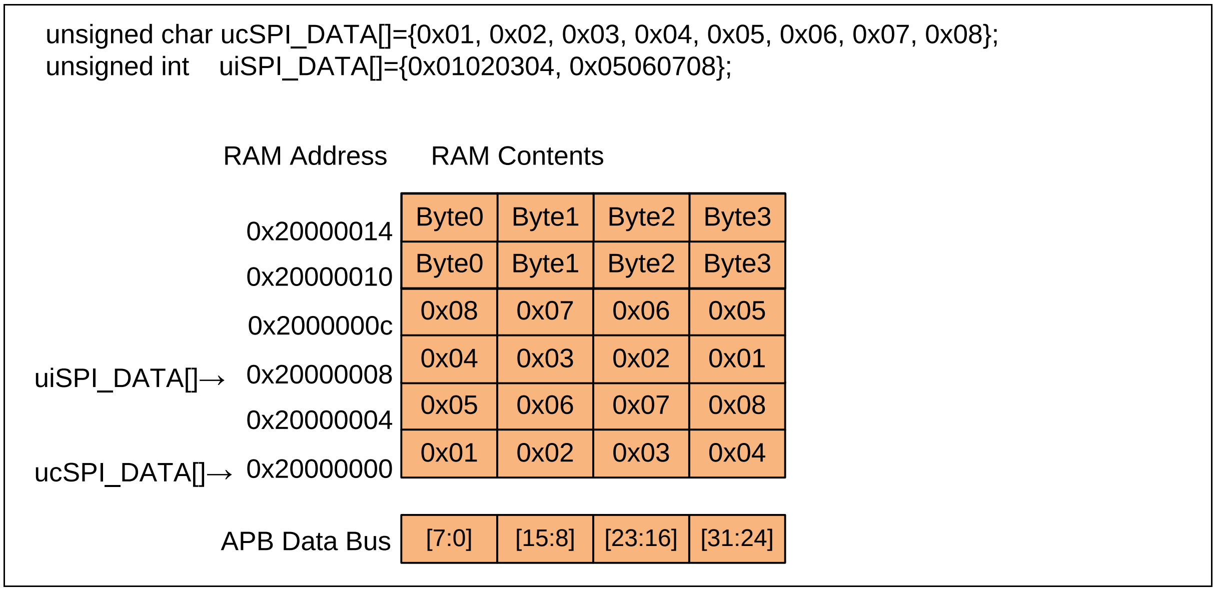


Figure 5-57 Byte Re-Ordering Transfer

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};



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

	RAM Address	RAM Contents			
	0x20000014	Byte0	Byte1	Byte2	Byte3
	0x20000010	Byte0	Byte1	Byte2	Byte3
	0x2000000c	0x08	0x07	0x06	0x05
uiSPI_DATA[]→	0x20000008	0x04	0x03	0x02	0x01
	0x20000004	0x05	0x06	0x07	0x08
ucSPI_DATA[]→	0x20000000	0x01	0x02	0x03	0x04
	APB Data Bus	[7:0]	[15:8]	[23:16]	[31:24]

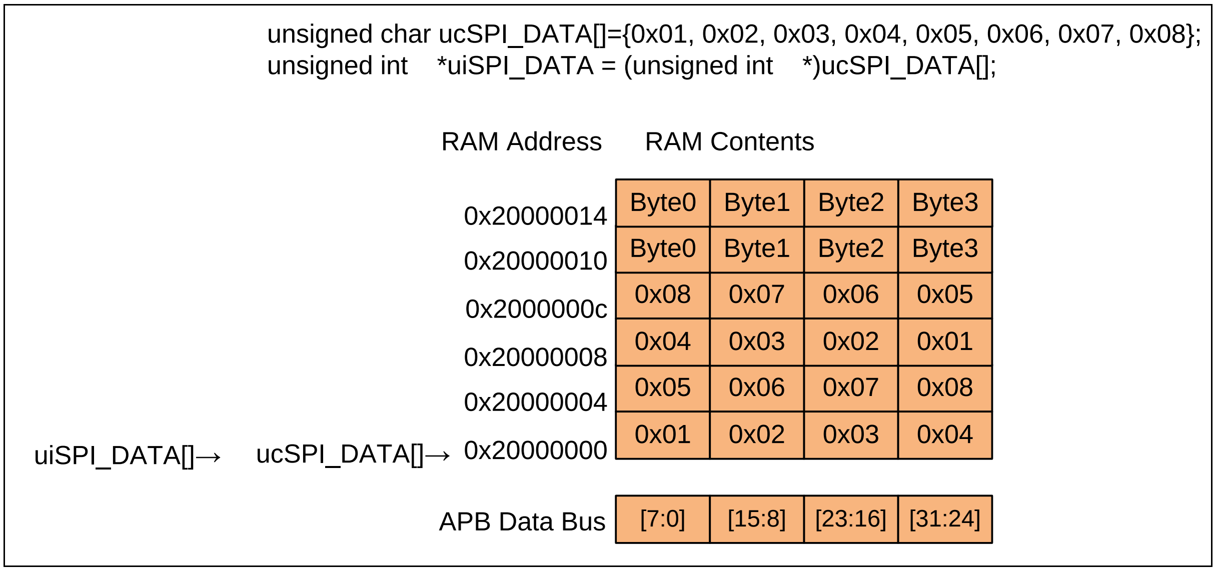
Figure 5-58 Byte Order in Memory

It can be seen from Figure 5-58 Byte Order in Memory 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 TXBITLEN=8 and send data byte-by-byte SPI1_TX[0] = ucSPI_DATA[i++]
2. Set TXBITLEN=32 and send word-by-word SPI1_TX[0] = 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 below figure where a int pointer points to the byte data array.



```
unsigned char ucSPI_DATA[]={0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08};
unsigned int    *uiSPI_DATA = (unsigned int    *)ucSPI_DATA[];
```

	RAM Address	RAM Contents			
	0x20000014	Byte0	Byte1	Byte2	Byte3
	0x20000010	Byte0	Byte1	Byte2	Byte3
	0x2000000c	0x08	0x07	0x06	0x05
	0x20000008	0x04	0x03	0x02	0x01
	0x20000004	0x05	0x06	0x07	0x08
uiSPI_DATA[]→ ucSPI_DATA[]→	0x20000000	0x01	0x02	0x03	0x04
	APB Data Bus	[7:0]	[15:8]	[23:16]	[31:24]

Figure 5-59 Byte Order in Memory

Now if we set TXBITLEN=32 and sent word-by-word SPI1_TX0 = uiSPI_DATA[i++], the order transmitted would be {0x04, 0x03, 0x02, 0x01, 0x08, 0x07, 0x06, 0x05}. However if we set BYTEENDIAN=1, we would reverse this order to the desired stream: {0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08}.

5.10.4.14 Byte Sleep Suspend

In master mode, if SPI1_CTL.BYTESLEEP is set to 1, the hardware will insert a suspend interval of SPI1_CTL.SLEEP+2 serial clock periods between two successive bytes in a transfer word. Note that the byte suspend function is only valid for 32bit word transfers, that is TXBITLEN=0x00.

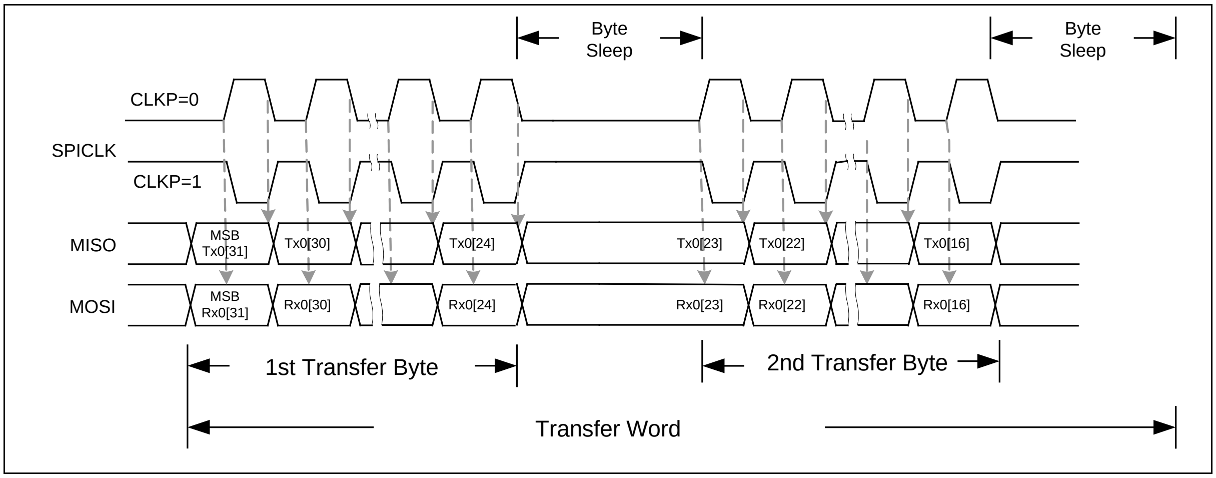


Figure 5-60 Byte Suspend Mode

5.10.4.15 Interrupt

The SPI controller can generate a CPU interrupt when data transfer is finished. When a transfer request triggered by EN is finished, the interrupt flag (SPI_CTL.IF) will be set by hardware. If the SPI interrupt is enabled (SPI1_CTL.IE) this will also generate a CPU interrupt. To clear the interrupt event flag, software must write a ‘1’ to it.

5.10.4.16 FIFO Mode

The SPI controller supports a dual buffer mode when SPI_CTL.FIFO is set as 1. In normal mode, software can only update the transmitted data when the current transmission is done. In FIFO mode, the next transmitted data can be written into the SPI_TX buffer at any time when in master mode or the EN bit is set in slave mode. This data will load into the transmit buffer when the current transmission done.

After the FIFO bit is set, transmission is repeated automatically when the transmitted data is updated in time and it will continue until this bit is cleared. When cleared, the transmission will finish when the current transmission done. The user can also read the received data at any time before the next transmission is complete, wherein the receive buffer will be updated with new received data. If transmit data isn’t updated before the current transmission is done, the transaction will stop. The transmission will resume automatically when transmit data is written into this buffer again.

Before the FIFO bit is set, the user can write first data into SPI1_TX buffer. Setting FIFO active will load the first data into the current transmission buffer. A subsequent write to SPI_TX will load the TX FIFO which will be loaded into the transmission buffer after the 1st transmission is done.

This function is also supported in slave mode. The EN must be set as 1 before the external serial clock input and it will keep going until the FIFO is cleared.

The delay period between two transmissions is programmable. It is the same as the suspend interval on SLEEP parameter.

5.10.4.17 Two Channel Mode

The SPI controller supports a two channel mode where data can be sent and received on alternate MOSI1 and MISO1 lines concurrently with data on MOSI0 and MISO0. The data for this second channel is the SPI1_RX1 and SPI1_TX1 buffers. Mode is enabled by setting the SPI_CTL.TWOB bit. This mode is only available when TXNUM=0.

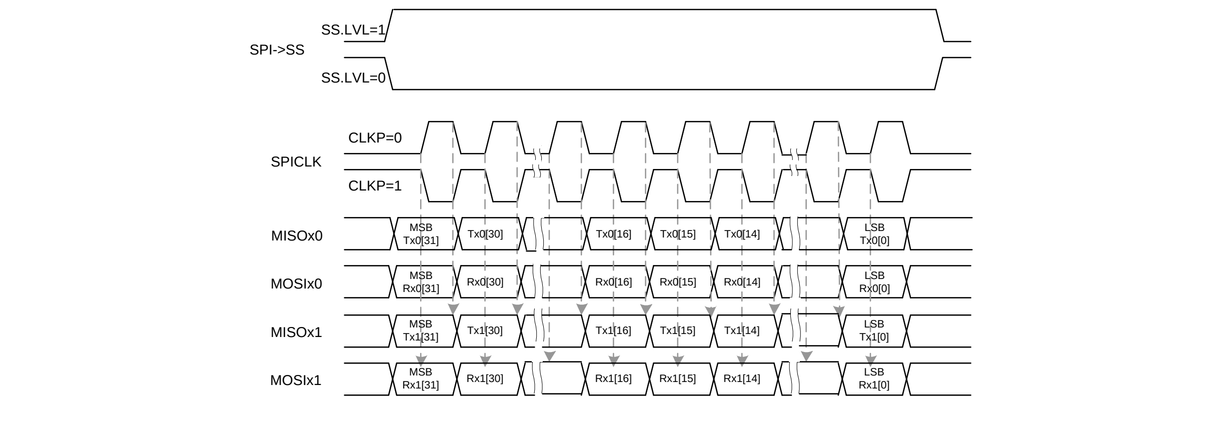


Figure 5-61 Two Bits Transfer Mode

5.10.4.18 Variable Serial Clock Frequency

In master mode 16 bit transfers, the output of serial clock can be programmed as variable frequency pattern if the Variable Clock Enable bit SPI1_CTL.VARCLKEN is enabled. The frequency pattern format is defined in SP1I_VARCLK register. If the bit content of VARCLK is ‘0’ the output period for that bit is determined by setting of SPI1_CLKDIV, if the bit content of VARCLK is ‘1’, the output period for that bit is determined by the SPI1_ CLKDIV.DIV1 register. The following figure shows the timing relationships of serial clock (SCLK), to the VARCLK, the DIV0 and the DIV1 registers. A two-bit combination in the VARCLK defines one clock cycle. The bit field VARCLK[31:30] defines the first clock cycle of SCLK. The bit field VARCLK[29:28] defines the second clock cycle of SCLK and so on. The clock source selections are defined in VARCLK and must be set 1 cycle before the next clock option. For example, if there are 5 CLK1 cycle in SPICLK, the VARCLK shall set 9 ‘0’ in the MSB of VARCLK. The 10th shall be set as ‘1’ in order to switch the next clock source is CLK2. Note that when VARCLKEN bit is set, the setting of TXBITLEN must be programmed as 0x10 (16 bits mode only).

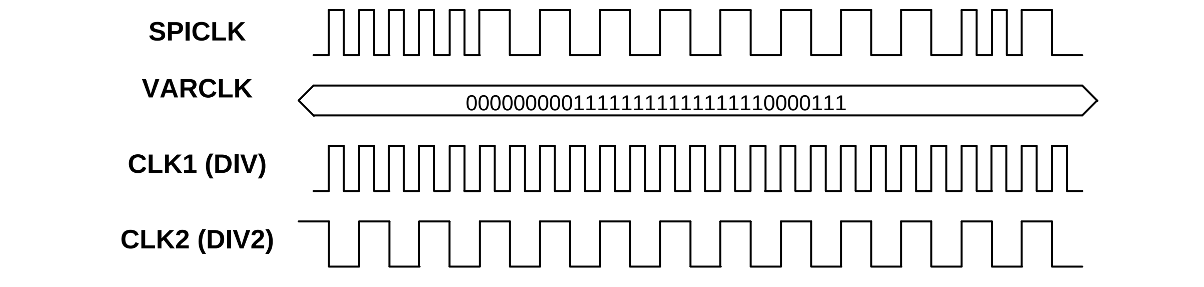
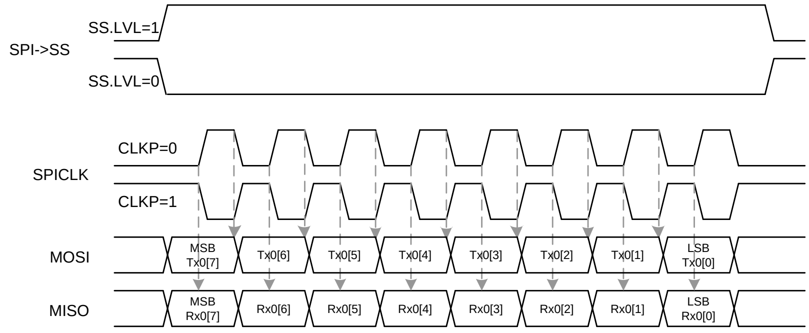


Figure 5-62 Variable Serial Clock Frequency

5.10.5 SPI Timing Diagram

In master/slave mode, the device address/slave select (SPI_SSB) signal can be configured as active low or active high by the SPI1_ SSCTL.SSLVL bit. In slave mode, the SPI1>- SSCTL.SSLTRIG will determine whether the slave select signal is treated as a level triggered or edge triggered signal.

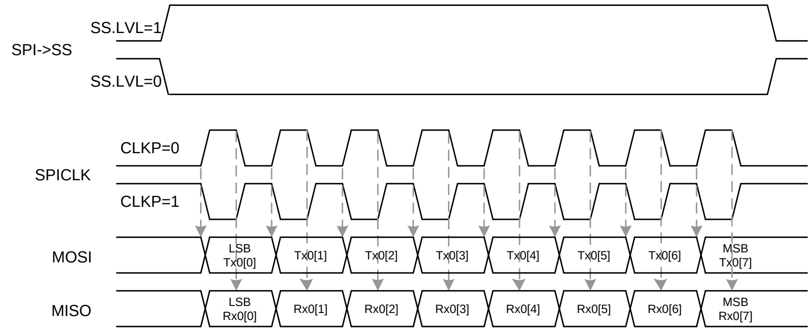
The serial clock phase and polarity is controlled by CLKP, RXNEG and TXNEG bits. The bit length of a transfer word is configured by the TXBITLEN parameter. Whether data transmission is MSB first or LSB first is controlled by the SPI1_CTL.LSB bit. Four examples of SPI timing diagrams for master/slave operations and the related settings are shown as below.



Master Mode: CTL.SLVAE=0, CTL.LSB=0, CTL.TXNUM=0x0, CTL.TXBITLEN=0x08

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

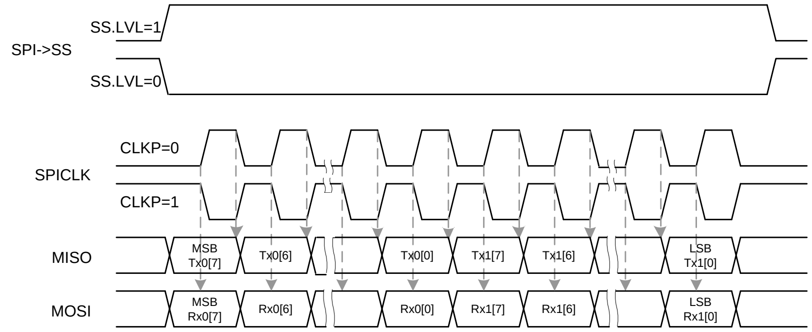
Figure 5-63 SPI Timing in Master Mode



Master Mode: CTL.SLVAE=0, CTL.LSB=1, CTL.TXNUM=0x0, CTL.TXBITLEN=0x08

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

Figure 5-64 SPI Timing in Master Mode (Alternate Phase of SPICLK)



Slave Mode: CTL.SLVAE=1, CTL.LSB=0, CTL.TXNUM=0x01, CTL.TXBITLEN=0x08

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

Figure 5-65 SPI Timing in Slave Mode

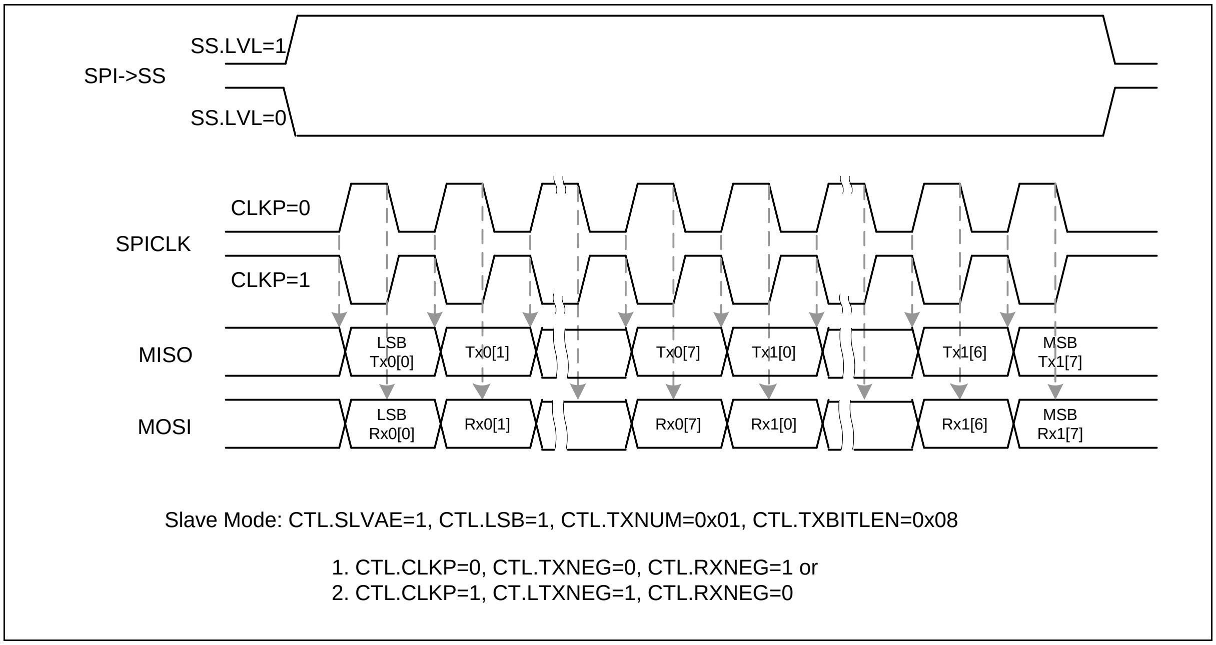

SPI->SS
SS.LVL=1
SS.LVL=0
SPICLK
CLKP=0
CLKP=1
MISO
LSB
Tx0[0]
Tx0[1]
Tx0[7]
Tx1[0]
Tx1[6]
MSB
Tx1[7]
MOSI
LSB
Rx0[0]
Rx0[1]
Rx0[7]
Rx1[0]
Rx1[6]
MSB
Rx1[7]
Slave Mode: CTL.SLVAE=1, CTL.LSB=1, CTL.TXNUM=0x01, CTL.TXBITLEN=0x08
1. CTL.CLKP=0, CTL.TXNEG=0, CTL.RXNEG=1 or
2. CTL.CLKP=1, CT.LTXNEG=1, CTL.RXNEG=0

Figure 5-66 SPI Timing in Slave Mode (Alternate Phase of SPICLK)

5.10.6 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:

10) Write a divisor into the SPI1_CLKDIV register to determine the output frequency of serial clock.

11) Configure the SPI1_ SSCTL register to address device. For example to manually address, set SPI1_SSCTL.ASS=0, SPI1_SSCTL.SSLVL=0 for active low SS. When software wishes to address device it will set SPI1_ SSCTL.SSR=1 to output an active SS on SPI1_SSB pin.

12) Configure the SPI1_CTL register. Set SPI1_CTL.SLAVE=0 for master mode, set SPI1_CTL.CLKP=0 for SCLK polarity normally low, set SPI1_CTL.TXNEG=1 so that data changes on falling edge of SCLK, set SPI1_CTL.RXNEG=0 so that data is latched into device on positive edge of SCLK, set SPI1_CTL.TXBITLEN=8 and SPI_CNTRL.TXNUM=0 for a single byte transfer and finally set SPI1_CTL.LSB=0 for MSB first transfer.

13) If manually selecting slave device set SPI1_ SSCTL.SSR=1.

14) To transmit one byte of data, write data to SPI1_TX0 register. If only doing a receive, write a dummy byte to SPI1_TX0 register.

15) Enable the SPI1_CTL.EN bit to start the data transfer over the SPI interface.

-- Wait for SPI transfer to finish. Can be interrupt driven (if the interrupt enable SPI_CTL.IE bit is set) or by polling the EN bit which will be cleared to 0 by hardware automatically at end of transmission.

16) Read out the received one byte data from SPI1_ RX0

17) Go to 5) to continue another data transfer or set SPI1_ SSCTL.SSR=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:

8) Configure the SPI1_ SSCTL register. SPI1_ SSCTL.SSLVL=1 for active high slave select, SPI1_ SSCTL.SSR.SSLTRIG=1 for level sensitive trigger.

9) Configure the SPI1_CTL register. Set SPI1_CTL.SLAVE=1 for slave mode, set SPI1_CTL.CLKP=1 for SCLK polarity idle high, set SPI1_CTL.TXNEG=1 so that data changes on falling edge of SCLK, set SPI1_CTL.RXNEG=0 so that data is latched into device on positive edge of SCLK, set SPI1_CTL.TXBITLEN=8 and SPI1_CTL.TXNUM=0 for a single byte transfer and finally set SPI1_CTL.LSB=1 for LSB first transfer.

10) If SPI slave is to transmit one byte of data to the off-chip master device, write first byte to TX[0] register. If no data to be transmitted write a dummy byte.

11) Enable the EN 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.

-- -- Wait for SPI transfer to finish. Can be interrupt driven (if the interrupt enable SPI1_CTL.IE bit is set) or by polling the EN bit which will be cleared to 0 by hardware automatically at end of transmission. --

12) Read out the received data from SPI1_RX0 register.

13) Go to 3) to continue another data transfer or disable the EN bit to stop data transfer.

5.10.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI1 Base Address: **SPI1_BA = 0x4003_8000**				
SPI1_CTL	SPI1_BA + 0x00	R/W	Control and Status Register	0x0500_0004
SPI1_CLKDIV	SPI1_BA + 0x04	R/W	Clock Divider Register (Master Only)	0x0000_0000
SPI1_SSCTL	SPI1_BA + 0x08	R/W	Slave Select Register	0x0000_0000
SPI1_RX0	SPI1_BA + 0x10	R	Data Receive Register 0	0x0000_0000
SPI1_RX1	SPI1_BA + 0x14	R	Data Receive Register 1	0x0000_0000
SPI1_TX0	SPI1_BA + 0x20	W	Data Transmit Register 0	0x0000_0000
SPI1_TX1	SPI1_BA + 0x24	W	Data Transmit Register 1	0x0000_0000
SPI1_VARCLK	SPI1_BA + 0x34	R/W	Variable Clock Pattern Register	0x007F_FF87
SPI1_PDMACTL	SPI1_BA + 0x38	R/W	SPI PDMA Control Register	0x0000_0000

5.10.8 Register Description

SPI Control and Status Register (SPI1_CTL)

Register	Offset	R/W	Description	Reset Value
SPI1_CTL	SPI1_BA + 0x00	R/W	Control and Status Register	0x0500_0004

31	30	29	28	27	26	25	24
Reserved			DMABURST	TXFULL	TXEMPTY	RXFULL	RXEMPTY
23	22	21	20	19	18	17	16
VARCLKEN	TWOB	FIFO	BYTEENDIAN	BYTESLEEP	SLAVE	IE	IF
15	14	13	12	11	10	9	8
SLEEP				CLKP	LSB	TXNUM	
7	6	5	4	3	2	1	0
TXBITLEN					TXNEG	RXNEG	EN

Bits	Description	
[31:28]	**Reserved**	Reserved
[28]	**DMABURST**	**Enable DMA Automatic SS function.** When enabled, interface will automatically generate a SS signal for an entire PDMA access transaction.
[27]	**TXFULL**	**Transmit FIFO FULL STATUS** 1 = The transmit data FIFO is full. 0 = The transmit data FIFO is not full.
[26]	**TXEMPTY**	**Transmit FIFO EMPTY STATUS** 1 = The transmit data FIFO is empty. 0 = The transmit data FIFO is not empty.
[25]	**RXFULL**	**Receive FIFO FULL STATUS** 1 = The receive data FIFO is full. 0 = The receive data FIFO is not full.
[24]	**RXEMPTY**	**Receive FIFO EMPTY STATUS** 1 = The receive data FIFO is empty. 0 = The receive data FIFO is not empty.
[23]	**VARCLKEN**	**Variable Clock Enable (Master Only)** 1 = SCLK output frequency is variable. The output frequency is determined by the value of VARCLK, DIVIDER, and DIVIDER2. 0 = The serial clock output frequency is fixed and determined only by the value of DIVIDER. Note that when enabled, the setting of TXBITLEN must be programmed as 0x10 (16 bits mode)

[22]	**TWOB**	**Two Bits Transfer Mode** 1 = Enable two-bit transfer mode. 0 = Disable two-bit transfer mode. Note that when enabled in master mode, MOSI data comes from SPI1_TX0 and MOSI data from SPI1_TX1. Likewise SPI1_RX0 receives bit stream from MISO0 and SPI1_RX1 from MISO1. Note that when enabled, the setting of TXNUM must be programmed as 0x00
[21]	**FIFO**	**FIFO Mode** 0 = No FIFO present on transmit and receive buffer. 1 = Enable FIFO on transmit and receive buffer.
[20]	**BYTEENDIAN**	**Byte Endian Reorder Function** This function changes the order of bytes sent/received to be least significant physical byte first.
[19]	**BYTESLEEP**	**Insert Sleep interval between Bytes** This function is only valid for 32bit transfers (TXBITLEN=0). If set then a pause of (SLEEP+2) SCLK cycles is inserted between each byte transmitted.
[18]	**SLAVE**	**Master Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**IE**	**Interrupt Enable** 0 = Disable SPI Interrupt. 1 = Enable SPI Interrupt to CPU.
[16]	**IF**	**Interrupt Flag** 0 = Indicates the transfer is not finished yet. 1 = Indicates that the transfer is complete. Interrupt is generated to CPU if enabled. **NOTE**: This bit is cleared by writing 1 to itself.
[15:12]	**SLEEP**	**Suspend Interval (Master Only)** These four bits provide configurable suspend interval between two successive transmit/receive transactions in a transfer. The suspend interval is from the last falling clock edge of the current transaction to the first rising clock edge of the successive transaction if CLKP = 0. If CLKP = 1, the interval is from the rising clock edge to the falling clock edge. The default value is 0x0. When TXNUM = 00b, setting this field has no effect on transfer except as determined by REORDER[0] setting. The suspend interval is determined according to the following equation: (SLEEP[3:0] + 2) * period of SCLK
[11]	**CLKP**	**Clock Polarity** 0 = SCLK idle low. 1 = SCLK idle high.
[10]	**LSB**	**LSB First** 0 = The MSB is transmitted/received first (which bit in SPI1_TX0/1 and SPI1_RX0/1 register that is depends on the TXBITLEN field). 1 = The LSB is sent first on the line (bit 0 of SPI1_TX0/1), and the first bit received from the line will be put in the LSB position in the Rx register (bit 0 of SPI1_RX0/1).

[9:8]	**TXNUM**	**Transmit/Receive Word Numbers** This field specifies how many transmit/receive word numbers should be executed in one transfer. 00 = Only one transmit/receive word will be executed in one transfer. 01 = Two successive transmit/receive word will be executed in one transfer. 10 = Reserved. 11 = Reserved.

[7:3]	**TXBITLEN**	**Transmit Bit Length** This field specifies how many bits are transmitted in one transmit/receive. Up to 32 bits can be transmitted. 00001 = 1 bit 00010 = 2 bit 00011 = 3 bit 00100 = 4 bit 00101 = 5 bit 00110 = 6 bit 00111 = 7 bit 01000 = 8 bit 01001 = 9 bit 01010 = 10 bit 01011 = 11 bit 01100 = 12 bit 01101 = 13 bit 01110 = 14 bit 01111 = 15 bit 10000 = 16 bit 10001 = 17 bit 10010 = 18 bit 10011 = 19 bit 10100 = 20 bit 10101 = 21 bit 10110 = 22 bit 10111 = 23 bit 11000 = 24 bit 11001 = 25 bit 11010 = 26 bit 11011 = 27 bit 11100 = 28 bit 11101 = 29 bit 11110 = 30 bit 11111 = 31 bit 00000 = 32 bit
[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]	**EN**	**Go and Busy Status** 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit starts the transfer. This bit remains set during the transfer and is automatically cleared after transfer finished. **NOTE**: All registers should be set before writing 1 to this EN bit. When a transfer is in progress, writing to any register of the SPI master/slave core has no effect.

SPI Divider Register (SPI1_DIVIDER)

Register	Offset	R/W	Description	Reset Value
SPI1_CLKDIV	SPI1_BA + 0x04	R/W	Clock Divider Register (Master Only)	0x0000_0000

31	30	29	28	27	26	25	24
CLKDIV1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CLKDIV1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CLKDIV0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKDIV0							

Bits	Description	
[31:16]	**CLKDIV1**	**Clock Divider 2 Register** (master only) The value in this field is the 2nd frequency divider of the system clock, PCLK, to generate the serial clock on the output SCLK. The desired frequency is obtained according to the following equation: $f_{sclk} = \frac{f_{pclk}}{(DIVIDER2+1)*2}$
[15:0]	**CLKDIV0**	**Clock Divider Register** (master only) The value in this field is the frequency division of the system clock, PCLK, to generate the serial clock on the output SCLK. The desired frequency is obtained according to the following equation: $f_{sclk} = \frac{f_{pclk}}{(DIVIDER+1)*2}$ In slave mode, the period of SPI clock driven by a master shall satisfy $f_{sclk} \le \frac{f_{pclk}}{5}$ In other words, the maximum frequency of SCLK clock is one fifth of the SPI peripheral clock.

SPI Slave Select Register (SPI1_SSCTL)

Register	Offset	R/W	Description	Reset Value
SPI1_SSCTL	SPI1_BA + 0x08	R/W	Slave 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		**LTRIGFLAG**	**SSLTRIG**	**ASS**	**SSLVL**	**Reserved**	**SSR**

Bits	Description	
[31:6]	**Reserved**	Reserved
[5]	**LTRIGFLAG**	**Level Trigger Flag** When the SSLTRIG bit is set in slave mode, this bit can be read to indicate the received bit number is met the requirement or not. 1 = The received number and received bits met the requirement which defines in TXNUM and TXBITLEN among one transfer. 0 = One of the received number and the received bit length doesn't meet the requirement in one transfer. Note: This bit is READ only
[4]	**SSLTRIG**	**Slave Select Level Trigger (Slave only)** 0 = The input slave select signal is edge-trigger. This is the default value. 1 = The slave select signal will be level-trigger. It depends on SSLVL to decide the signal is active low or active high.
[3]	**ASS**	**Automatic Slave Select (Master only)** 0 = If this bit is cleared, slave select signals are asserted and de-asserted by setting and clearing related bits in SSCTL register. 1 = If this bit is set, SPISS signals are generated automatically. It means that device/slave select signal, which is set in SSCTL register is asserted by the SPI controller when transmit/receive is started by setting EN, and is de-asserted after each transmit/receive is finished.
[2]	**SSLVL**	**Slave Select Active Level** It defines the active level of device/slave select signal (SPISSx0/1). 0 = The slave select signal SPISSx0/1 is active at low-level/falling-edge. 1 = The slave select signal SPISSx0/1 is active at high-level/rising-edge.
[1]	**Reserved**	Reserved

[0]	**SSR**	**Slave Select Register (Master only)** If ASS bit is cleared, writing 1 to any bit location of this field sets the proper SPISSx0/1 line to an active state and writing 0 sets the line back to inactive state. If ASS bit is set, writing 1 to any bit location of this field will select appropriate SPISS 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 level of SPISSx0/1 is specified in SSLVL). **Note:** SPISS is always defined as device/slave select input signal in slave mode.

SPI Data Receive Register (SPI1_RX0)

Register	Offset	R/W	Description	Reset Value
SPI1_RX0	SPI1_BA + 0x10	R	Data Receive Register 0	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** The Data Receive Registers hold the value of received data of the last executed transfer. Valid bits depend on the transmit bit length field in the SPI1_CTL register. For example, if TXBITLEN is set to 0x08 and TXNUM is set to 0x0, bit SPI1_RX0[7:0] holds the received data. **Note:** The Data Receive Registers are read only registers.

SPI Data Receive Register (SPI1_RX1)

Register	Offset	R/W	Description	Reset Value
SPI1_RX1	SPI1_BA + 0x14	R	Data Receive Register 1	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** The Data Receive Registers hold the value of received data of the last executed transfer. Valid bits depend on the transmit bit length field in the SPI1_CTL register. For example, if TXBITLEN is set to 0x08 and TXNUM is set to 0x0, bit SPI1_RX0[7:0] holds the received data. **NOTE:** The Data Receive Registers are read only registers.

SPI Data Transmit Register (SPI1_TX0)

Register	Offset	R/W	Description	Reset Value
SPI1_TX0	SPI1_BA + 0x20	W	Data Transmit Register 0	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** The Data Transmit Registers hold the data to be transmitted in the next transfer. Valid bits depend on the transmit bit length field in the SPI1_CTL register. For example, if TXBITLEN is set to 0x08 and the TXNUM is set to 0x0, the bit SPI1_TX0[7:0] will be transmitted in next transfer. If TXBITLEN is set to 0x00 and TXNUM is set to 0x1, the core will perform two 32-bit transmit/receive successive using the same setting (the order is SPI1_TX0[31:0], SPI1_TX1[31:0]).

SPI Data Transmit Register (SPI1_TX1)

Register	Offset	R/W	Description	Reset Value
SPI1_TX1	SPI1_BA + 0x24	W	Data Transmit Register 1	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** The Data Transmit Registers hold the data to be transmitted in the next transfer. Valid bits depend on the transmit bit length field in the SPI1_CTL register. For example, if TXBITLEN is set to 0x08 and the TXNUM is set to 0x0, the bit SPI1_TX0[7:0] will be transmitted in next transfer. If TXBITLEN is set to 0x00 and TXNUM is set to 0x1, the core will perform two 32-bit transmit/receive successive using the same setting (the order is SPI1_TX0[31:0], SPI1_TX1[31:0]).

SPI Variable Clock Pattern Flag Register (SPI1_VARCLK)

Register	Offset	R/W	Description	Reset Value
SPI1_VARCLK	SPI1_BA + 0x34	R/W	Variable Clock Pattern Register	0x007F_FF87

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

Bits	Description	
[31:0]	**VARCLK**	**Variable Clock Pattern** The value in this field is the frequency pattern of the SPI clock. If the bit field of VARCLK is '0', the output frequency of SCLK is given by the value of DIVIDER. If the bit field of VARCLK is '1', the output frequency of SCLK is given by the value of CLKDIV1. Refer to register CLKDIV0. Refer to Figure 5-62 Variable Serial Clock Frequency paragraph for detailed description. Note: Used for CLKP = 0 only, 16 bit transmission.

DMA Control Register (SPI1_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPI1_PDMACTL	SPI1_BA + 0x38	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						**RXMDAEN**	**TXMDAEN**

Bits	Description	
[31:2]	**Reserved**	Reserved
[1]	**RXMDAEN**	**Receive DMA Start** 1 = Enable 0 = Disable Set this bit to 1 will start the receive DMA process. SPI module will issue request to DMA module automatically.
[0]	**TXMDAEN**	**Transmit DMA Start** 1 = Enable 0 = Disable Set this bit to 1 will start the transmit DMA process. SPI module will issue request to DMA module automatically. If using DMA mode to transfer data, remember not to set EN bit of SPI_CTL register. The DMA controller inside SPI module will set it automatically whenever necessary.

5.11 Timer Controller

5.11.1 General Timer Controller

The ISD91200 provides two general 24bit timer modules, TIMER0 and TIMER1. They allow the user to implement event counting or provide timing control for applications. The timer can perform functions such as frequency measurement, event counting, interval measurement, clock generation and delay timing. The timer can generates an interrupt signal upon timeout and provide the current value of count during operation.

5.11.2 Features

- Independent clock source for each channel(TMR0_CLK, TMR1_CLK).
- Time out period = (Period of timer clock input) * (8-bit prescale + 1) * (24-bit CMPDAT)
- Maximum count cycle time = (1 / TMR_CLK) * (2^8) * (2^24).
- Internal 24-bit up counter is readable through TIMERx_CNT (Timer Data Register).

5.11.3 Timer Controller Block Diagram

Each channel is equipped with an 8-bit pre-scale counter, a 24-bit up-counter, a 24-bit compare register and an interrupt request signal. Refer to below figure for the timer controller block diagram. There are five options of clock source for each channel, Figure 5-68 Clock Source of Timer Controller illustrate the clock source control function.

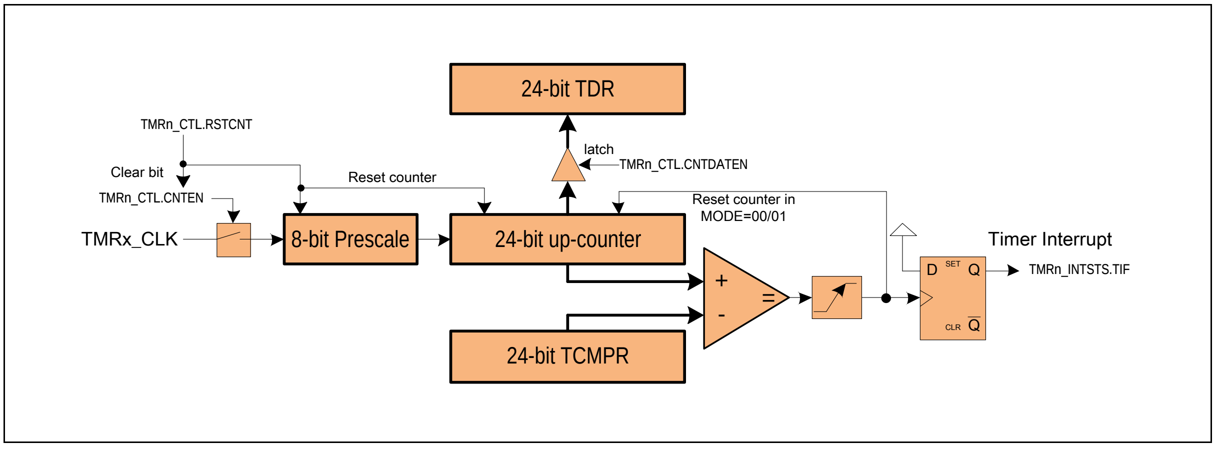


Figure 5-67 Timer Controller Block Diagram

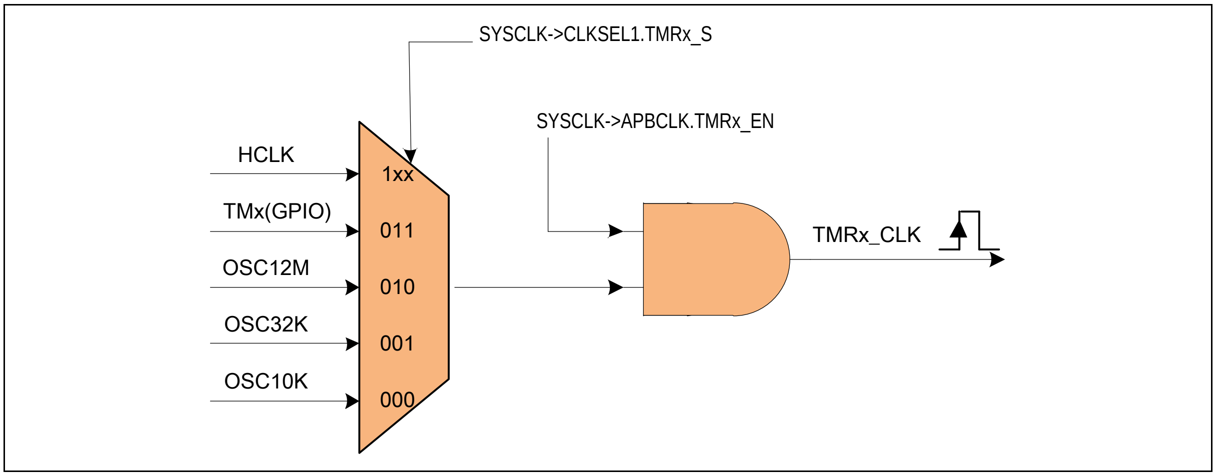


Figure 5-68 Clock Source of Timer Controller

5.11.4 Register Map

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

Register	Offset	R/W	Description	Reset Value
TMR Base Address: **TMRn_BA=0x4001_0000+(0x20*n)** **n=0,1**				
TMRn_CTL	TMRn_BA+0x00	R/W	Timer Control and Status Register	0x0000_0005
TMRn_CMP	TMRn_BA+0x04	R/W	Timer Compare Register	0x0000_0000
TMRn_INTSTS	TMRn_BA+0x08	R/W	Timer Interrupt Status Register	0x0000_0000
TMRn_CNT	TMRn_BA+0x0C	R/W	Timer Data Register	0x0000_0000

5.11.5 Register Description

Timer Control Register (TIMERn_CTL)

Register	Offset	R/W	Description	Reset Value
TMRn_CTL	TMRn_BA+0x00	R/W	Timer 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							**CNTDATEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC							

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**CNTEN**	**Counter Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. Note1: Setting CNTEN = 1 enables 24-bit counter. It continues count from last value. Note2: This bit is auto-cleared by hardware in one-shot mode (OPMODE = 00b) when the timer interrupt is generated (INTEN = 1b).
[29]	**INTEN**	**Interrupt Enable Bit** 0 = Disable TIMER Interrupt. 1 = Enable TIMER Interrupt. If timer interrupt is enabled, the timer asserts its interrupt signal when the count is equal to TIMERx_CMP.
[28:27]	**OPMODE**	**Timer Operating Mode** 0 = The timer is operating in the one-shot mode. The associated interrupt signal is generated once (if INTEN is enabled) and CNTEN is automatically cleared by hardware. 1 = The timer is operating in the periodic mode. The associated interrupt signal is generated periodically (if INTEN is enabled). 2 = Reserved. 3 = The timer is operating in continuous counting mode. The associated interrupt signal is generated when CNT = TIMERx_CMP (if INTEN is enabled); however, the 24-bit up-counter counts continuously without reset.
[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 24-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.
[24:17]	**Reserved**	Reserved.
[16]	**CNTDATEN**	**Data Latch Enable** When CNTDATEN is set, TIMERx_CNT (Timer Data Register) will be updated continuously with the 24-bit up-counter value as the timer is counting. 1 = Timer Data Register update enable. 0 = Timer Data Register update disable.
[15:8]	**Reserved**	Reserved.
[7:0]	**PSC**	**Pre-scale Counter** Clock input is divided by PSC+1 before it is fed to the counter. If PSC = 0, then there is no scaling.

Timer Compare Register (TIMERn_CMP)

Register	Offset	R/W	Description	Reset Value
TMRn_CMP	TMRn_BA+0x04	R/W	Timer Compare Register	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24:0]	**CMPDAT**	**Timer Comparison Value** CMPDAT is a 24-bit comparison register. When the 24-bit up-counter is enabled and its value is equal to CMPDAT value, a Timer Interrupt is requested if the timer interrupt is enabled with TIMERx_CTL.INTEN = 1. The CMPDAT value defines the timer cycle time. Time out period = (Period of timer clock input) * (8-bit PSC + 1) * (24-bit CMPDAT). NOTE1: Never set CMPDAT to 0x000 or 0x001. Timer will not function correctly. NOTE2: Regardless of CEN state, whenever a new value is written to this register, TIMER will restart counting using this new value and abort previous count.

Timer Interrupt Status Register (TIMERn_INTSTS)

Register	Offset	R/W	Description	Reset Value
TMRn_INTSTS	TMRn_BA+0x08	R/W	Timer 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.
[0]	**TIF**	**Timer Interrupt Flag** This bit indicates the interrupt status of Timer. TIF bit is set by hardware when the 24-bit counter matches the timer comparison value (CMPDAT). It is cleared by writing 1.

Timer Data Register (TIMERn_CNT)

Register	Offset	R/W	Description	Reset Value
TMRn_CNT	TMRn_BA+0x0C	R/W	Timer Data Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Timer Data Register** When TIMERx_CTL.CNTDATEN is set to 1, the internal 24-bit timer up-counter value will be latched into CNT. User can read this register for the up-counter value.

5.12 Watchdog Timer

The purpose of Watchdog Timer is to perform a system reset if software is not responding as designed. This prevents system from hanging for an infinite period of time. The watchdog timer includes a 18-bit free running counter with programmable time-out intervals.

Setting WDTEN 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 will be set immediately to request a WDT interrupt if the watchdog timer interrupt enable bit INTEN is set, in the meantime, a specified delay time follows the time-out event. User must set RSTCNT (Watchdog timer reset) high to reset the 18-bit WDT counter to prevent Watchdog timer reset before the delay time expires. RSTCNT 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. If the WDT counter has not been cleared after the specific delay time expires, the watchdog timer will set Watchdog Timer Reset Flag (RSTF) high and reset CPU. This reset will last 64 WDT clocks then CPU restarts executing program from reset vector (0x0000 0000). RSTF will not be cleared by Watchdog reset. User may poll WTFR by software to recognize the reset source.

If the application uses any sleep modes (calling wfi or wfe instructions), the watchdog reset may not fully reset the M0 core due to parts of the core being un-clocked. In this case application should detect the RSTF in boot sequence and perform a Deep Power Down (DPD) to ensure complete reset. See the Timer driver sample code for example.

Table 5-9 Watchdog Timeout Interval Selection

TOUTSEL	Interrupt Timeout	Watchdog Reset Timeout	RSTCNT Timeout Interval (WDT_CLK=HIRC 49.152 MHz)	RSTCNT Timeout Interval (WDT_CLK=LXT 32kHz)
000	2^4 WDT_CLK	(2^4 + 1024) WDT_CLK	21.2us	31.7 ms
001	2^6 WDT_CLK	(2^6 + 1024) WDT_CLK	22.1 us	33.2 ms
010	2^8 WDT_CLK	(2^8 + 1024) WDT_CLK	26.0 us	39 ms
011	2^{10} WDT_CLK	(2^{10} + 1024) WDT_CLK	41.7 us	64 ms
100	2^{12} WDT_CLK	(2^{12} + 1024) WDT_CLK	104.2 us	160 ms
101	2^{14} WDT_CLK	(2^{14} + 1024) WDT_CLK	354.2 us	544 ms
110	2^{16} WDT_CLK	(2^{16} + 1024) WDT_CLK	1.4 ms	2080 ms
111	2^{18} WDT_CLK	(2^{18} + 1024) WDT_CLK	5.4 ms	8224 ms

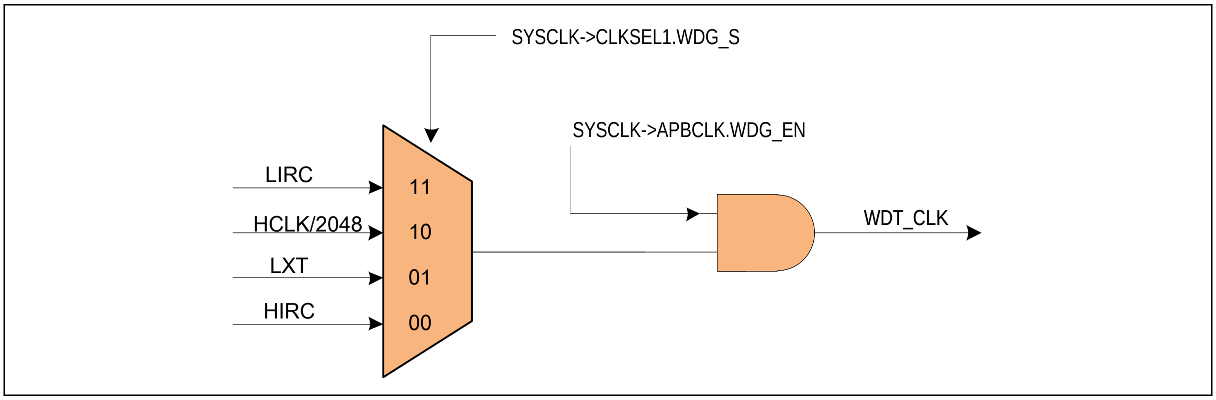
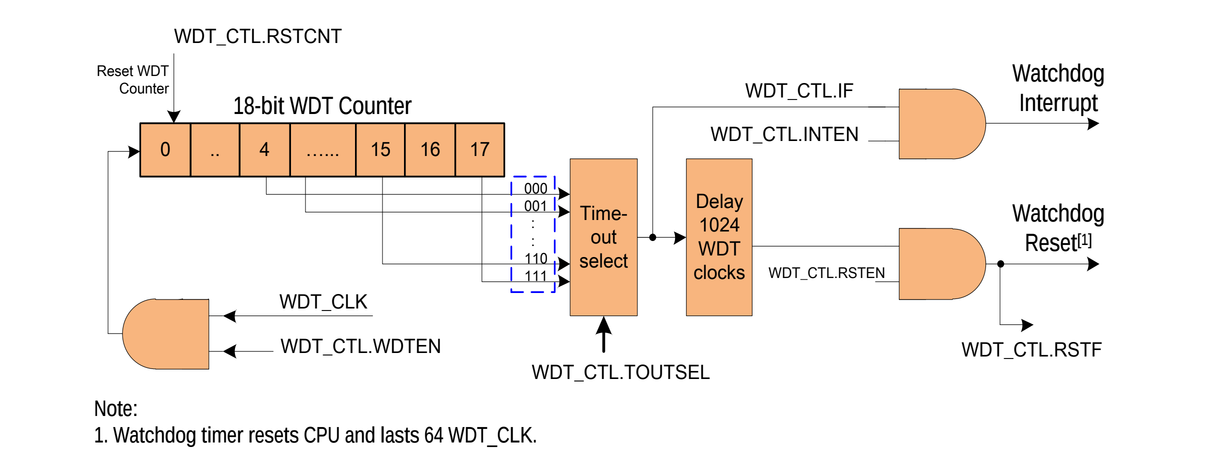


Figure 5-69 Watchdog Timer Clock Control



Note:
1. Watchdog timer resets CPU and lasts 64 WDT_CLK.

Figure 5-70 Watchdog Timer Block Diagram

5.12.1 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

5.12.2 Register Description

Watchdog Timer Control Register (WDT_CTL)

This is a protected register, to write to register, first issue the unlock sequence (refer to 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.
[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 WDG not reset. The timeout is given by: Interrupt Timeout = 2^(2xTOUTSEL+4) x WDT_CLK. Reset Timeout = (2^(2xTOUTSEL+4) +1024) x WDT_CLK. Where WDT_CLK is the period of the Watchdog Timer clock source.
[7]	**WDTEN**	**Watchdog Timer Enable** 0 = Disable the Watchdog timer (This action will reset the internal counter). 1 = Enable the Watchdog timer.
[6]	**INTEN**	**Watchdog Timer Interrupt Enable** 0 = Disable the Watchdog timer interrupt. 1 = Enable the Watchdog timer interrupt.
[5:4]	**Reserved**	Reserved.
[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** Setting this bit will enable the Watchdog timer reset function. 0 = Disable Watchdog timer reset function. 1= Enable Watchdog timer reset function.
[0]	**RSTCNT**	**Clear Watchdog Timer** 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. **Note:** This bit will auto clear after few clock cycle

5.13 UART Interface Controller

The ISD91200 includes a Universal Asynchronous Receiver/Transmitter (UART). The UART supports high speed operation and flow control functions as well as protocols for Serial Infrared (IrDA) and Local interconnect Network (LIN).

5.13.1 Overview

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 controller also supports LIN(Local Interconnect Network) master mode function and IrDA SIR (Serial Infrared)function. The UART channel supports seven types of interrupts including transmitter FIFO empty interrupt(THERINT), 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), Buffer error interrupt (BUFERRINT) and LIN receiver break field detected interrupt.

The UART has a 8-byte transmit FIFO (TX_FIFO) and a 8-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 (UART_BAUD). Table 5-10 UART Baud Rate Equation 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.

The UART controller also provides Serial IrDA (SIR, Serial Infrared) function (UART_FUNCSEL.IRDAEN =1 to enable IrDA function). The SIR specification defines a short-range infrared asynchronous serial transmission mode with one start bit, 8 data bits, and 1 stop bit. The maximum data rate is 115.2 Kbps (half duplex). The IrDA SIR block contains an IrDA SIR Protocol encoder/decoder. The IrDA SIR protocol is half-duplex only. So it cannot transmit and receive data at the same time. The IrDA SIR physical layer specifies a minimum 10ms transfer delay between transmission and reception. This delay must be implemented by software.

The alternate function of UART controller is LIN (Local Interconnect Network) function. The LIN mode is selected by setting the UART_FUNCSEL.LINEN bit. In LIN mode, one start bit, 8-bit data format with 1-bit stop bit are generated in accordance with the LIN standard.

Table 5-10 UART Baud Rate Equation

Mode	BAUDM1	BAUDM0	EDIVM1[3:0]	BRD[15:0]	Baud Rate Equation
0	0	0	B	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 ≥ 3

Table 5-11 UART Baud Rate Setting Table

System Clock = 49.152MHz						
Baud rate	Mode0	%err	Mode1	%err	Mode2	%err
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

5.13.2 Features of UART controller

- UART supports 8 byte FIFO for receive and transmit data payloads.
- PDMA access support.
- Auto flow control function (/CTS, /RTS) supported.
- Programmable baud-rate generator.
- 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.
- IrDA SIR Function.
- LIN master mode.

5.13.3 Block Diagram

The UART clock control and block diagram are shown as following.

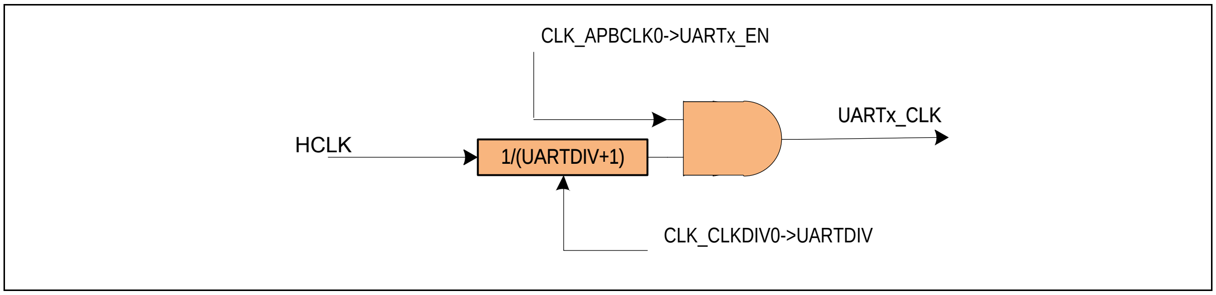


Figure 5-71 UART Clock Control Diagram

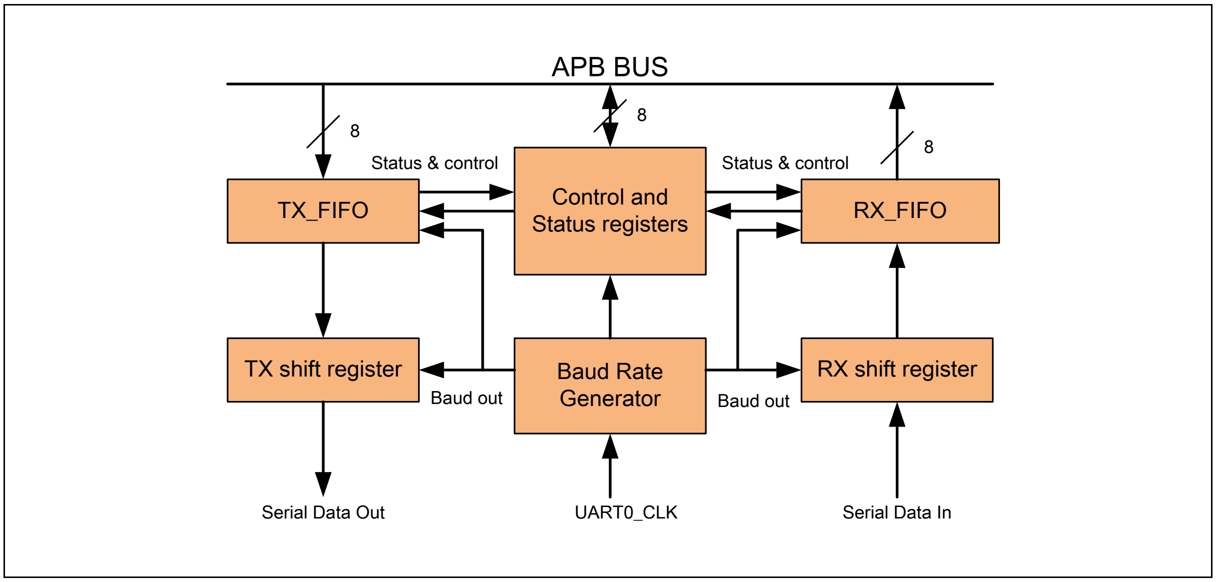


Figure 5-72 UART Block Diagram

TX_FIFO
The transmitter is buffered with an 8 byte FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO
The receiver is buffered with an 8 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 5-11 UART Baud Rate Setting Table for the baud rate equation.

Control and Status Register
This is a register set, including the FIFO control registers (UART_FIFO), FIFO status registers (UART_FIFOSTS), and line control register (UART_LINE) for transmitter and receiver. The time out control register (UART_TOUT) identifies the condition of time out interrupt. This register set also includes the interrupt enable register (UART_INTEN) and interrupt status register (UART_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(THERINT), 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 5-73 Auto Flow Control Block Diagram demonstrates the auto-flow control block diagram.

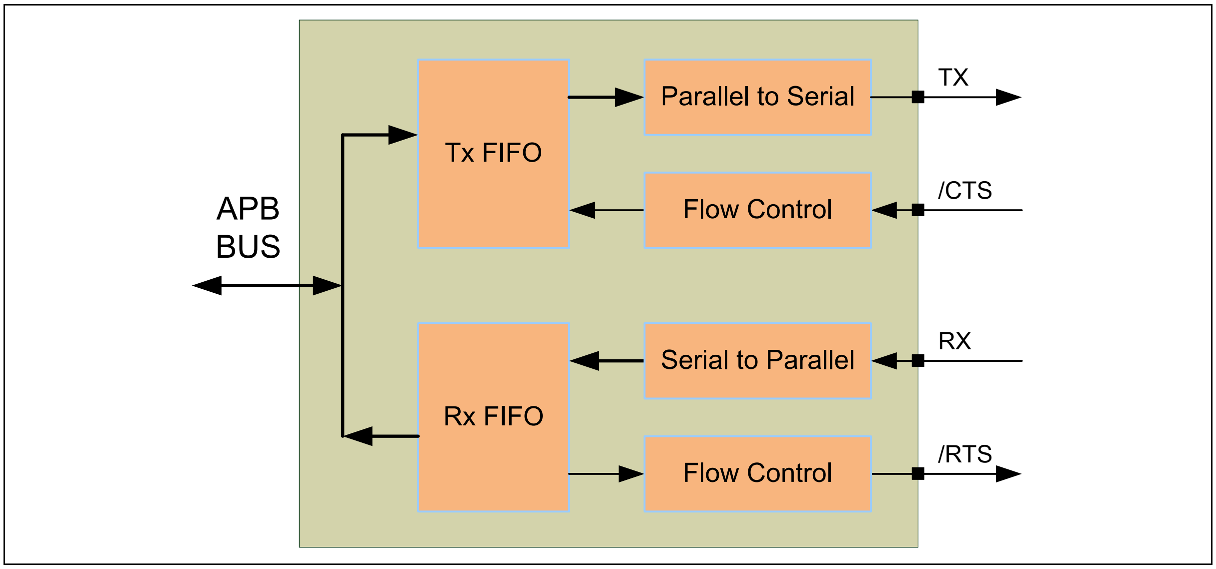


Figure 5-73 Auto Flow Control Block Diagram

5.13.4 IrDA Mode

The UART supports IrDA SIR (Serial Infrared) Transmit Encoder and Receive Decoder. IrDA mode is selected by setting the UART_FUNCSEL.IRDAENbit.

When in IrDA mode, the UART_BAUD.BAUDM1 register must be zero and baud rate is given by:

Baud Rate = UART_CLK / (16 * BRD), where BRD is Baud Rate Divider in the UART_BAUD.BRD register.

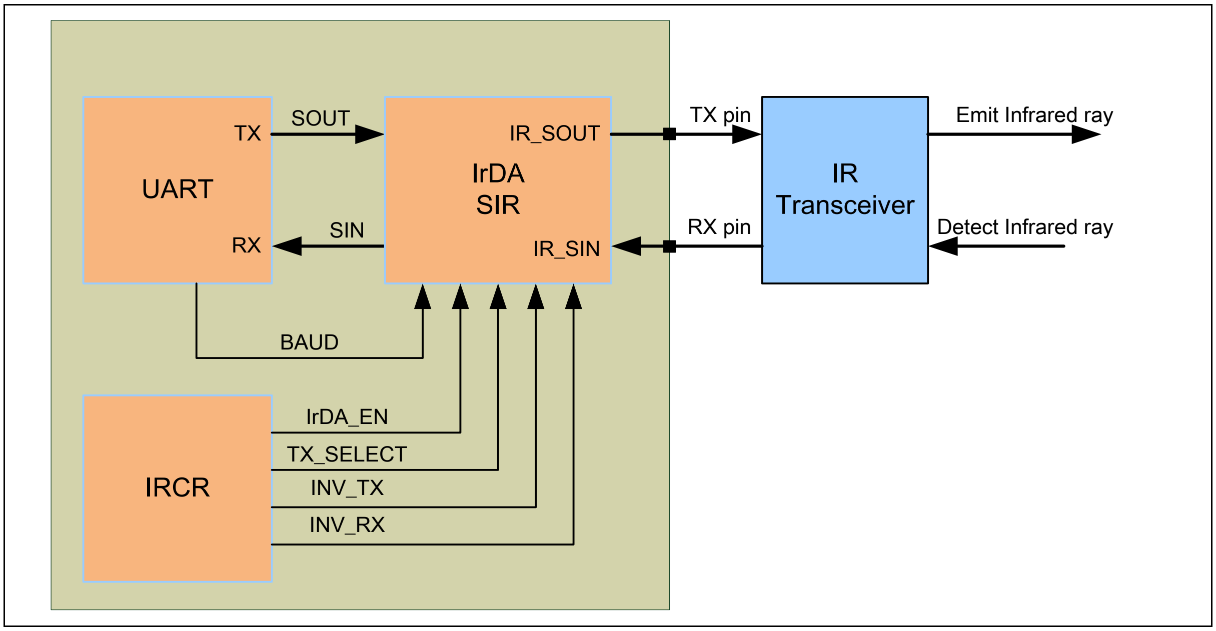


Figure 5-74 IrDA Block Diagram

5.13.4.1 IrDA SIR Transmit Encoder

The IrDA SIR Transmit Encoder modulates Non-Return-to Zero (NRZ) transmission bit stream from UART serial output. The IrDA SIR physical layer specifies use of Return-to-Zero, Inverted (RZI) modulation scheme which represents logic 0 as an infrared light pulse. The modulated output pulse stream is transmitted to an external output driver and infrared LED (Light Emitting Diode).In normal mode, the transmitted pulse width is specified as 3/16the period of the baud rate.

5.13.4.2 IrDA SIR Receive Decoder

The IrDA SIR Receive Decoder demodulates the return-to-zero bit stream from the input detector and outputs the NRZ serial bit stream to the UART received data input. The IR_SIN decoder input is normally high in the idle state. Because of this, UART_IRDA.RXINV should be set 1 by default). A start bit is detected when the IR_SIN decoder input is LOW.

5.13.4.3 IrDA SIR Operation

The IrDA SIR Encoder/decoder provides functionality which converts between UART data stream and half duplex serial SIR interface. Below figure shows the IrDA encoder/decoder waveform:

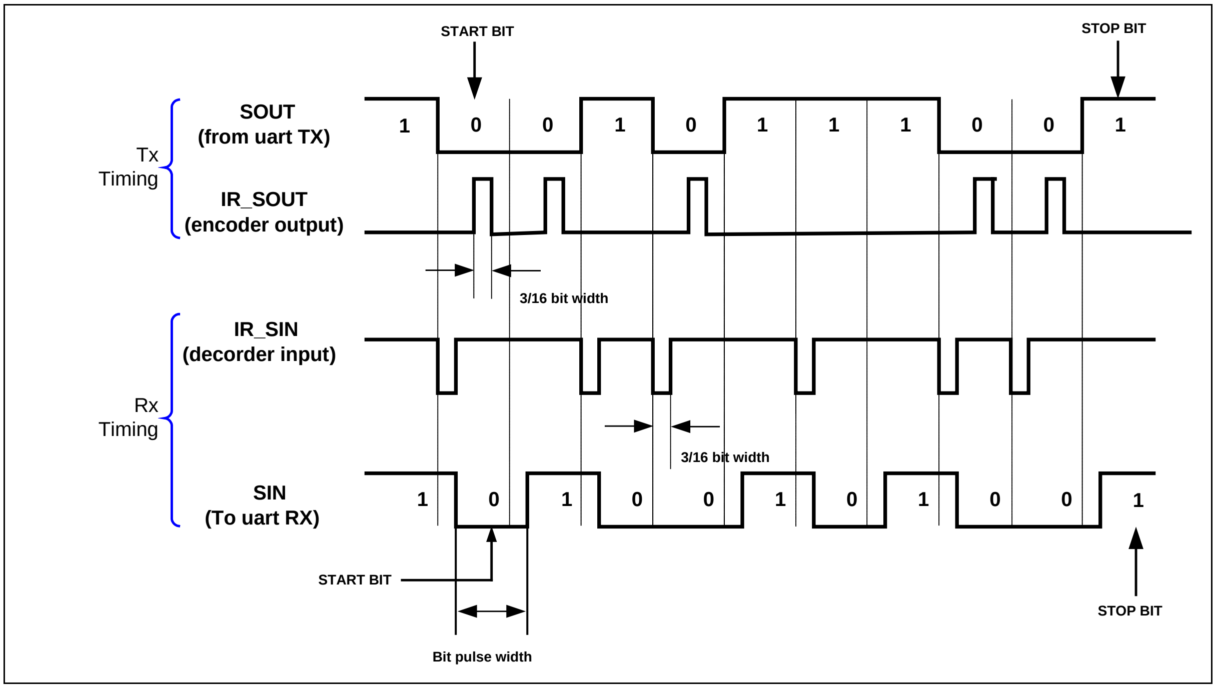


Figure 5-75 IrDA Tx/Rx Timing Diagram

5.13.5 LIN (Local Interconnection Network) mode

The UART supports a Local Interconnection Network (LIN) function. LIN mode is selected by setting the UART_FUNCSEL.LINEN bit. In LIN mode, each byte field is initiated by a start bit with value zero (dominant), followed by 8 data bits (LSB is first)and ended by 1 stop bit with value one (recessive) in accordance with the LIN standard (http://www.lin-subbus.org/).

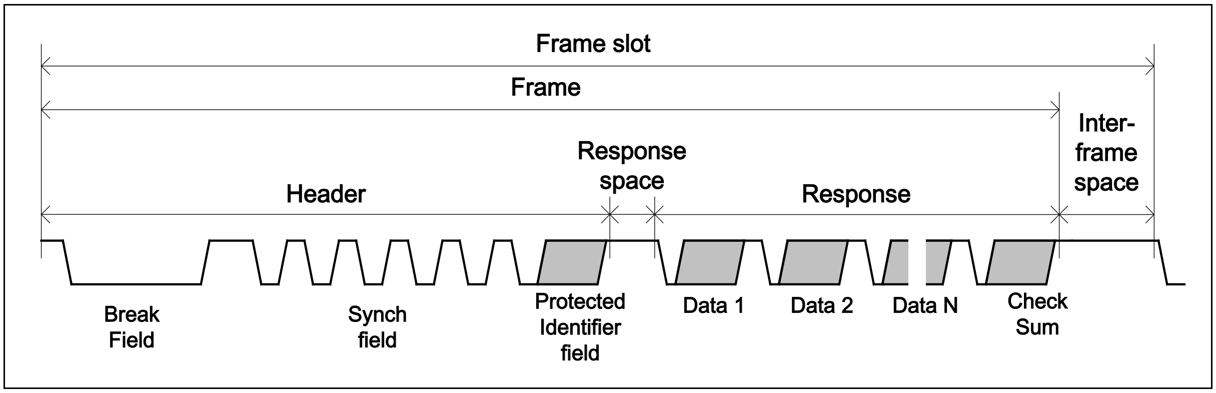


Figure 5-76 Structure of LIN Frame

The program flow of LIN Bus Transmit transfer (Tx) is shown as following

1. Set the UART_FUNCSEL.LINEN bit to enable LIN Bus mode.
2. Set UART_ALTCTL.BRKFL to choose break field length. The break field length is BRKFL+2.
3. Fill 0x55 to UART_DAT to request synch field transmission.
4. Request Identifier Field transmission by writing the protected identifier value to UART_DAT
5. Set the UART_ALTCTL.LINTXEN bit to start transmission (When break filed operation is finished, LINTX_EN will be cleared automatically).
6. When the STOP bit of the last byte UART_DAT has been sent to bus, hardware will set flag UART_FIFOSTS.TXEMPTYF to 1.
7. Fill N bytes data and Checksum to UART_DAT then repeat step 5 and 6 to transmit the data.

The program flow of LIN Bus Receiver transfer (Rx) is show as following

1. Set the UART_FUNCSEL.LINEN bit to enable LIN Bus mode.
2. Set the UART_ALTCTL.LINRXEN bit register to enable LIN Rx mode.
3. Wait for the flag UART_INTSTS.LINIF to indicate Rx received Break field or not.
4. Wait for the flag UART_INTSTS.RDAIF read back the UART_DAT register.

5.13.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UART Base Address: **UARTn_BA = 0x4005_0000 + (n*0x8000)** **n=0,1**				
UARTn_DAT	UARTn_BA+0x00	R/W	UART Receive/Transfer FIFO Register.	0x0000_0000
UARTn_INTEN	UARTn_BA+0x04	R/W	UART Interrupt Enable Register.	0x0000_0000
UARTn_FIFO	UARTn_BA+0x08	R/W	UART FIFO Control Register.	0x0000_0000
UARTn_LINE	UARTn_BA+0x0C	R/W	UART Line Control Register.	0x0000_0000
UARTn_MODEM	UARTn_BA+0x10	R/W	UART Modem Control Register.	0x0000_0000
UARTn_MODEMSTS	UARTn_BA+0x14	R/W	UART Modem Status Register.	0x0000_0010
UARTn_FIFOSTS	UARTn_BA+0x18	R/W	UART FIFO Status Register.	0x1040_4000
UARTn_INTSTS	UARTn_BA+0x1C	R/W	UART Interrupt Status Register.	0x0000_0002
UARTn_TOUT	UARTn_BA+0x20	R/W	UART Time Out Register	0x0000_0000
UARTn_BAUD	UARTn_BA+0x24	R/W	UART Baud Rate Divisor Register	0x0F00_0000
UARTn_IRDA	UARTn_BA+0x28	R/W	UART IrDA Control Register.	0x0000_0040
UARTn_ALTCTL	UARTn_BA+0x2C	R/W	UART LIN Control Register.	0x0000_0000
UARTn_FUNCSEL	UARTn_BA+0x30	R/W	UART Function Select Register.	0x0000_0000

5.13.7 Register Description

Receive FIFO Data Register (UARTn_DAT)

Register	Offset	R/W	Description	Reset Value
UARTn_DAT	UARTn_BA+0x00	R/W	UART Receive/Transfer 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**	
[7:0]	**DAT**	**Receive 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).

Interrupt Enable Register (UARTn_INTEN)

Register	Offset	R/W	Description	Reset Value
UARTn_INTEN	UARTn_BA+0x04	R/W	UART Interrupt Enable Register.	0x0000_0000

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td>DMARXEN</td><td>DMATXEN</td><td>ATOCTSEN</td><td>ATORTSEN</td><td>TOCNTEN</td><td colspan="2">Reserved</td><td>LINIEN</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">Reserved</td><td>BUFERRIEN</td><td>RXTOIEN</td><td>MODEMIEN</td><td>RLSIEN</td><td>THREIEN</td><td>RDAIEN</td></tr>
</table>

Bits	Description	
[31:16]	**Reserved**	Reserved.
[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. 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 asserted).
[12]	**ATORTSEN**	**RTS Auto Flow Control Enable** 0 = Disable RTS auto flow control. 1 = Enable. When RTS auto-flow is enabled, if the number of bytes in the Rx FIFO equals UART_FIFO.RTSTRGLV, the UART will de-assert the RTS signal.
[11]	**TOCNTEN**	**Time-out Counter Enable** 0 = Disable Time-out counter. 1 = Enable.
[10:9]	**Reserved**	Reserved.
[8]	**LINIEN**	**LIN RX Break Field Detected Interrupt Enable** 0 = Mask off Lin bus Rx break field interrupt. 1 = Enable Lin bus Rx break field interrupt.
[7:6]	**Reserved**	Reserved.

[5]	**BUFERRIEN**	**Buffer Error Interrupt Enable** 0 = Mask off BUFERRINT. 1 = Enable IBUFERRINT.
[4]	**RXTOIEN**	**Receive Time Out Interrupt Enable** 0 = Mask off RXTOINT. 1 = Enable RXTOINT.
[3]	**MODEMIEN**	**Modem Status Interrupt Enable** 0 = Mask off MODEMINT. 1 = Enable MODEMINT.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable** 0 = Mask off RLSINT. 1 = Enable RLSINT.
[1]	**THREIEN**	**Transmit FIFO Register Empty Interrupt Enable** 0 = Mask off THERINT. 1 = Enable THERINT.
[0]	**RDAIEN**	**Receive Data Available Interrupt Enable** 0 = Mask off RDAINT. 1 = Enable RDAINT.

FIFO Control Register (UARTn_FIFO)

Register	Offset	R/W	Description	Reset Value
UARTn_FIFO	UARTn_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.
[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). Value : Trigger Level (Bytes) 0 : 1 1 : 4 2 : 8
[15:8]	**Reserved**	Reserved.
[7:4]	**RFITL**	**Receive FIFO Interrupt (RDAINT) Trigger Level** When the number of bytes in the receive FIFO equals the RFITL then the RDAIF will be set and, if enabled, an RDAINT interrupt will generated. Value : INTR_RDA Trigger Level (Bytes) 0 : 1 1 : 4
[3]	**Reserved**	Reserved.
[2]	**TXRST**	**Transmit FIFO Reset** When TXRST 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.

[1]	**RXRST**	**Receive FIFO Reset** When RXRST 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 receiving internal state machine and pointers. Note: This bit will auto-clear after 3 UART engine clock cycles.
[0]	**Reserved**	Reserved.

Line Control Register (UARTn_LINE)

Register	Offset	R/W	Description	Reset Value
UARTn_LINE	UARTn_BA+0x0C	R/W	UART Line Control Register.	0x0000_0000

7	6	5	4	3	2	1	0
Reserved	BCB	SPE	EPE	PBE	NSB	WLS	

Bits	Description	
[31:7]	**Reserved**	Reserved.
[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.
[4]	**EPE**	**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.
[3]	**PBE**	**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.
[2]	**NSB**	**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;.
[1:0]	**WLS**	**Word Length Select** 0 (5bits), 1(6bits), 2(7bits), 3(8bits)

MODEM Control Register (UARTn_MODEM)

Register	Offset	R/W	Description	Reset Value
UARTn_MODEM	UARTn_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			LBMEN	Reserved		RTS	Reserved

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**RTSSTS**	**RTS Pin State (Read Only)** This bit is the pin status of RTS.
[12:10]	**Reserved**	Reserved.
[9]	**RTSACTLV**	**Request-to-send (RTS) Active Trigger Level** This bit can change the RTS trigger level. 0= RTS is active low level. 1= RTS is active high level.
[8:5]	**Reserved**	Reserved.
[4]	**LBMEN**	**Loopback Mode Enable** 0=Disable. 1=Enable.
[3:2]	**Reserved**	Reserved.
[1]	**RTS**	**RTS (Request-to-send) Signal** If UART_INTEN.ATORTSEN = 0, this bit controls whether RTS pin is active or not. 0 = Drive RTS inactive (= ~RTSACTLV). 1 = Drive RTS active (= RTSACTLV).
[0]	**Reserved**	Reserved.

Modem Status Register (UARTn_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UARTn_MODEMSTS	UARTn_BA+0x14	R/W	UART Modem Status Register.	0x0000_0010

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.
[8]	**CTSACTLV**	**Clear-to-send (CTS) Active Trigger Level** This bit can change the CTS trigger level. 0= CTS is active low level. 1= CTS is active high level.
[7:5]	**Reserved**	Reserved.
[4]	**CTSSTS**	**CTS Pin Status (Read Only)** This bit is the pin status of CTS.
[3:1]	**Reserved**	Reserved.
[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 UART_INTEN.MODEMIEN = 1. NOTE: This bit is cleared by writing 1 to itself.

FIFO Status Register (UARTn_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
UARTn_FIFOSTS	UARTn_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.
[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.
[24]	**TXOVIF**	**Tx Overflow Error Interrupt Flag** If the Tx FIFO (UART_DAT) is full, an additional write to UART_DAT 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 Tx FIFO 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 Tx FIFO, 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 Rx FIFO 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.
[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.
[0]	**RXOVIF**	**Rx Overflow Error Interrupt Flag** If the Rx FIFO (UART_DAT) 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.

Interrupt Status Register (UARTn_INTSTS)

Register	Offset	R/W	Description	Reset Value
UARTn_INTSTS	UARTn_BA+0x1C	R/W	UART Interrupt Status Register.	0x0000_0002

31	30	29	28	27	26	25	24
DLININT	Reserved	DBERRINT	DRXTOINT	DMODEMI	DRLSINT	Reserved	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DLINIF	Reserved	DBERRIF	DRXTOIF	DMODEMIF	DRLSIF	Reserved	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
LININT	Reserved	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THERINT	RDAINT
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LINIF	Reserved	BUFERRIF	RXTOIF	MODENIF	RLSIF	THREIF	RDAIF

Bits	Description	
[31]	**DLININT**	**DMA MODE LIN Bus Rx Break Field Detected Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DLINIF.
[30]	**Reserved**	Reserved.
[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]	**DMODEMI**	**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]	**Reserved**	Reserved.
[24]	**Reserved**	Reserved.
[23]	**DLINIF**	**DMA MODE LIN Bus Rx Break Field Detected Flag** This bit is set when LIN controller detects a break field. This bit is cleared by writing a 1.
[22]	**Reserved**	Reserved.

[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.TOUT_IEN is enabled a CPU interrupt request will be generated. NOTE: This bit is read only and user can read FIFO to clear it.
[19]	**DMODEMIF**	**DMA MODE MODEM Interrupt Flag (Read Only)** This bit is set when the CTS pin has changed state (UART_MODEMSTS.CTSDETF=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.CTSDETF 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.
[17:16]	**Reserved**	Reserved.
[15]	**LININT**	**LIN Bus Rx Break Field Detected Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.LINIEN and LINIF.
[14]	**Reserved**	Reserved.
[13]	**BUFERRINT**	**Buffer Error Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.BUFERRIEN and BUFERRIF.
[12]	**RXTOINT**	**Time Out Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.RXTOIEN and RXTOIF.
[11]	**MODEMINT**	**MODEM Status Interrupt Indicator to Interrupt** Logical AND of UART_INTEN.MODEMIEN and MODENIF.
[10]	**RLSINT**	**Receive Line Status Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.RLSIEN and RLSIF.
[9]	**THERINT**	**Transmit Holding Register Empty Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.THREIEN and THREIF.
[8]	**RDAINT**	**Receive Data Available Interrupt Indicator to Interrupt Controller** Logical AND of UART_INTEN.RDAIEN and RDAIF.
[7]	**LINIF**	**LIN Bus Rx Break Field Detected Flag** This bit is set when LIN controller detects a break field. This bit is cleared by writing a 1.
[6]	**Reserved**	Reserved.

[5]	**BUFERRIF**	**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.
[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 UART_INTEN.TOUT_IEN is enabled a CPU interrupt request will be generated. NOTE: This bit is read only and user can read FIFO to clear it.
[3]	**MODENIF**	**MODEM Interrupt Flag (Read Only)** This bit is set when the CTS pin has changed state (UART_MODEMSTS.CTSDETF=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.CTSDETF 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, 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.
[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 UART_INTEN.THREIEN 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 UART_FIFO.RFITL then the RDAIF will be set. If UART_INTEN.RDAIEN 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).

When the DMA controller is used to transmit or receive data to the UART, an alternate set of flags and interrupt indicators are generated. These are equivalent to the normal mode set above and are summarized in below tables.

Table 5-12 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
LIN RX Break Field Detected interrupt	LINIEN	DLININT	DLINIF	Write ‘1’ to LINIF
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	DMODEMI	DMODEMIF = (CTSDETF)	Write ‘1’ to CTSDETF
Receive Line Status Interrupt RLSINT	RLSIEN	DRLSINT	DRLSIF = (BIF or FEF or PEF)	Write ‘1’ to BIF/FEF/PEF

Table 5-13 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
LIN RX Break Field Detected interrupt	LINIEN	LININT	LINIF	Write ‘1’ to LINIF
Buffer Error Interrupt BUF_ERR_INT	BUFERRIEN	BUFERRINT	BUFERRIF = (TXOVIF or RXOVIF)	Write ‘1’ to TXOVIF/ RXOVIF
Rx Timeout Interrupt RXTOINT	RXTOIEN	RXTOINT	RXTOIF	Read data FIFO
Modem Status Interrupt MODEMINT	MODEMIEN	MODEMINT	MODENIF = (CTSDETF)	Write ‘1’ to CTSDETF
Receive Line Status Interrupt RLSINT	RLSIEN	RLSINT	RLSIF = (BIF or FEF or PEF)	Write ‘1’ to BIF/FEF/PEF
Transmit Holding Register Empty Interrupt THERINT	THREIEN	THERINT	THREIF	Write data FIFO
Receive Data Available Interrupt RDAINT	RDAIEN	RDAINT	RDAIF	Read data FIFO

Time Out Register (UARTn_TOUT)

Register	Offset	R/W	Description	Reset Value
UARTn_TOUT	UARTn_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.
[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 is equal to that of time out interrupt comparator (TOIC), a receiver time out interrupt (RXTOINT) is generated if UART_INTEN.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.

Baud Rate Divider Register (UARTn_BAUD)

Register	Offset	R/W	Description	Reset Value
UARTn_BAUD	UARTn_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 5-10 UART Baud Rate Equation.

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.
[29]	**BAUDM1**	**Divider X Enable** 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 5-11 UART Baud Rate Setting Table for more information. NOTE: When in IrDA mode, this bit must disabled.
[28]	**BAUDM0**	**Divider X Equal 1** 0: M = EDIVM1+1, with restriction EDIVM1 ≥ 8. 1: M = 1, with restriction BRD[15:0] ≥ 3. Refer to Table 5-11 UART Baud Rate Setting Table for more information.
[27:24]	**EDIVM1**	**Divider x** The baud rate divider M = EDIVM1+1.
[23:16]	**Reserved**	Reserved.
[15:0]	**BRD**	**Baud Rate Divider** Refer to Table 5-11 UART Baud Rate Setting Table for more information.

IrDA Control Register (UARTn_IRDA)

Register	Offset	R/W	Description	Reset Value
UARTn_IRDA	UARTn_BA+0x28	R/W	UART IrDA 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**RXINV**	**TXINV**	**Reserved**		**LOOPBACK**	**TXEN**	**Reserved**

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**RXINV**	**Receive Inversion Enable** 0= No inversion. 1= Invert Rx input signal.
[5]	**TXINV**	**Transmit Inversion Enable** 0= No inversion. 1= Invert Tx output signal.
[4:3]	**Reserved**	Reserved.
[2]	**LOOPBACK**	**IrDA Loopback Test Mode** Loopback Tx to Rx.
[1]	**TXEN**	**Transmit/Receive Selection** 0=Enable IrDA receiver. 1= Enable IrDA transmitter.
[0]	**Reserved**	Reserved.

UART LIN Network Control Register (UARTn_ALTCTL)

Register	Offset	R/W	Description	Reset Value
UARTn_ALTCTL	UARTn_BA+0x2C	R/W	UART LIN 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**
LINTXEN	**LINRXEN**	**Reserved**		**BRKFL**			

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**LINTXEN**	**LIN TX Break Mode Enable** 0 = Disable LIN Tx Break Mode. 1 = Enable LIN Tx Break Mode. NOTE: When Tx break field transfer operation finished, this bit will be cleared automatically.
[6]	**LINRXEN**	**LIN RX Enable** 0 = Disable LIN Rx mode. 1 = Enable LIN Rx mode.
[5:4]	**Reserved**	Reserved.
[3:0]	**BRKFL**	**UART LIN Break Field Length Count** This field indicates a 4-bit LIN Tx break field count. NOTE: This break field length is BRKFL + 2

UART Function Select Register (UARTn_FUNCSEL)

Register	Offset	R/W	Description	Reset Value
UARTn_FUNCSEL	UARTn_BA+0x30	R/W	UART Function 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						**IRDAEN**	**LINEN**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**IRDAEN**	**Enable IrDA Function** 0 = UART Function. 1 = Enable IrDA Function.
[0]	**LINEN**	**Enable LIN Function** 0 = UART Function. 1 = Enable LIN Function. Note that IrDA and LIN functions are mutually exclusive: both cannot be active at same time.

5.14 I2S Audio PCM Controller

5.14.1 Overview

The I2S controller is a peripheral for serial transmission and reception of audio PCM (Pulse-Code Modulated) signals across a 4-wire bus. The bus consists of a bit clock (I2S_BCLK) a frame synchronization clock (I2S_FS) and serial data in (I2S_SDI) and out (I2S_SDO) lines. This peripheral allows communication with an external audio CODEC or DSP. The peripheral is capable of mono or stereo audio transmission with 8-32bit word sizes. Audio data is buffered in 8 word deep FIFO buffers and has DMA capability.

5.14.2 Features

- I2S can operate as either master or slave
- Master clock generation for slave device synchronization.
- Capable of handling 16, 24 and 32 bit word sizes.
- Mono and stereo audio data supported.
- I2S and MSB justified data format supported.
- 8 word FIFO data buffers for transmit and receive.
- Generates interrupt requests when buffer levels crosses programmable boundary.
- Two DMA requests, one for transmit and one for receive.

5.14.3 I2S Block Diagram

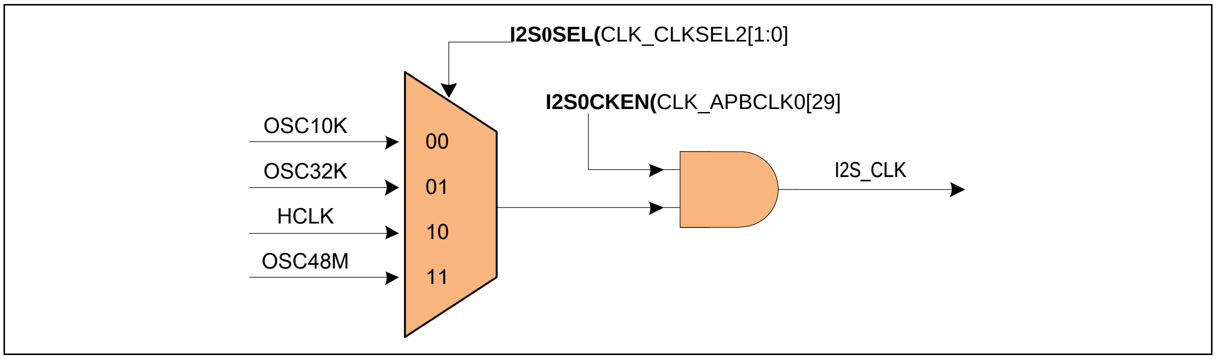


Figure 5-77 I2S Clock Control Diagram

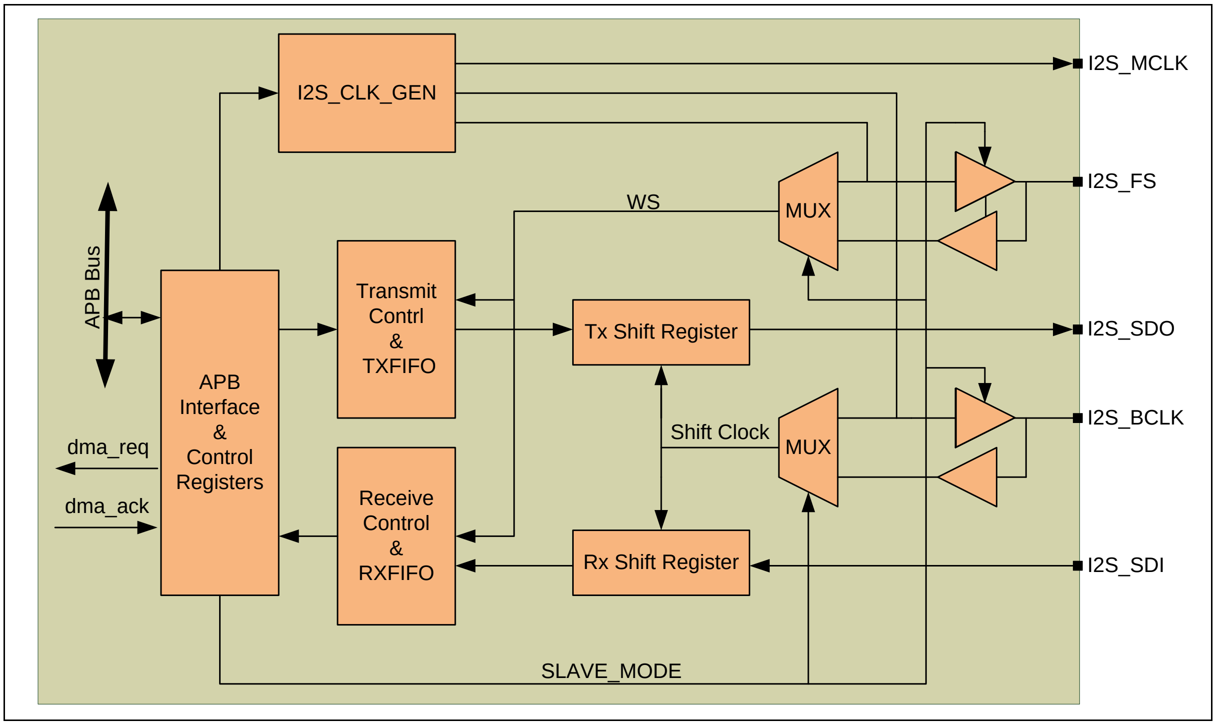


Figure 5-78 I2S Controller Block Diagram

5.14.4 I2S Operation

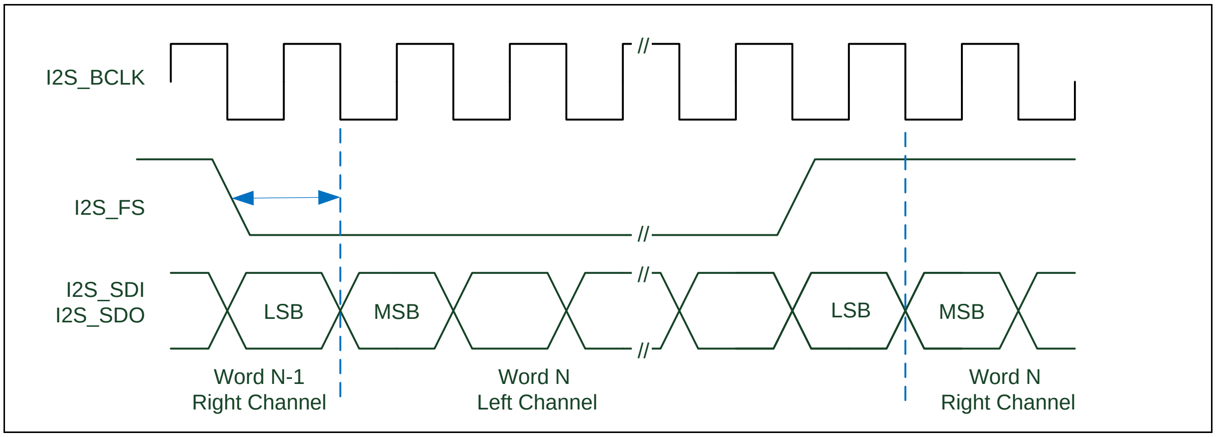


Figure 5-79 I2S Bus Timing Diagram (Format =0)

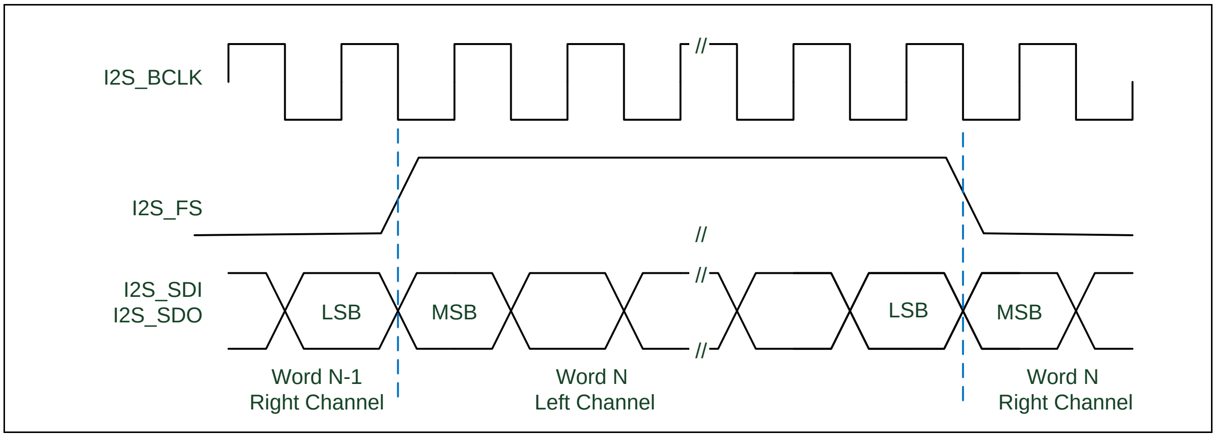


Figure 5-80 MSB Justified Timing Diagram (Format=1)

5.14.5 FIFO operation

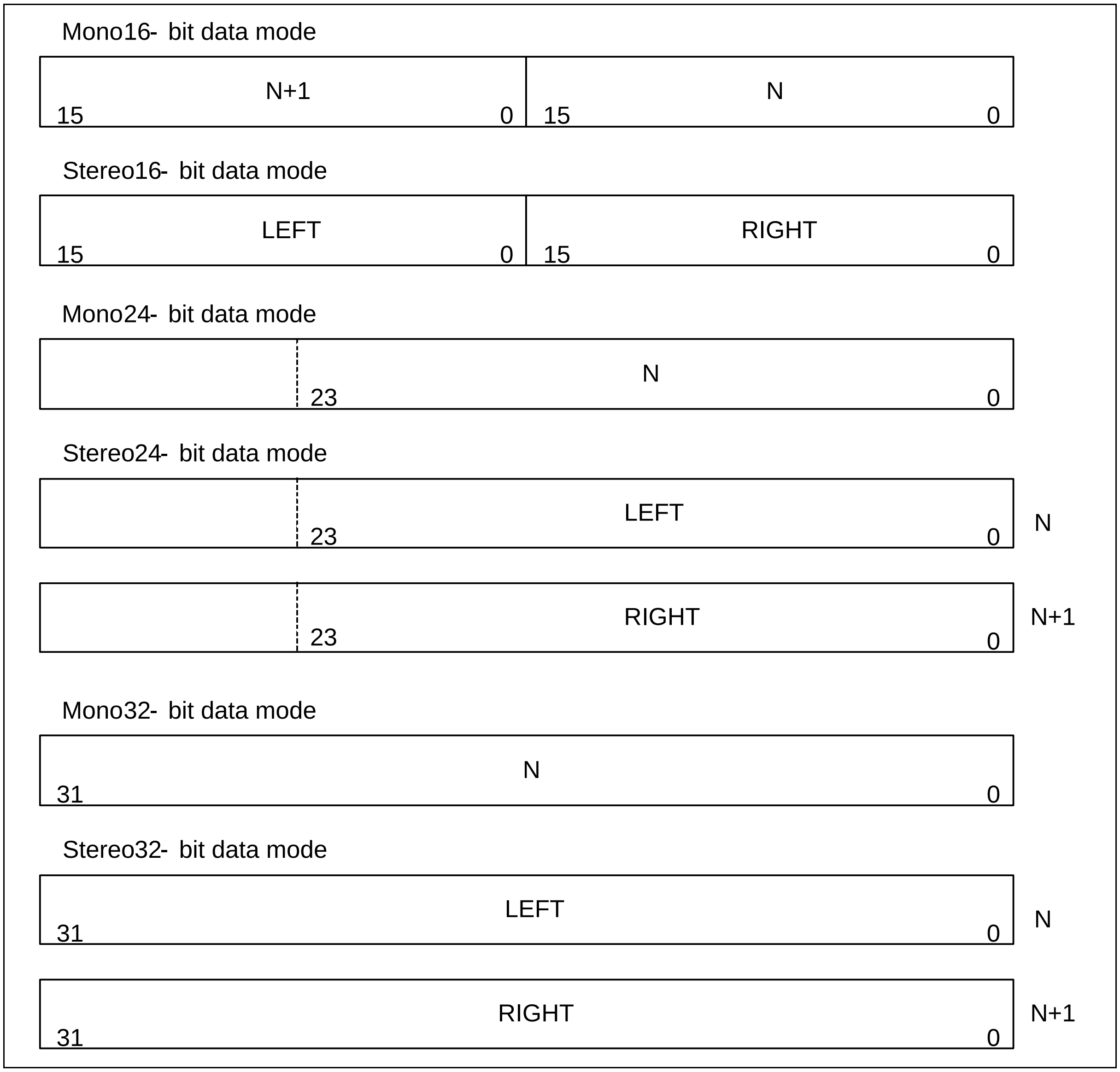


Figure 5-81 FIFO contents for various I2S modes

5.14.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
I2S Base Address: **I2S_BA = 0x400A_0000**				
I2S_CTL	I2S_BA + 0x00	R/W	I2S Control Register	0x0000_0000
I2S_CLK	I2S_BA + 0x04	R/W	I2S Clock Divider Register	0x0000_0000
I2S_IEN	I2S_BA + 0x08	R/W	I2S Interrupt Enable Register	0x0000_0000
I2S_STATUS	I2S_BA + 0x0C	R/W	I2S Status Register	0x0014_1100
I2S_TX	I2S_BA + 0x10	W	I2S Transmit FIFO Register	0xXXXX_XXXX
I2S_RX	I2S_BA + 0x14	R	I2S Receive FIFO Register	0xXXXX_XXXX

5.14.7 Register Description

I2S Control Register (I2S_CTL)

Register	Offset	R/W	Description	Reset Value
I2S_CTL	I2S_BA + 0x00	R/W	I2S Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	Reserved	RXPDMAEN	TXPDMAEN	RXCLR	TXCLR	LZCEN	RZCEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MCLKEN	RXTH			TXTH			SLAVE
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FORMAT	MONO	WDWIDTH		MUTE	RXEN	TXEN	I2SEN

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21]	**RXPDMAEN**	**Enable Receive DMA** When RX DMA is enabled, I2S requests DMA to transfer data from receive FIFO to SRAM if FIFO is not empty. 0 = Disable RX DMA. 1 = Enable RX DMA.
[20]	**TXPDMAEN**	**Enable Transmit DMA** When TX DMA is enables, I2S request DMA to transfer data from SRAM to transmit FIFO if FIFO is not full. 0 = Disable TX DMA. 1 = Enable TX DMA.
[19]	**RXCLR**	**Clear Receive FIFO** Write 1 to clear receive FIFO, internal pointer is reset to FIFO start point, and I2S_STATUS.RXCNT[3:0] returns to zero and receive FIFO becomes empty. This bit is cleared by hardware automatically when clear operation complete.
[18]	**TXCLR**	**Clear Transmit FIFO** Write 1 to clear transmit FIFO, internal pointer is reset to FIFO start point, and I2S_STATUS.TXCNT[3:0] returns to zero and transmit FIFO becomes empty. Data in transmit FIFO is not changed. This bit is cleared by hardware automatically when clear operation complete.
[17]	**LZCEN**	**Left Channel Zero Cross Detect Enable** If this bit is set to 1, when left channel data sign bit changes, or data bits are all zero, the LZCIF flag in I2S_STATUS register will be set to 1. 0 = Disable left channel zero cross detect. 1 = Enable left channel zero cross detect.

[16]	**RZCEN**	**Right Channel Zero Cross Detect Enable** If this bit is set to 1, when right channel data sign bit changes, or data bits are all zero, the RZCIF flag in I2S_STATUS register will be set to 1. 0 = Disable right channel zero cross detect. 1 = Enable right channel zero cross detect.
[15]	**MCLKEN**	**Master Clock Enable** The ISD91200 can generate a master clock signal to an external audio CODEC to synchronize the audio devices. If audio devices are not synchronous, then data will be periodically corrupted. Software needs to implement a way to drop/repeat or interpolate samples in a jitter buffer if devices are not synchronized. The master clock frequency is determined by the I2S_CLKDIV.MCLKDIV[2:0] register. 0 = Disable master clock. 1 = Enable master clock.
[14:12]	**RXTH**	**Receive FIFO Threshold Level** When received data word(s) in buffer is equal or higher than threshold level then RXTHI flag is set. Threshold = RXTH+1 words of data in receive FIFO.
[11:9]	**TXTH**	**Transmit FIFO Threshold Level** If remaining data words in transmit FIFO less than or equal to the threshold level then TXTHI flag is set. Threshold = TXTH words remaining in transmit FIFO.
[8]	**SLAVE**	**Slave Mode** I2S can operate as a master or slave. For master mode, I2S_BCLK and I2S_FS pins are outputs and send bit clock and frame sync from ISD91200. In slave mode, I2S_BCLK and I2S_FS pins are inputs and bit clock and frame sync are received from external audio device. 0 = Master mode. 1 = Slave mode.
[7]	**FORMAT**	**Data Format** 0 = I2S data format. 1 = MSB justified data format. See Figure 5-79 I2S Bus Timing Diagram (Format =0) and Figure 5-80 MSB Justified Timing Diagram (Format=1) for timing differences.
[6]	**MONO**	**Monaural Data** This parameter sets whether mono or stereo data is processed. See Figure 5-81 FIFO contents for various I2S modes for details of how data is formatted in transmit and receive FIFO. 0 = Data is stereo format. 1 = Data is monaural format.
[5:4]	**WDWIDTH**	**Word Width** This parameter sets the word width of audio data. See Figure 5-81 FIFO contents for various I2S modes for details of how data is formatted in transmit and receive FIFO. 00 = data is 8 bit. 01 = data is 16 bit. 10 = data is 24 bit. 11 = data is 32 bit.

[3]	**MUTE**	**Transmit Mute Enable** 0 = Transmit data is shifted from FIFO. 1= Transmit channel zero.
[2]	**RXEN**	**Receive Enable** 0 = Disable data receive. 1 = Enable data receive.
[1]	**TXEN**	**Transmit Enable** 0 = Disable data transmit. 1 = Enable data transmit.
[0]	**I2SEN**	**Enable I2S Controller** 0 = Disable. 1 = Enable.

I2S Clock Divider (I2S_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2S_CLKDIV	I2S_BA + 0x04	R/W	I2S 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**
BCLKDIV							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**MCLKDIV**		

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**BCLKDIV**	**Bit Clock Divider** If I2S operates in master mode, bit clock is provided by ISD91200. Software can program these bits to generate bit clock frequency for the desired sample rate. For sample rate Fs, the desired bit clock frequency is: F_{BCLK} = Fs x Word_width_in_bytes x 16. For example if Fs = 16kHz, and word width is 2-bytes (16bit) then desired bit clock frequency is 512kHz. The bit clock frequency is given by: $F_{BCLK} = F_{I2S\_CLKDIV}$ / (2x (BCLKDIV+1)). Or, BCLKDIV = F_{I2S_CLKDIV} / (2 x F_{BCLK}) -1. So if F_{I2S_CLKDIV} = HCLK = 49.152MHz$F_{I2S\ CLK} = HCLK = 49.152MHz$, desired F_{BCLK} = 512kHz$F_{BCLK} = 512kHz$ then BCLKDIV = 47.$BCLK\_DIV = 47$
[7:3]	**Reserved**	Reserved.
[2:0]	**MCLKDIV**	**Master Clock Divider** ISD91200can generate a master clock to synchronously drive an external audio device. If MCLKDIV is set to 0, MCLK is the same as I2S_CLKDIV clock input, otherwise MCLK frequency is given by: $F_{MCLK} = F_{I2S\_CLKDIV}$ / (2 x MCLKDIV). Or, MCLKDIV = F_{I2S_CLKDIV} / (2 x F_{MCLK}). If the desired MCLK frequency is 4.092MHz (= 256Fs) and Fs = 16kHz then MCLKDIV = 6 @ F_{I2S_CLKDIV} = 49.152MHz.

I2S Interrupt Enable Register (I2S_IEN)

Register	Offset	R/W	Description	Reset Value
I2S_IEN	I2S_BA + 0x08	R/W	I2S 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			**LZCIEN**	**RZCIEN**	**TXTHIEN**	**TXOVIEN**	**TXUDIEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**RXTHIEN**	**RXOVIEN**	**RXUDIEN**

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**LZCIEN**	**Left Channel Zero Cross Interrupt Enable** Interrupt will occur if this bit is set to 1 and left channel has zero cross event 0 = Disable interrupt. 1 = Enable interrupt.
[11]	**RZCIEN**	**Right Channel Zero Cross Interrupt Enable** Interrupt will occur if this bit is set to 1 and right channel has zero cross event 0 = Disable interrupt. 1 = Enable interrupt.
[10]	**TXTHIEN**	**Transmit FIFO Threshold Level Interrupt Enable** Interrupt occurs if this bit is set to 1 and data words in transmit FIFO is less than TXTH[2:0]. 0 = Disable interrupt. 1 = Enable interrupt.
[9]	**TXOVIEN**	**Transmit FIFO Overflow Interrupt Enable** Interrupt occurs if this bit is set to 1 and transmit FIFO overflow flag is set to 1 0 = Disable interrupt. 1 = Enable interrupt.
[8]	**TXUDIEN**	**Transmit FIFO Underflow Interrupt Enable** Interrupt occur if this bit is set to 1 and transmit FIFO underflow flag is set to 1. 0 = Disable interrupt. 1 = Enable interrupt.
[7:3]	**Reserved**	Reserved.

[2]	**RXTHIEN**	**Receive FIFO Threshold Level Interrupt** Interrupt occurs if this bit is set to 1 and data words in receive FIFO is greater than or equal to RXTH[2:0]. 0 = Disable interrupt. 1 = Enable interrupt.
[1]	**RXOVIEN**	**Receive FIFO Overflow Interrupt Enable** 0 = Disable interrupt. 1 = Enable interrupt.
[0]	**RXUDIEN**	**Receive FIFO Underflow Interrupt Enable** If software read receive FIFO when it is empty then RXUDIF flag in I2SSTATUS register is set to 1. 0 = Disable interrupt. 1 = Enable interrupt.

I2S Status Register (I2S_STATUS)

Register	Offset	R/W	Description	Reset Value
I2S_STATUS	I2S_BA + 0x0C	R/W	I2S Status Register	0x0014_1100

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
LZCIF	RZCIF	TXBUSY	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				RIGHT	TXIF	RXIF	I2SIF

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Level (Read Only)** TXCNT = number of words in transmit FIFO.
[27:24]	**RXCNT**	**Receive FIFO Level (Read Only)** RXCNT = number of words in receive FIFO.
[23]	**LZCIF**	**Left Channel Zero Cross Flag (Write ‘1’ to Clear, or Clear LZCEN)** 0 = No zero cross detected. 1 = Left channel zero cross is detected.
[22]	**RZCIF**	**Right Channel Zero Cross Flag (Write ‘1’ to Clear, or Clear RZCEN)** 0 = No zero cross. 1 = Right channel zero cross is detected.
[21]	**TXBUSY**	**Transmit Busy (Read Only)** This bit is cleared when all data in transmit FIFO and Tx shift register is shifted out. It is set when first data is loaded to Tx shift register. 0 = Transmit shift register is empty. 1 = Transmit shift register is busy.
[20]	**TXEMPTY**	**Transmit FIFO Empty (Read Only)** This is set when transmit FIFO is empty. 0 = Not empty. 1 = Empty.
[19]	**TXFULL**	**Transmit FIFO Full (Read Only)** This bit is set when transmit FIFO is full. 0 = Not full. 1 = Full.

[18]	**TXTHIF**	**Transmit FIFO Threshold Flag (Read Only)** When data word(s) in transmit FIFO is less than or equal to the threshold value set in TXTH[2:0] the TXTHIF bit becomes to 1. It remains set until transmit FIFO level is greater than TXTH[2:0]. Cleared by writing to I2S_TX register until threshold exceeded. 0 = Data word(s) in FIFO is greater than threshold level. 1 = Data word(s) in FIFO is less than or equal to threshold level.
[17]	**TXOVIF**	**Transmit FIFO Overflow Flag (Write '1' to Clear)** This flag is set if data is written to transmit FIFO when it is full. 0 = No overflow. 1 = Overflow.
[16]	**TXUDIF**	**Transmit FIFO Underflow Flag (Write '1' to Clear)** This flag is set if I2S controller requests data when transmit FIFO is empty. 0 = No underflow. 1 = Underflow.
[15:13]	**Reserved**	Reserved.
[12]	**RXEMPTY**	**Receive FIFO Empty (Read Only)** This is set when receive FIFO is empty. 0 = Not empty. 1 = Empty.
[11]	**RXFULL**	**Receive FIFO Full (Read Only)** This bit is set when receive FIFO is full. 0 = Not full. 1 = Full.
[10]	**RXTHIF**	**Receive FIFO Threshold Flag (Read Only)** When data word(s) in receive FIFO is greater than or equal to threshold value set in RXTH[2:0] the RXTHIF bit becomes to 1. It remains set until receive FIFO level is less than RXTH[2:0]. It is cleared by reading I2S_RX until threshold satisfied. 0 = Data word(s) in FIFO is less than threshold level. 1 = Data word(s) in FIFO is greater than or equal to threshold level.
[9]	**RXOVIF**	**Receive FIFO Overflow Flag (Write '1' to Clear)** This flag is set if I2S controller writes to receive FIFO when it is full. Audio data is lost. 0 = No overflow. 1 = Overflow.
[8]	**RXUDIF**	**Receive FIFO Underflow Flag (Write '1' to Clear)** This flag is set if attempt is made to read receive FIFO while it is empty. 0 = No underflow. 1 = Underflow.
[7:4]	**Reserved**	Reserved.
[3]	**RIGHT**	**Right Channel Active (Read Only)** This bit indicates current data being transmitted/received belongs to right channel 0 = Left channel. 1 = Right channel.

[2]	**TXIF**	**I2S Transmit Interrupt (Read Only)** This indicates that there is an active transmit interrupt source. This could be TXOVIF, TXUDIF, TXTHIF, LZCIF or RZCIF if corresponding interrupt enable bits are active. To clear interrupt the corresponding source(s) must be cleared. 0 = No transmit interrupt. 1 = Transmit interrupt occurred.
[1]	**RXIF**	**I2S Receive Interrupt (Read Only)** This indicates that there is an active receive interrupt source. This could be RXOVIF, RXUDIF or RXTHIF if corresponding interrupt enable bits are active. To clear interrupt the corresponding source(s) must be cleared. 0 = No receive interrupt. 1 = Receive interrupt occurred.
[0]	**I2SIF**	**I2S Interrupt (Read Only)** This bit is set if any enabled I2S interrupt is active. 0 = No I2S interrupt. 1 = I2S interrupt active.

I2S Transmit FIFO (I2S_TX)

Register	Offset	R/W	Description	Reset Value
I2S_TX	I2S_BA + 0x10	W	I2S Transmit FIFO Register	0xXXXX_XXXX

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**	**Transmit FIFO Register (Write Only)** A write to this register pushes data onto the transmit FIFO. The transmit FIFO is eight words deep. The number of words currently in the FIFO can be determined by reading I2S_STATUS.TXCNT.

I2S Receive FIFO (I2S_RX)

Register	Offset	R/W	Description	Reset Value
I2S_RX	I2S_BA + 0x14	R	I2S Receive FIFO Register	0xXXXX_XXXX

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**	**Receive FIFO Register (Read Only)** A read of this register will pop data from the receive FIFO. The receive FIFO is eight words deep. The number of words currently in the FIFO can be determined by reading I2S_STATUS.RXCNT.

5.15 PDMA Controller

5.15.1 Overview

The ISD91200 incorporates a Peripheral Direct Memory Access (PDMA) controller that transfers data between SRAM and APB devices. The PDMA has four channels of DMA PDMA CH0~CH3). PDMA transfers are unidirectional and can be Peripheral-to-SRAM,SRAM-to-Peripheral or SRAM-to-SRAM.

The peripherals available for PDMA transfer are SPI, UART, I2S, SDADC, SARADC and DPWM.

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 by the transfer size, update by an arbitrary value (span mode) or wrap around a circular buffer. When PDMA operation is complete, controller can be configured to provide CPU with an interrupt.

5.15.2 Features

- Provides access to SPI, UART, I2S, SDADC and DPWM 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, wrap-around and spanned addressing.

5.15.3 Block Diagram

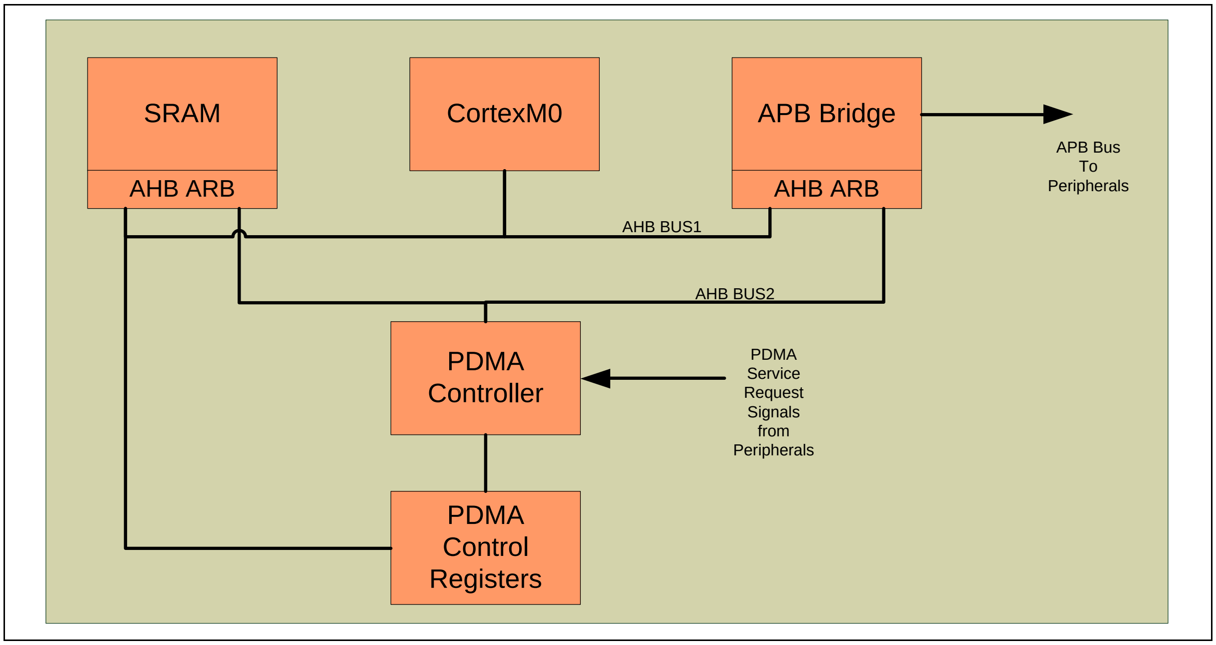


Figure 5-82 PDMA Controller Block Diagram

5.15.4 Function Description

The PDMA controller has four 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 have 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_GCTLn.CHnCKEN**
- Enable PDMA channel *n* by setting **PDMA_CTLn.CHEN**
- Set source address in **PDMA_SADDRn**
- Set destination address in **PDMA_DADDRn**
- Set the transfer count in **PDMA_TXCNTn**
- Set transfer mode and address increment mode in **PDMA_CTLn.MODESEL**
- Route peripheral PDMA request signal to channel *n* in service selection register.
- Trigger transfer **PDMA_CTLn.TXEN**

If the source or destination address is not in wraparound mode, the PDMA will continue the transfer until **PDMA_CURTXCNTn** decrements to zero (**PDMA_CURTXCNTn** is initialized to **PDMA_TXCNTn**, in wraparound mode, **PDMA_CURTXCNTn** will reload and continue until **PDMA_GCTL.CHnCKEN** is disabled). If an error occurs during the PDMA operation, the channel stops until software clears the error condition and sets the **PDMA_CTnL.SWRST** bit to reset the PDMA channel. After reset the **PDMA_CTLn.CHEN** and **PDMA_CTLn.TXEN** bits would need to be set to start a new operation.

5.15.5 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: **PDMAn_BA = 0x5000_8000 +(n*0x100)** **n=0,1,2,3** **PDMA Global Control Base Address:** **PDMA_GCR_BA = 0x5000_8F00**				
PDMA_CTLn	PDMAn_BA+0x00	R/W	PDMA Control Register of Channel n	0x0000_0000
PDMA_SADDRn	PDMAn_BA+0x04	R/W	PDMA Transfer Source Address Register of Channel n	0x4000_0000
PDMA_DADDRn	PDMAn_BA+0x08	R/W	PDMA Transfer Destination Address Register of Channel n	0x4000_0000
PDMA_TXCNTn	PDMAn_BA+0x0C	R/W	PDMA Transfer Byte Count Register of Channel n	0x0000_0000
PDMA_INTPNTn	PDMAn_BA+0x10	R	PDMA Internal Buffer Pointer Register of Channel n	0xXXXX_XX00
PDMA_CURSADDRn	PDMAn_BA+0x14	R	PDMA Current Source Address Register of Channel n	0xFFFF_FFFF
PDMA_CURDADDRn	PDMAn_BA+0x18	R	PDMA Current Destination Address Register of Channel n	0xFFFF_FFFF
PDMA_CURTXCNTn	PDMAn_BA+0x1C	R	PDMA Current Transfer Byte Count Register of Channel n	0x0000_0000
PDMA_INTENn	PDMAn_BA+0x20	R/W	PDMA Interrupt Enable Control Register of Channel n	0x0000_0001
PDMA_INTSTSn	PDMAn_BA+0x24	R/W	PDMA Interrupt Status Register of Channel n	0x0000_0000
PDMA_SPANn	PDMAn_BA+0x34	R	PDMA Span Increment Register of Channel n	0x0000_0000
PDMA_CURSPANn	PDMAn_BA+0x38	R/W	PDMA Current Span Increment Register of Channel n	0x0000_0000
PDMA_GCTL	PDMA_GCR_BA+0x00	R/W	PDMA Global Control Register	0x0000_0000
PDMA_SVCSEL0	PDMA_GCR_BA+0x04	R/W	PDMA Service Selection Control Register 0	0xFFFF_FFFF
PDMA_SVCSEL1	PDMA_GCR_BA+0x08	R/W	PDMA Service Selection Control Register 1	0xFFFF_FFFF
PDMA_GINTSTS	PDMA_GCR_BA+0x0C	R	PDMA Global Interrupt Status Register	0x0000_0000

5.15.6 Register Description

PDMA Control Register (PDMA_CTLn)

Register	Offset	R/W	Description	Reset Value
PDMA_CTLn	PDMAn_BA+0x00	R/W	PDMA Control Register of Channel n	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXEN	Reserved		TXWIDTH		Reserved		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
WAINTSEL				Reserved			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DASEL		SASEL		MODESEL		SWRST	CHEN

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**TXEN**	**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.
[20:19]	**TXWIDTH**	**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.

[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 PDMA_TXCNTn = 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 PDMA_TXCNTn = 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.
[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 PDMA_CURTXCNTn (Current Byte Count) equals zero, the PDMA_CURDADDR (Current Destination Address) and PDMA_CURTXCNTn registers will be reloaded from the PDMA_DADDRn (Destination Address) and PDMA_TXCNTn (Byte Count) registers automatically and PDMA will start another transfer. Cycle continues until software sets PDMA_CTLn.CHEN =0. When PDMA_CTLn.CHEN 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 PDMA_CURTXCNTn (Current Byte Count) equals zero, the PDMA_CURSADDRn (Current Source Address) and PDMA_CURTXCNTn registers will be reloaded from the PDMA_SADDRn (Source Address) and PDMA_TXCNT (Byte Count) registers automatically and PDMA will start another transfer. Cycle continues until software sets PDMA_CTLn.CHEN = 0. When PDMA_CTLn.CHEN 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]	**CHEN**	**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. Note: SWRST will clear this bit.

PDMA Transfer Source Address Register (PDMA_SADDRn)

Register	Offset	R/W	Description	Reset Value
PDMA_SADDRn	PDMAn_BA+0x04	R/W	PDMA Transfer Source Address Register of Channel n	0x4000_0000

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

Bits	Description	
[31:0]	**ADDR**	**PDMA Transfer Source Address Register** This register holds the initial Source Address of PDMA transfer. **Note:** The source address must be word aligned.

PDMA Transfer Destination Address Register (PDMA_DADDRn)

Register	Offset	R/W	Description	Reset Value
PDMA_DADDRn	PDMAn_BA+0x08	R/W	PDMA Transfer Destination Address Register of Channel n	0x4000_0000

Bits	Description	
[31:0]	**ADDR**	**PDMA Transfer Destination Address Register** This register holds the initial Destination Address of PDMA transfer. **Note:** The destination address must be word aligned.

PDMA Transfer Byte Count Register (PDMA_TXCNTn)

Register	Offset	R/W	Description	Reset Value
PDMA_TXCNTn	PDMAn_BA+0x0C	R/W	PDMA Transfer Byte Count Register of Channel n	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.
[15:0]	**CNT**	**PDMA Transfer Byte Count Register** This register controls the transfer byte count of PDMA. Maximum value is 0xFFFF. Note: When in memory-to-memory (TXBCCHn.MODESEL = 00b) mode, the transfer byte count must be word aligned, that is multiples of 4bytes.

PDMA Internal Buffer Pointer Register (PDMA_INTPNTn)

Register	Offset	R/W	Description	Reset Value
PDMA_INTPNTn	PDMAn_BA+0x10	R	PDMA Internal Buffer Pointer Register of Channel n	0xXXXX_XX00

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				**POINTER**			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**POINTER**	**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 Current Source Address Register (PDMA_CURSADDRn))

Register	Offset	R/W	Description	Reset Value
PDMA_CURSADDRn	PDMAn_BA+0x14	R	PDMA Current Source Address Register of Channel n	0xFFFF_FFFF

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

Bits	Description	
[31:0]	**ADDR**	**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 PDMA_SADDRn when PDMA is triggered or when a wraparound occurs.

PDMA Current Destination Address Register (PDMA_CURDADDRn)

Register	Offset	R/W	Description	Reset Value
PDMA_CURDADDRn	PDMAn_BA+0x18	R	PDMA Current Destination Address Register of Channel n	0xFFFF_FFFF

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

Bits	Description	
[31:0]	**ADDR**	**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 PDMA_DADDRn when PDMA is triggered or when a wraparound occurs.

PDMA Current Transfer Byte Count Register (PDMA_CURTXCNTn)

Register	Offset	R/W	Description	Reset Value
PDMA_CURTXCNTn	PDMAn_BA+0x1C	R	PDMA Current Transfer Byte Count Register of Channel n	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.
[15:0]	**CNT**	**PDMA Current Byte Count Register (Read Only)** This field indicates the current remaining byte count of PDMA transfer. This register is initialized with CNT register when PDMA is triggered or when a wraparound occurs

PDMA Interrupt Enable Control Register (PDMA_INTENn)

Register	Offset	R/W	Description	Reset Value
PDMA_INTENn	PDMAn_BA+0x20	R/W	PDMA Interrupt Enable Control Register of Channel n	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.
[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 PDMA_DSCTn_CTL.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 Interrupt Status Register (PDMA_INTSTSn)

Register	Offset	R/W	Description	Reset Value
PDMA_INTSTSn	PDMAn_BA+0x24	R/W	PDMA Interrupt Status Register of Channel n	0x0000_0000

31	30	29	28	27	26	25	24
INTSTS	**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]	**INTSTS**	**Interrupt Pin Status (Read Only)** This bit is the Interrupt pin status of PDMA channel.
[30:12]	**Reserved**	Reserved.
[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 (PDMA_CURTXCNT == 0). 0100 = Current transfer half complete flag (PDMA_CURTXCNT == PDMA_TXCNT /2).
[7:2]	**Reserved**	Reserved.
[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 Span Increment Register (PDMA_SPANn)

Register	Offset	R/W	Description	Reset Value
PDMA_SPANn	PDMAn_BA+0x34	R	PDMA Span Increment Register of Channel n	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**
SPAN							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**SPAN**	**Span Increment Register** This is a signed number in range [-128,127] for use in spanned address mode. If destination or source addressing mode is set as spanned, then this number is added to the address register each transfer. The size of the transfer is determined by the APB_TW setting. Note that span increment must be a multiple of the transfer width otherwise a memory addressing HardFault will occur. Also SPAN may be a negative number.

PDMA Current Span Increment Register (PDMA_CURSPANn)

Register	Offset	R/W	Description	Reset Value
PDMA_CURSPANn	PDMAn_BA+0x38	R/W	PDMA Current Span Increment Register of Channel n	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**
SPAN							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**SPAN**	**Current Span Increment Register** This is a signed read only register for use in spanned address mode. It provides the current address offset from SADDR or DADDR if either is set to span mode.

PDMA Global Control Register (PDMA_GCTL)

Register	Offset	R/W	Description	Reset Value
PDMA_GCTL	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
Reserved				CH3CKEN	CH2CKEN	CH1CKEN	CH0CKEN
7	6	5	4	3	2	1	0
Reserved							SWRST

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**CH3CKEN**	**PDMA Controller Channel 3 Clock Enable Control** 1: Enable Channel 3 clock. 0: Disable Channel 3 clock.
[10]	**CH2CKEN**	**PDMA Controller Channel 2 Clock Enable Control** 1: Enable Channel 2 clock. 0: Disable Channel 2 clock.
[9]	**CH1CKEN**	**PDMA Controller Channel 1 Clock Enable Control** 1: Enable Channel 1 clock. 0: Disable Channel 1 clock.
[8]	**CH0CKEN**	**PDMA Controller Channel 0 Clock Enable Control** 1: Enable Channel 0 clock. 0: Disable Channel 0 clock.
[7:1]	**Reserved**	Reserved.
[0]	**SWRST**	**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 not be cleared. This bit will auto clear after several clock cycles. Note: This bit can reset all channels (global reset).

PDMA Service Selection Control Register 0 (PDMA_SVCSEL0)

Register	Offset	R/W	Description	Reset Value
PDMA_SVCSEL0	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 3 is to be used to transfer data from memory to DPWM peripheral, then DPWMTXSEL should be set to 3. This will route the DPWM transmit request signal to PDMA channel 3, whenever DPWM has space in FIFO it will request transmission of data from PDMA. When not used the selection should be set to 0xFF.

31	30	29	28	27	26	25	24
I2STXSEL				I2SRXSEL			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
UART0XSEL				UART0RXSEL			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DPWMTXSEL				SDADCRXSEL			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPI0TXSEL				SPI0RXSEL			

Bits	Description	
[31:28]	**I2STXSEL**	**PDMA I2S Transmit Selection** This field defines which PDMA channel is connected to I2S peripheral transmit (PDMA destination) request.
[27:24]	**I2SRXSEL**	**PDMA I2S Receive Selection** This field defines which PDMA channel is connected to I2S peripheral receive (PDMA source) request.
[23:20]	**UART0XSEL**	**PDMA UART0 Transmit Selection** This field defines which PDMA channel is connected to UART0 peripheral transmit (PDMA destination) request.
[19:16]	**UART0RXSEL**	**PDMA UART0 Receive Selection** This field defines which PDMA channel is connected to UART0 peripheral receive (PDMA source) request.
[15:12]	**DPWMTXSEL**	**PDMA DPWM Transmit Selection** This field defines which PDMA channel is connected to DPWM peripheral transmit (PDMA destination) request.
[11:8]	**SDADCRXSEL**	**PDMA SDADC Receive Selection** This field defines which PDMA channel is connected to SDADC peripheral receive (PDMA source) request.
[7:4]	**SPI0TXSEL**	**PDMA SPI0 Transmit Selection** This field defines which PDMA channel is connected to SPI0 peripheral transmit (PDMA destination) request.

[3:0]	**SPI0RXSEL**	**PDMA SPI0 Receive Selection** This field defines which PDMA channel is connected to SPI0 peripheral receive (PDMA source) request.

PDMA Service Selection Control Register 1 (PDMA_SVCSEL1)

Register	Offset	R/W	Description	Reset Value
PDMA_SVCSEL1	PDMA_GCR_BA+0x08	R/W	PDMA Service Selection Control Register 1	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 3 is to be used to transfer data from memory to DPWM peripheral, then DPWMTXSEL should be set to 3. This will route the DPWM transmit request signal to PDMA channel 3, whenever DPWM has space in FIFO it will request transmission of data from PDMA. When not used the selection should be set to 0xFF.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				SARADCRXSEL			
15	14	13	12	11	10	9	8
SPI1TXSEL				SPI1RXSEL			
7	6	5	4	3	2	1	0
UART1XSEL				UART1RXSEL			

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**SARADCRXSEL**	**PDMA SARADC Receive Selection** This field defines which PDMA channel is connected to SARADC peripheral receive (PDMA source) request.
[15:12]	**SPI1TXSEL**	**PDMA SPI1 Transmit Selection** This field defines which PDMA channel is connected to SPI1 peripheral transmit (PDMA destination) request.
[11:8]	**SPI1RXSEL**	**PDMA SPI1 Receive Selection** This field defines which PDMA channel is connected to SPI1 peripheral receive (PDMA source) request.
[7:4]	**UART1XSEL**	**PDMA UART1 Transmit Selection** This field defines which PDMA channel is connected to UART1 peripheral transmit (PDMA destination) request.
[3:0]	**UART1RXSEL**	**PDMA UART1 Receive Selection** This field defines which PDMA channel is connected to UART1 peripheral receive (PDMA source) request.

PDMA Global Interrupt Status Register (PDMA_GINTSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_GINTSTS	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**
Reserved				**CH3INTSTS**	**CH2INTSTS**	**CH1INTSTS**	**CH0INTSTS**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**CH3INTSTS**	**Interrupt Pin Status of Channel 3 (Read Only)** This bit is the interrupt pin status of PDMA channel 3.
[2]	**CH2INTSTS**	**Interrupt Pin Status of Channel 2 (Read Only)** This bit is the interrupt pin status of PDMA channel 2.
[1]	**CH1INTSTS**	**Interrupt Pin Status of Channel 1 (Read Only)** This bit is the interrupt pin status of PDMA channel 1.
[0]	**CH0INTSTS**	**Interrupt Pin Status of Channel 0 (Read Only)** This bit is the interrupt pin status of PDMA channel 0.

5.16 Volume Control

5.16.1 Overview and feature

The volume control function is digital domain gain control and supports both side of SDADC and DPWM. The audio signal can be changed from 36 dB to -108dB if using this feature.

The volume control is enabled by setting **SDADCVOLEN** and DPWMVOL_EN. The volume control shares a clock source with the BIQ filter so CLK_APBCLK0.BIQALCEN, also the volume value will multiply BIQ result so BIQ_**DLCOEFF** must be set to operate volume control.

5.16.2 Register Map

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

Register	Offset	R/W	Description	Reset Value
VOLCTRL Base Address: **VOLCTRL_BA = 0x400B_00A0**				
VOLCTRL_EN	VOLCTRL_BA+0x00	R/W	Volume Control Enable Register	0x0000_0000
VOLCTRL_ADCVAL	VOLCTRL_BA+0x04	R/W	ADC Volume Control Value	0x0004_0000
VOLCTRL_DPWMVAL	VOLCTRL_BA+0x08	R/W	DPWM Volume Control Value	0x0004_0000

5.16.3 Register Description

Volume Control Enable Register (VOLCTRL_EN)

Register	Offset	R/W	Description	Reset Value
VOLCTRL_EN	VOLCTRL_BA+0x00	R/W	Volume Control 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				DPWMZCEN	SDADCZCEN	DPWMVOLEN	SDADCVOLEN

Bits	Description	
[31:4]	**Reserved**	**Reserved**
[3]	**DPWMZCEN**	**DPWM Audio Signal Volume Zero Crossing Enable** 0 = disable zero crossing update gain. 1 = enable Zero crossing update gain.
[2]	**SDADCZCEN**	**Delta-Sigma ADC Signal Volume Zero Crossing Enable** 0 = disable zero crossing update gain. 1 = enable Zero crossing update gain.
[1]	**DPWMVOLEN**	**DPWM Audio Signal Volume Control Enable** 0 = bypass the volume control function. 1 = enable the volume control function.
[0]	**SDADCVOLEN**	**Delta-Sigma ADC Signal Volume Control Enable** 0 = bypass the volume control function. 1 = enable the volume control function.

ADC Volume Control Value (VOLCTRL_ADCVAL)

Register	Offset	R/W	Description	Reset Value
VOLCTRL_ADCVAL	VOLCTRL_BA+0x04	R/W	ADC Volume Control Value	0x0004_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**VALUE**	**Delta-Sigma ADC Signal Volume Control Value** Format <6,18>, gain range from -108.3dB to 36.1dB 0x00_0001 --- -108.3dB … 0x04_0000 ---- 0dB(default) … 0xCC_CCCC ---- 34.1dB Others reserved Volume db = 20*log10(VALUE)

DAC Volume Control Value (VOLCTRL_DPWMVAL)

Register	Offset	R/W	Description	Reset Value
VOLCTRL_DPWMVAL	VOLCTRL_BA+0x08	R/W	DPWM Volume Control Value	0x0004_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**VALUE**	**DPWM Audio Signal Volume Control Value** Format <6,18>. Gain range from 108.3dB to 36.1dB 0x00_0001 ---- -108.3dB … 0x40_0000 ---- 0dB (default) … 0xFF_FFFF ---- 36.1dB Volume db = 20*log10(VALUE)

5.17 Flash Memory Controller (FMC)

5.17.1 Overview

The ISD91200 series is available with 64K/128K bytes 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 enable 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 4K 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 page address boundary. Erasable page size is 512 Byte. This boundary is also specified in the CONFIG memory.

5.17.2 Features

- AHB interface compatible
- Runs up to 49 MHz with zero wait-state for continuous address read access
- Mini-cache to reduce flash access and power consumption.
- 128/64KB application program memory (APROM)
- 4KB in system programming (ISP) boot loader program memory (LDROM)
- Configurable data flash with 512 Bytes page erase unit
- Programmable data flash start address.
- In System Program (ISP) capability to update on chip Flash EEPROM

5.17.3 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:

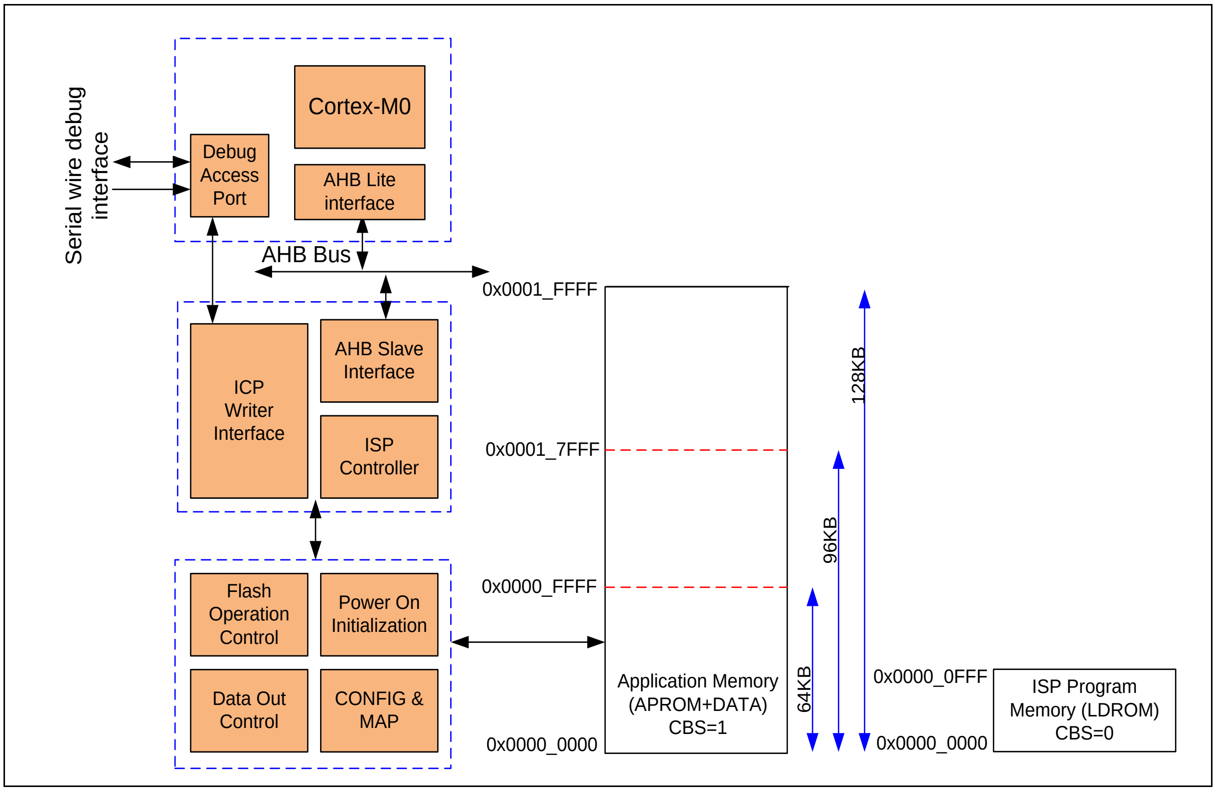


Figure 5-83 Flash Memory Control Block Diagram

5.17.4 Flash Memory Organization

The ISD91200 flash memory consists of Application Program (APROM) memory (128/64KB), data flash (DATAF), ISP boot loader (LDROM) program memory (4KB), 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. 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 (FMC_DFBA) is defined in CONFIG memory and determines the relative size of the APROM and DATAF partitions.

Table 5-14 Memory Address Map

Block Name	Size	Start Address	End Address
APROM	128 KB	0x0000_0000	0x0001_FFFF (128KB) DFBA-1 if DFEN!=0
DATAF	User Configurable	DFBA	0x0001_FFFF (128KB)
LDROM	4 KB	0x0010_0000	0x0010_0FFF
CONFIG	8B	0x0030_0000	0x0030_0007

The Flash memory organization is shown as below:

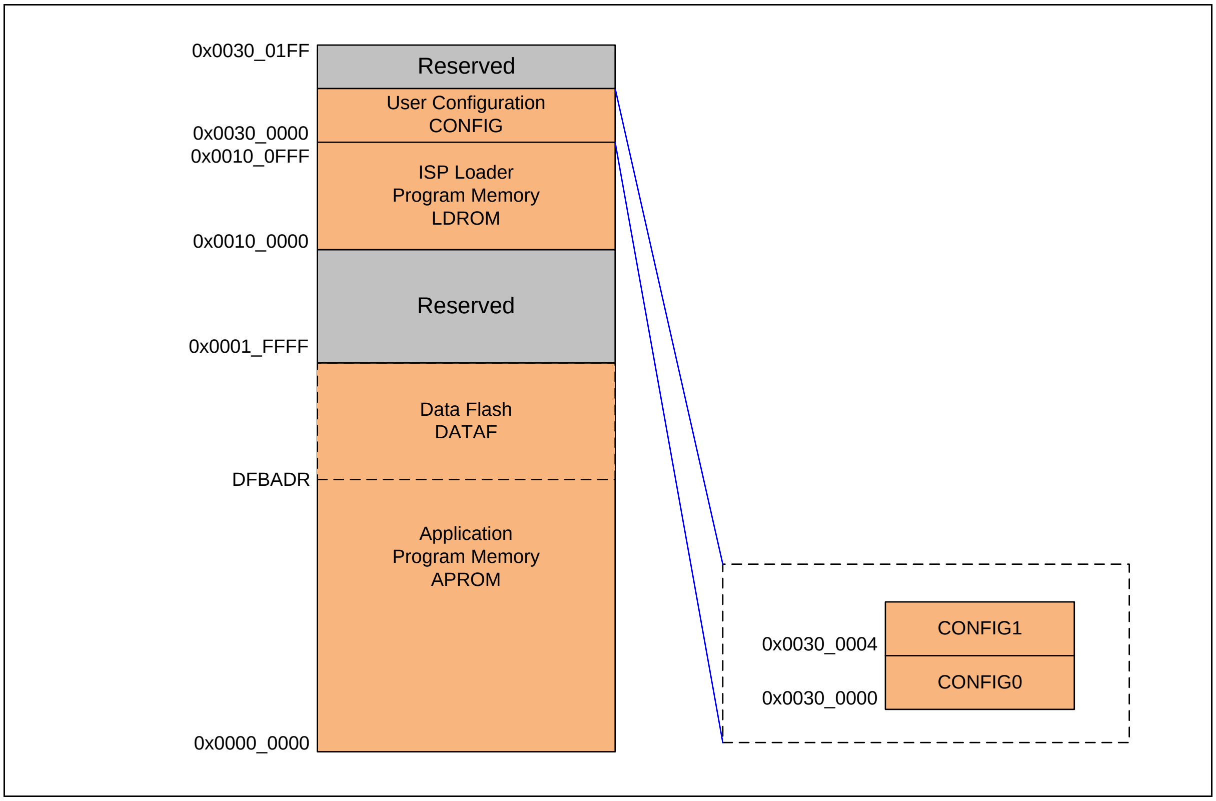


Figure 5-84 Flash Memory Organization

5.17.5 Boot Selection

The ISD91200 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 4KBboot 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 or I2C to fetch new application code. The memory area from which the ISD91200 boots is controlled by the CBS bit in Config0 register.

5.17.6 Data Flash (DATAF)

The ISD91200 provides a data flash partition for user to store non-volatile data such as audio recordings. It accessed through ISP procedures via the Flash Memory Controller (FMC). The size of each erasable page is 512 byte and minimum write size is one word (4Bytes).An erase operation resets all memory in page to value 0xFF. A write operation can only change a ‘1’ bit to a ‘0’ bit. If a subset of the page needs to be changed, the entire 512B page must be copied to another page or into SRAM in advance as entire page must be erased before modification. Data flash and application program memory share the same memory space. If DFENB bit in Config0 is enabled (‘0’), the data flash base address is defined by DFBA and application program memory size is (X-N)KB and data flash size is N KB, where X is the total device memory size (128/64) and N is number of Kbytes reserved for data flash.

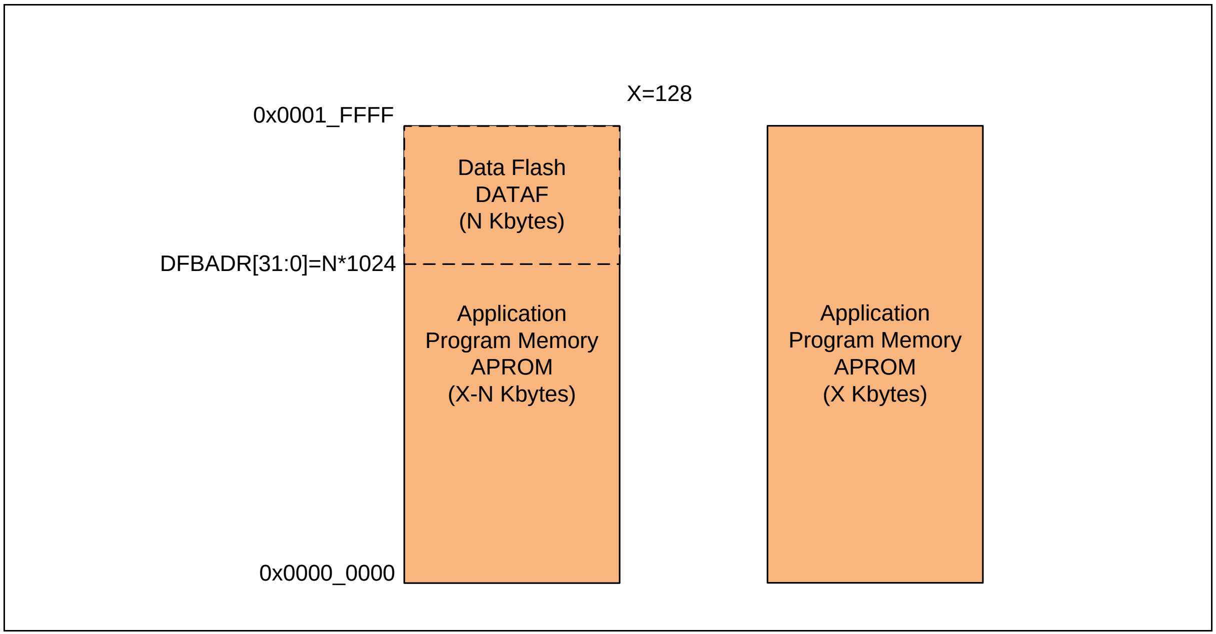


Figure 5-85 Flash Memory Structure

5.17.7 User Configuration (CONFIG)

CONFIG0 (ISP Address = 0x0030_0000)

31	30	29	28	27	26	25	24
Reserved				CLVR	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
CBS	Reserved					LOCK	DFEN

Config0	Address = 0x0030_0000	
Bits	Description	
[31:28]	Reserved	Reserved for future use
[27]	CLVR	**Configuration LVR** 0 = LVR enabled after power on or chip reset 1 = LVR disabled after power on or chip reset
[26:8]	Reserved	Reserved for future use
[7]	CBS	**Configuration Boot Selection** 0 = Chip will boot from LDROM 1 = Chip will boot from APROM
[6:2]	Reserved	Reserved for future use
[1]	LOCK	**Security Lock** 0 = Flash data is locked 1 = Flash data is not locked. When flash data is locked, only device ID, Config0 and Config1 can be read by ICP through serial debug interface. Other data is locked as 0xFFFFFFFF. Once locked no SWD debugging is possible. ISP can read data anywhere regardless of LOCK bit value.
[0]	DFENB	**Data Flash Enable Bar** When data flash is enabled, flash memory is partitioned between APROM and DATAF memory depending on the setting of data flash base address in Config1 register. If set to ‘0’ then no DATAF partition exists. 0 = Enable data flash 1 = Disable data flash

CONFIG1 (Address = 0x0030_0004)

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

Config1	Address = 0x0030_0004	
Bits	**Description**	
[31:20]	**Reserved**	Reserved It is mandatory to program 0x00 to these Reserved bits
[19:0]	**DFBA**	**Data Flash Base Address** This pointer sets the address for the start of data flash memory. Address must be on a 512 Byte page boundary so DFBA[8:0] must be 0x000.

5.17.8 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 ISD91200 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.

5.17.9 ISP Procedure

The ISD91200 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 FMC_ISPCTL.BS can be set to ’1’ and a software reset issued. This will cause the chip to boot from LDROM. An example flow diagram of the ISP sequence is shown in Figure 6-5.

The FMC_ISPCTL register is a protected register, user must first follow the unlock sequence 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 (CLK_AHBCLK.ISPCKEN) is present then set the FMC_ISPCTL.ISPEN bit.

Several error conditions are checked after software writes the ISPTRIG register. If an error condition occurs, ISP operation is not started and the ISP fail flag (FMC_ISPCTL.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 ISPTRIG 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.

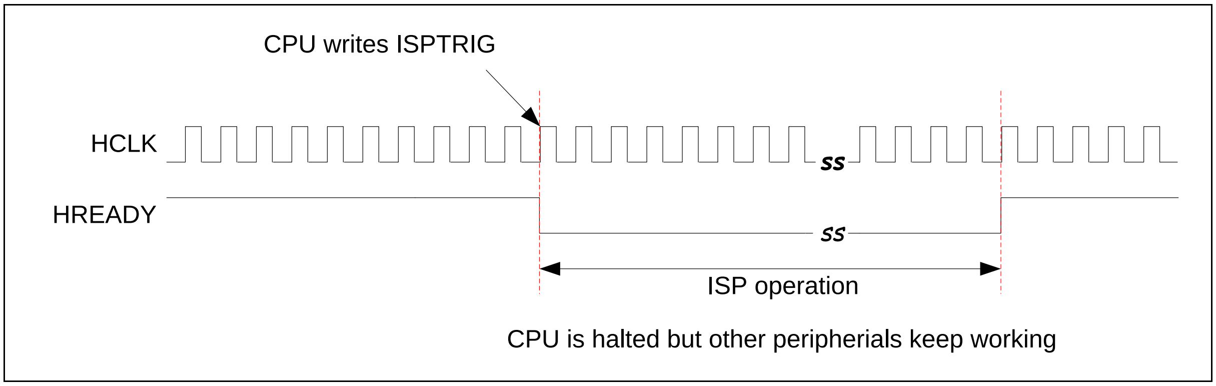


Figure 5-86 ISP Operation Timing

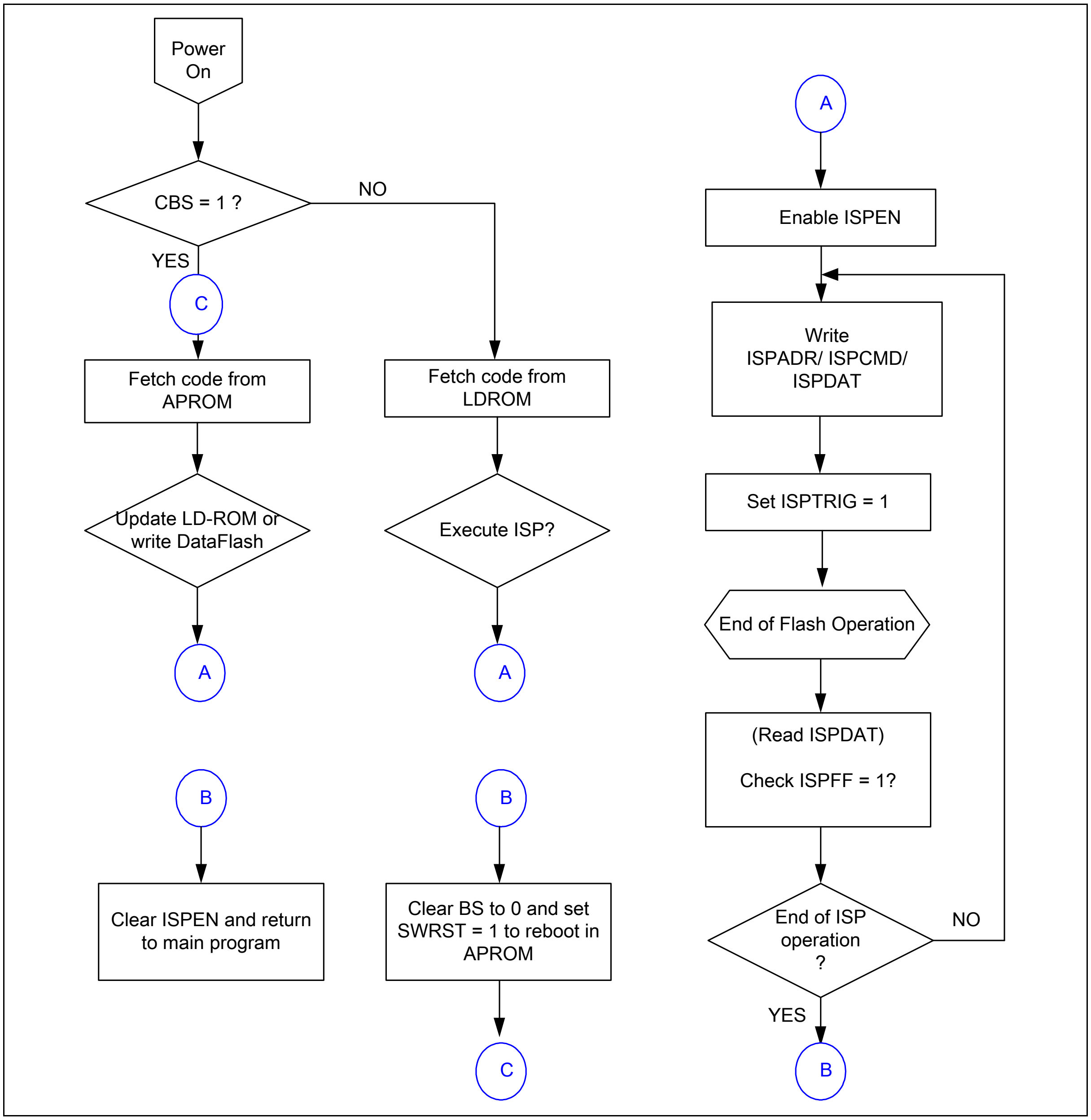


Figure 5-87 Boot Sequence and ISP Procedure

The ISP command set is shown in Table 5-15. Three registers determine the action of a command: FMC_ISPCMD is the command register and accepts commands for reading ID registers and read/write/erase of flash memory. The FMC_ISPADDR is the address register where the flash memory address for access is written. FMC_ISPDAT is the data register that input data is written to and return data read from. An ISP command is executed by setting FMC_ISPCMD, FMC_ISPDAT and FMC_ISPADDR then writing to the trigger register ISPTRIG.

Table 5-15 ISP Command Set

ISP Mode	FMC_ISPCMD	FMC_ISPADDR			FMC_ISPDAT
	CMD[5:0]	A21	A20	A[19:0]	D[31:0]
Standby	0x3x	x	x	x	x
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

5.17.10 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_ISPADDR	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_DFBA	FMC_BA+0x14	R	Data Flash Base Address	0xXXXX_XXXX

5.17.11 Register Description

ISP Control Register (FMC_ISPCTL)

The FMC_ISPCTL register is a protected register, user must first follow the unlock sequence 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		**CACHEDIS**	**Reserved**				
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.
[21]	**CACHEDIS**	**Cache Disable** When set to 1, caching of flash memory reads is disabled.
[18:8]	**Reserved**	Reserved.
[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. 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** 1 = APROM write to itself. 0 = APROM can’t write itself. ISPFF with “1”
[2]	**Reserved**	Reserved.
[1]	**BS**	**Boot Select** 0 = APROM. 1 = LDROM. Modify this bit to select which ROM next boot is to occur. This bit also 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_ISPADDR)

Register	Offset	R/W	Description	Reset Value
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**ISPADDR**	**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 ISPADDR [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							
23	22	21	20	19	18	17	16
ISPDAT							
15	14	13	12	11	10	9	8
ISPDAT							
7	6	5	4	3	2	1	0
ISPDAT							

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		**CMD**					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**CMD**	**ISP Command** Operation Mode : CMD Standby : 0x3X Read : 0x00 Program : 0x21 Page Erase : 0x22 Read CID : 0x0B Read DID : 0x0C

ISP Trigger Control Register (FMC_ISPTRG)

The FMC_ISPTRG register is a protected register, user must first follow the unlock sequence 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.
[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 to gain access.

Data Flash Base Address Register (FMC_DFBA)

Register	Offset	R/W	Description	Reset Value
FMC_DFBA	FMC_BA+0x14	R	Data Flash Base Address	0xXXXX_XXXX

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

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.

5.18 Sigma- Delta Analog-to-Digital Converter (SDADC)

5.18.1 Functional Description

The ISD91200 includes a Sigma-Delta 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 64/128/192 and 384. The decimation input is 6.144 MHz oversample rate. The SINC filter and low pass filter can down sample up to 64000. Low pass filter is IIR filter implemented by BIQ filter.

Note: No SDADC function in ISD91200B Bridge Sense series, but FIFO and interrupt are used for Bridge Sense function.

5.18.2 Features

- Programmable volume control from -108dB to 36dB in 0.5dB steps.
- Support Automatic Level Controller (ALC) function.
- Configurable down-sampling to support sample rate 64KHz.
- Programmable Bi-quad filter to support multiple sample rate 64KHz.
- DMA support for minimal CPU intervention.

5.18.3 Block Diagram

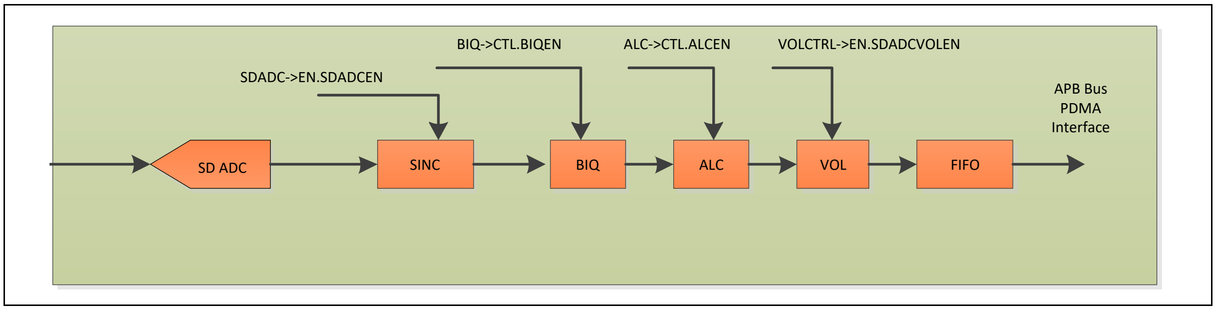


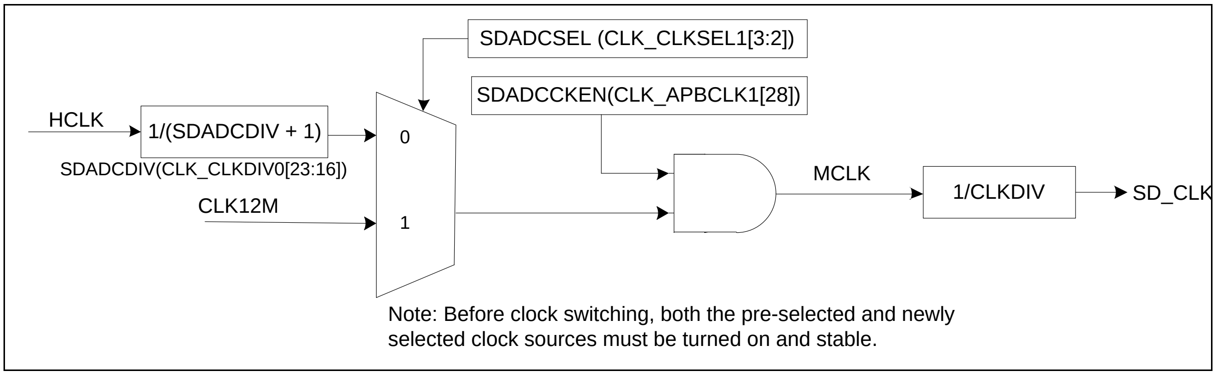
Figure 5-88 ADC Signal Path Block Diagram

5.18.4 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.

5.18.4.1 SDADC Clock Generator


SDADCSEL (CLK_CLKSEL1[3:2])
SDADCCKEN(CLK_APBCLK1[28])
HCLK
1/(SDADCDIV + 1)
SDADCDIV(CLK_CLKDIV0[23:16])
CLK12M
0
1
MCLK
1/CLKDIV
SD_CLK

Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

5.18.4.2 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, minimum 1.024MHz.

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 SDADC)

DSR = SDADC_CTL.DSRATE (when BIQ is on DPWM or BIQ off)

DSRATE and SDADCWNSR are the filter down sampling rates to expect sampling rate.

Table 5-17 Sample Rates for MCLK 24.576MHz (when BIQ is on SDADC)

Fs	DSR	SD_CLK	CLKDIV	SDADC_CTL.DSRATE	BIQ_CTL.SDADCWNSR
48KHz	128	6.144MHz	4	64	2
			4	32	4
	64	3.072MHz	8	32	2
			8	16	4
			8	64	1
32KHz	192	6.144MHz	4	64	3
	128	4.096MHz	6	64	2
				32	4
	64	2.048MHz	12	32	2
				16	4
				64	1
16KHz	384	6.166MHz	4	64	6
	256	4.096MHz	6	64	4
	128	2.048MHz	12	64	2
				32	4
	64	1.024MHz	24	32	2
				16	4
				64	1
8KHz	384	3.072MHz	8	64	6
	256	2.048MHz	12	64	4
	128	1.024MHz	24	64	2
				32	4
	64	512KHz	48	32	2
				16	4
				64	1

Table 5-18 Sample Rates for MCLK 12.288MHz (when BIQ is on SDADC)

Fs	DSR	SD_CLK	CLKDIV	SDADC_CTL.DSRATE	BIQ_CTL.SDADCWNSR
48KHz	64	3.072MHz	4	32	2
				16	4
				64	1
32KHz	64	2.048MHz	6	32	2
				16	4
				64	1
16KHz	128	2.048MHz	6	64	2
				32	4
	64	1.024MHz	12	32	2
				16	4
				64	1
8KHz	384	3.072MHz	4	64	6
	256	2.048MHz	6	64	4
	128	1.024MHz	12	64	2
				32	4
	64	512KHz	24	32	2
				16	4
				64	1

Table 5-19 Sample Rates for SINC (for I91200B series)

CHIP Decimator Output Sample Rate	SINC down factor	Low pass filter BIQ	BS sample rate	Software down sample (get one out of 40/20/10/5/8)
384Hz	16000 (BSRATE =3)	10th order, 5 stage low_decm384_3.txt	9.6Hz	384/9.6 = 40
384Hz	16000 (BSRATE =3)	10th order, 5 stage low_decm384_6.txt	19.2Hz	384/19.2 = 20
384Hz	16000 (BSRATE =3)	10th order, 5 stage low_decm384_12.txt	38.4Hz	384/38.4 =10
384Hz	16000 (BSRATE =3)	10th order, 5 stage low_decm384_24.txt	76.8Hz	384/76.8 = 5
3072Hz	2000 (BS_OSR=0)	10th order, 5 stage low_decm3072_120.txt	384Hz	3072/384=8

Table 5-20 Sample Rates and cut-off frequency for BIQ (for I91200B series)

Sample rate (SPS)	-3dB Cut-off frequency(Hz)	FIFO out sample rate(Hz)
9.6	3	384
19.2	6	384
38.4	12	384
76.8	24	384
384	120	3072

Table 5-21 Low pass filter coefficient table

low_decm384_3	low_decm384_6	low_decm384_12	low_decm384_24	low_decm3072_120
00ae7;//0.042480	00ae8;//0.042599	00b41;//0.043957	00d0e;//0.050989	044f2;//0.268669
7ea6d;//-0.084068	7eb09;//-0.081892	7ecce;//-0.074978	7f2c8;//-0.051643	07587;//0.457977
00ae7;//0.042480	00ae8;//0.042599	00b41;//0.043957	00d0e;//0.050989	044f2;//0.268669
10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000
6061e;//-1.976105	60c6b;//-1.951500	61b10;//-1.894287	63ef0;//-1.754158	06151;//0.380135
0fa45;//0.977615	0f505;//0.957108	0ea8d;//0.916214	0d6e9;//0.839493	0a46c;//0.642273
00070;//0.001709	0008a;//0.002111	000f5;//0.003744	00298;//0.010124	062df;//0.385269
7ff49;//-0.002789	7ff84;//-0.001886	00068;//0.001579	003c5;//0.014725	0c107;//0.752182
00070;//0.001709	0008a;//0.002111	000f5;//0.003744	00298;//0.010124	062df;//0.385269
10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000
608e1;//-1.965317	6111b;//-1.933189	621d5;//-1.867851	642ee;//-1.738556	79038;//-0.436653
0f738;//0.965691	0ef40;//0.934570	0df8a;//0.873199	0c25a;//0.759186	03583;//0.209030
0348c;//0.204758	0349e;//0.205538	035ad;//0.209667	03a5c;//0.227968	06036;//0.374916
79781;//-0.407187	798fe;//-0.402370	79d7f;//-0.384785	7ae6f;//-0.318617	094cb;//0.579803
0348c;//0.204758	0349e;//0.205538	035ad;//0.209667	03a5c;//0.227968	06036;//0.374916
10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000
601cf;//-1.992943	604a8;//-1.981812	60e05;//-1.945244	62ec0;//-1.817383	0cae6;//0.792564
0fed7;//0.995461	0fdc2;//0.991234	0fb8f;//0.982643	0f74f;//0.966042	0eff5;//0.937325
004b1;//0.018282	004c5;//0.018627	0053f;//0.020485	00754;//0.028631	08ead;//0.555977
7f6d9;//-	7f74f;//-	7f8ce;//-	7fe14;//-	105f2;//1.020729

0.035671	0.033953	0.028109	0.007509	
004b1;//0.018282	004c5;//0.018627	0053f;//0.020485	00754;//0.028631	08ead;//0.555977
10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000
607db;//-1.969315	60f63;//-1.939896	61f6d;//-1.877251	641d4;//-1.742859	7fde9;//-0.008163
0f85b;//0.970139	0f165;//0.942947	0e39e;//0.889130	0c9f0;//0.788818	06e5b;//0.431068
02826;//0.156454	0281d;//0.156698	028df;//0.159651	02cdd;//0.175240	07ade;//0.478783
7b040;//-0.310770	7b1d0;//-0.305422	7b64d;//-0.287893	7c5d7;//-0.227188	07ade;//0.478783
02826;//0.156454	0281d;//0.156698	028df;//0.159651	02cdd;//0.175240	00000;//0.000000
10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000	10000;//1.000000
60408;//-1.984261	608be;//-1.965858	61539;//-1.917107	63912;//-1.777069	7ada2;//-0.321747
0fc85;//0.986404	0f94e;//0.973846	0f2d7;//0.948586	0e691;//0.900642	00000;//0.000000

5.18.4.3 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 and SDADC Volume control:

- 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 1 or 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 FEPGA and MICBIAS.

Setup sample rate based on current MCLK(in

- Table 5-17 & Table 5-18) frequency and Set SDADC_CLKDIV.
- Setup FIFO data width SDADC_CTL.FIFOBITS

Setup down ampling rate, set SDADC_CTL.RATESEL=0 then set SDADC_CTL.DSRATE & BIQ_CTL.SDADCWNSR for expected DSR & sampling rate (refer to

- Table 5-17)
- 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 and SDADC Volume control:

- 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 1 or 2 RC time.
- Enable SDADC clock source (CLK_APBCLK0.SDADCCKEN, CLKSEL1.SDADCSEL).
- Reset SDADC block. (SYS_IPRST1.EADCRST).
- Enable SDADC power (SDCHOP.SDADC_PD = 0)
- Power up FEPGA and MICBIAS.

- Setup sample rate based on current MCLK frequency and Set SDADC_CLKDIV
- Setup FIFO data width SDADC_CTL.FIFOBITS

Setup down ampling rate, set SDADC_CTL.RATESEL=0 then set SDADC_CTL.DSRATE & BIQ_CTL.SDADCWNSR for expected DSR & sampling rate (refer to

- Table 5-17).
- 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(BIQ_COEFF)
- Setup SDADC volume control (VOLCTRL_EN.SDADCVOLEN,VOLCTRL_ADCVAL). (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).

5.18.4.4 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.

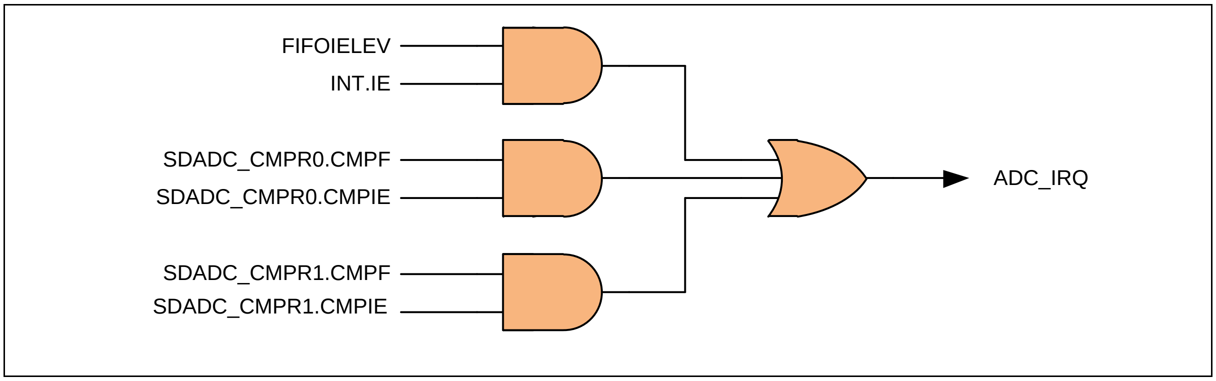


Figure 5-89 SDADC Controller Interrupt

5.18.4.5 Peripheral DMA Request

Normal use of the SDADC is with PDMA. In this mode ADC 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.

5.18.5 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
SDADC Base Address: **SDADC_BA = 0x400E_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_SDCHOP	SDADC_BA+0x20	R/W	Sigma Delta Analog Block Control Register	0x001c_1021

5.18.6 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							
23	22	21	20	19	18	17	16
RESULT							
15	14	13	12	11	10	9	8
RESULT							
7	6	5	4	3	2	1	0
RESULT							

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 to determine if valid data is present in FIFO. Note 1: Each FIFO read is 32 bits including PDMA access. Each FIFO contains one audio sample data, no matter FIFOBITS setting. Note 2: Maximum 8 levels of FIFO available.

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.
[1]	**DINEDGE**	**ADC data input clock edge selection** (for DMIC only, default 0 for AMIC) 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							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**CLKDIV**	**SDADC Clock Divider** SDADC internal clock divider. CLKDIV should be set to give a SD_CLK frequency in the range of 1.024-6.144MHz. (Refer to section 5.18.4.2.) CLKDIV = MCLK/SD_CLK (CLKDIV should ≧ 4, refer to section 5.18.4.1)

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				**RATESEL**	**BSRATE**		**DMICEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
THIE	**FIFOTH**			**FIFOBITS**		**DSRATE**	

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**RATESEL**	**Sample Rate Selection** 0 = DSRATE for SDADC, for non I91200B 1 = BSRATE for BS, only for I91200B
[10:9]	**BSRATE**	**Down Sampling for BS, only I91200B** 00 = down sample 2000 for SPS384 01 = down sampel 4000(reserved) 10 = down sample 8000(reserved) 11 = down sample 16000 for SPS9.6, SPS19.2, SPS38.4 and SPS76.8
[8]	**DMICEN**	**Digital MIC Enable** 1 = turn digital MIC function input from GPIO. 0 = keep SDADC function.
[7]	**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.
[6:4]	**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** 0 = 32 bits 1 = 16 bits 2 = 8 bits 3 = 24 bits

[1:0]	**DSRATE**	**Down Sampling Ratio (for non I91200B series)** 0 = reserved 1 = down sample X 16 2 = down sample X 32 3 = down sample X 64

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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POINTER				**Reserved**	**THIF**	**EMPTY**	**FULL**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:4]	**POINTER**	**SDADC FIFO Pointer (Read Only)** The FULL bit and POINTER[3:0] indicates the field that the valid data count within the SDADC FIFO buffer. The Maximum value shown in POINTER is 8. When the using level of SDADC FIFO Buffer equal to 8, The FULL bit is set to 1. Note: ADC will not shift data into FIFO when FULL flag is set
[3]	**Reserved**	Reserved.
[2]	**THIF**	**ADC FIFO Threshold Interrupt Status (Read Only)** 1 = The valid data count within the SDADC 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.
[0]	**PDMAEN**	**Enable SDADC PDMA Receive Channel** 1 = Enable SDADC PDMA. 0 = Disable SDADC PDMA.

A/D 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**	**Reserved**

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 CMPCOND and CMPMATCNT 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]	**Reserved**	Reserved.

A/D 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**	**Reserved**

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 CMPCOND and CMPMATCNT 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]	**Reserved**	Reserved.

SD Analog Block Control Register (SDADC_SDCHOP)

Register	Offset	R/W	Description	Reset Value
SDADC_SDCHOP	SDADC_BA+0x20	R/W	Sigma Delta Analog Block Control Register	0x001c_1021

31	30	29	28	27	26	25	24
AUDIOPATHSEL		**Reserved**	**Reserved**	**Reserved**	**Reserved**	**Reserved**	**Reserved**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**PGAADCDC**		**PGAADCUP**	**PGABYPS**	**PGACLASSA**	**PGACMLCKADJ**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PGACMLCK	**PGADISCH**	**PGAGAIN**	**PGAIBLOOP**	**PGAIBCTR**			**PGAMODE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PGAMODE		**PGAMUTE**	**PGAPU**	**Reserved**	**BIAS**		**PD**

Bits	Description	
[31:30]	**AUDIOPATHSEL**	**Audio Path Selection, Connect SDADC input to** 00 = PGA (default) 01 = MICN and MICP pins (bypass PGA) 10 = Bridge Sense 11 = Reserved
[29:23]	**Reserved**	Reserved
[22:21]	**PGA_ADCDC**	**Action takes effect when PGA_DISCH=1** Bit[21]: ACDC_CTRL[0] charges INP to VREF Bit[22]: ACDC_CTRL[1] charges INN to VREF 00=Default
[20]	**PGA_HZMODE**	**Select input impedance** 0 = 12k Ohm input impedance 1 = 500k Ohm input impedance (default)
[19]	**PGA_TRIMOBC**	**Trim current in output driver** 0=disable 1=enable (default)
[18]	**PGA_CLASSA**	**Enable Class A mode of operation** 0 = Class AB 1 = Class A (default)

<table>
<tr><td>[17:16]</td><td><b>PGA_CMLCKADJ</b></td><td><b>Common mode Threshold lock adjust. Action takes effect when PGA_CMLCK=0</b>
00---0.98 (default)
01---0.96
10---1.01
11---1.04
default=00</td></tr>
<tr><td>[15]</td><td><b>PGA_CMLCK</b></td><td><b>Common mode Threshold lock adjust enable</b>
0 = Enable
1 = Disable</td></tr>
<tr><td>[14]</td><td><b>PGA_DISCH</b></td><td><b>Charge inputs selected by PGA_ACDC[1:0] to VREF</b>
1 = Enable
0 = Disable</td></tr>
<tr><td>[13]</td><td><b>PGA_GAIN</b></td><td><b>PGA Gain (default=0)</b>
<table>
<tr><td>PGA_GAIN</td><td>PGA_HZMODE=0</td><td>PGA_HZMODE=1</td></tr>
<tr><td>0</td><td>0dB</td><td>6dB</td></tr>
<tr><td>1</td><td>6dB</td><td>12dB</td></tr>
</table></td></tr>
<tr><td>[12]</td><td><b>PGA_IBLOOP</b></td><td><b>Trim PGA current</b>
1=default</td></tr>
<tr><td>[11:9]</td><td><b>PGA_IBCTR</b></td><td><b>Trim PGA Current</b>
0=default</td></tr>
<tr><td>[8:6]</td><td><b>PGA_MODE</b></td><td><b>Each bit has respective function as following.</b>
PGA_MODE[0] = Disable anti-aliasing filter adjust
PGA_MODE [1] = Short the inputs (for calibrarion)
PGA_MODE[2] = Noise reduction enable.
1 = Enable
0 = Disable (default)</td></tr>
<tr><td>[5]</td><td><b>PGA_MUTE</b></td><td><b>Mute control signal</b>
0—disable
1—enable</td></tr>
<tr><td>[4]</td><td><b>PGA_PU</b></td><td><b>Power up PGA</b>
0—disable
1—enable</td></tr>
<tr><td>[3]</td><td><b>Reserved</b></td><td>Reserved</td></tr>
<tr><td>[2:1]</td><td><b>BIAS</b></td><td><b>SDADC Bias Current Selection</b>
00 = 1
01 = 0.75
10 = 0.5
11 = 1.25</td></tr>
</table>

[0]	**PD**	**SDADC Power Down** 0 = SDADC power on 1 = SDADC power off

5.19 Bridge Sense

5.19.1 Functional Description

The Bridge Sense block consists of three parts, a low noise bandgap, an LDO and an Instrumentation Amplifier (IA). See

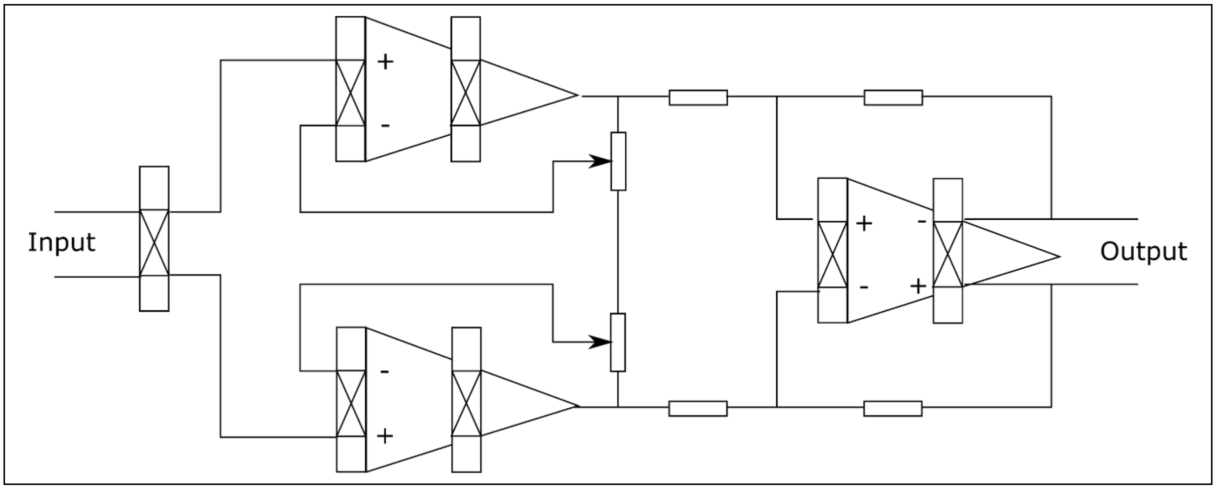


Figure 5-91 for the block diagram.

The bandgap circuit generates a reference voltage, VBG. The LDO uses VBG to make a stable voltage that can be used to bias a Wheatstone bridge. The maximum output current of the LDO is 10mA. The IA can sense and amplify the output signal of the Wheatstone bridge and buffer it before it goes to a high resolution Sigma Delta ADC (SDADC).

Note: Bridge Sense function only supports in ISD91200B series.

5.19.2 Features

Bandgap

- Low noise
- Trimmable

LDO

- Output voltage from 1.5 to 4.5V with 0.3V steps.
- Current limit

Instrumentation Amplifier

- Gain adjustable from 1x to 128x
- System chopper at the input

5.19.3 Block Diagram

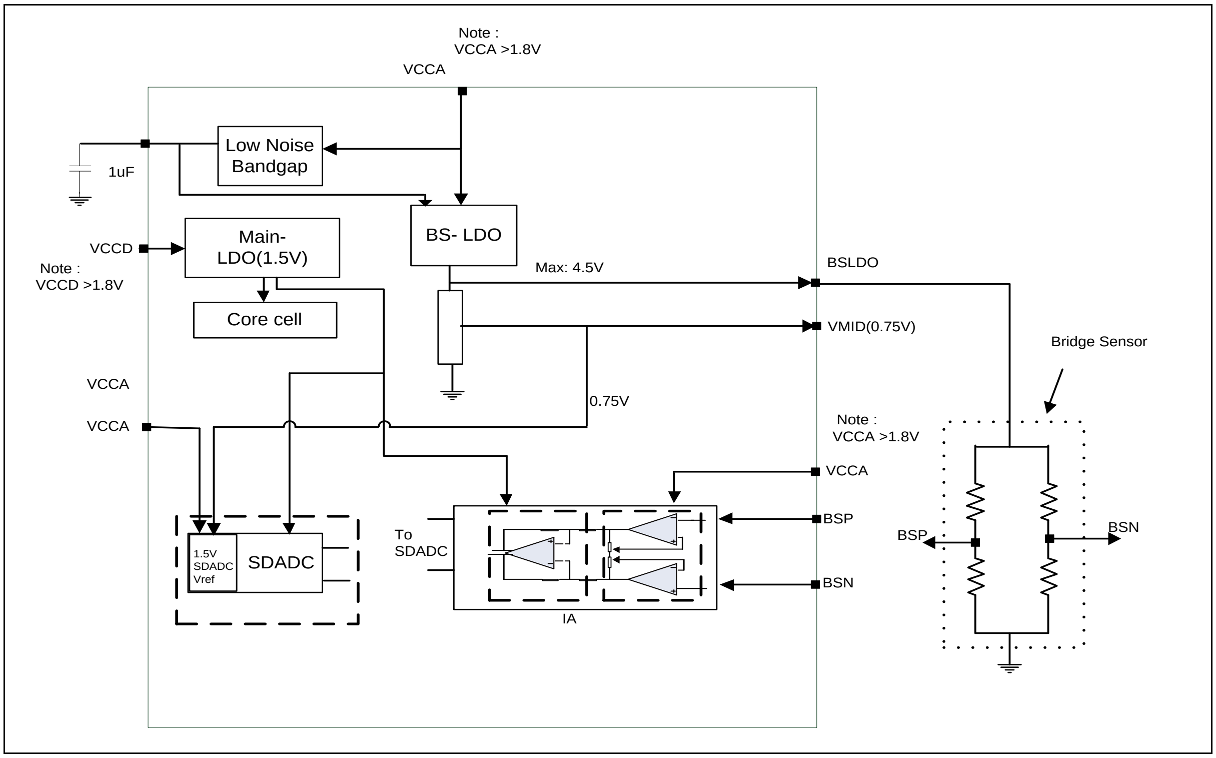


Figure 5-90 Block diagram of Bridge Sense.

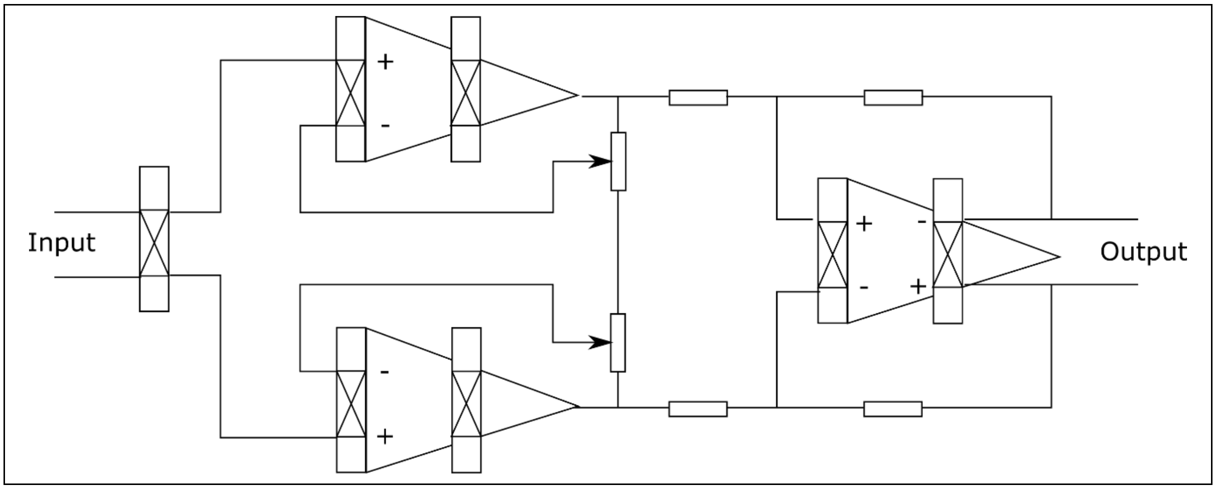


Figure 5-91 Block diagram of instrumentation amplifier (IA).

5.19.4 Operation

5.19.4.1 Startup

To power on the bridge sense part of the I92160B the bridge sense bandgap and the LDO have to be turned on first.

To charge the external filter capacitor connected to pins VBG and VSSA faster bit BFLTCHRG of

BS_BANDGP_LDO has to be set high. This bit can be set low 2ms after starting the bandgap.

The output voltage of the LDO can be set before or after the LDO is turned on.

To ensure that the noise of the LDO is cancelled out, the VMID Reference Generator has to be turned off and bit DIVEN in register BS_BANDGAP_LDO has to be enabled.

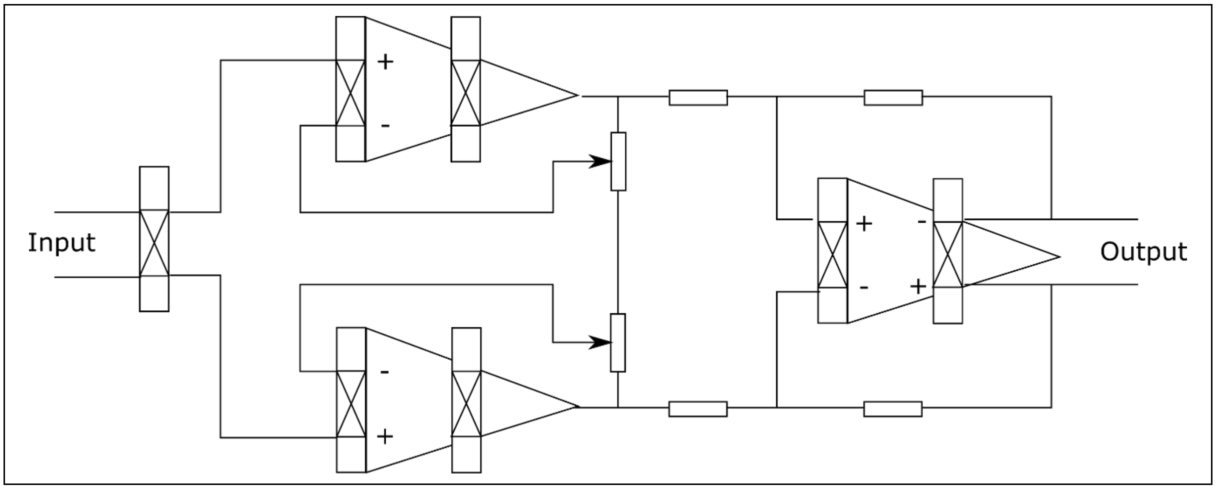
After that the instrumentation amplifier (IA) and the SDADC can be enabled.

See section 5.18.4.2 on how to set the output sample rate of the SDADC.

5.19.4.2 Choppers

The bandgap has nested choppers around the opamp to reduce the effect of its offset and 1/f noise on the bandgap output. The chopping frequencies can be adjusted and the choppers can also be turned off. The low frequency chopper enable bit (BCHOPLFEN) in register BS_BANDGPLDO can be set high while BCHOPEN is low, but the low frequency chopper will only function when BCHOPEN is set high.

The instrumentation amplifier also has two sets of choppers, one set running at a high frequency and one running at a low frequency. For the low frequency chopper the demodulation has to be done by software. See below figure.



As with the bandgap the chopper frequencies can be adjusted.

5.19.4.3 Ripple Trim Instrumentation Amplifier

The purpose of the choppers in the instrumentation amplifier (IA) is to prevent that the offset and the 1/f noise of the opamps are added to the signal. This is done by moving the offset and 1/f noise to the chopper frequency. If the gain of the IA is set high the resulting ripple will reduce the input voltage window of the IA. To reduce this impact, the offset of the input opamps in the IA can be trimmed. To do this, the IA can be set in the offset trimmode, by setting bit OFFSETTRIM in register BS_INSTRAMP high.

In the offset trim mode the input of the IA is shorted and connected to the 1.5V supply. The choppers of the two input opamps can be turned off and on independently and when a chopper is off its phase can be changed as well. Changing the chopper phase of one of the input opamps should change the output value of the sigma delta converter. This change in output value is a measure of the offset of that opamp and should be minimized by adjusting the appropriate OFFSET bits in the register BS_INSTRAMP.

Following are compensation examples for Offset Minimum Calibration.

Step1: Set bit INSTRAMP[10] =1 (BS Instrumentation Amplifier offset negative

input opamp enable), (note INSTRAMP[11] =0)

Step2: Set bit INSTRAMP[9:4] = 0x3f ,code 0x20 doesn't exist

Step3: Ignore the next 64 output cycles, for signal stable

Step4: Capture 64 cycles of output data and average store them in an array ;

Step5: decrease INSTRAMP[9:4] ,

if INSTRAMP[9:4]>= 0 , Go back Step3 to get next average cycle

Step6: Get BSN Minimum offset value in array.

Step7: Set bit INSTRAMP[11] =1 (BS Instrumentation Amplifier offset positive input opamp enable), (note INSTRAMP[10] =0)

Step8: Set bit INSTRAMP[17:12] = 0x3f code 0x20 doesn't exist

Step9: Ignore the next 64 output cycles, for signal stable

Step10: Capture 64 cycles of output data and average store them in an array ;

Step11: decrease INSTRAMP[9:4] ,

if INSTRAMP[9:4]>= 0 , Go back Step9 to get next average circle

Step12: Get BSP Minimum offset value in array.

Following are compensation examples for one sample measure.

Step1: Set bit INSTRAMP[30] =0 (BS Instrumentation Amplifier System chopper phase);

Step2: Ignore the next N2 output cycles;

Step3: Capture N1 cycles of output data and store them in an array DOUTP[1:N1];

Step4: Invert the input by setting bit INSTRAMP[30] =1;

Step5: Ignore the next N2 output cycles;

Step6: Capture N1 cycles of output data and store them in an array DOUTN[1:N1];

Step7: Calculate the chopped output:

$$DOUTchp = \frac{1}{2 \cdot N1} \cdot \left(\left(\sum_{n=1}^{N1} Doutp(n) \right) - \left(\sum_{n=1}^{N1} Doutn(n) \right) \right)$$

Step8: Go back Step1 to get next sample .

N1=2, N2=3

5.19.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
BS_BANDGPLDO	SDADC_BA+0x24	R/W	Bridge Sense Bandgap and LDO Control Register	0x0000_400e
BS_INSTRAMP	SDADC_BA+0x28	R/W	Bridge Sense Instrumentation Amplifier Control Register	0x000c_0c00

5.19.6 Register Description

Bridge Sense Bandgap and LDO Control Register (BS_BANDGPLDO)

Register	Offset	R/W	Description	Reset Value
BS_BANDGPLDO	BS_BA+0x0	R/W	Bridge Sense Bandgap and LDO Control Register	0x0000_400e

31	30	29	28	27	26	25	24
CLKEN	CLKDIV						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				LDOSET			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	DIVEN	DISCHRG	LDOEN	Reserved	BGCCDIVF		BCHOPPH
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BTRIM				BCHOPLFEN	BCHOPEN	BFLTCHRG	BGEN

Bits	Description	
[31]	**CLKEN**	**Bridge Sense clock enable for 400KHz, 200KHz and 160Hz** 0 = disable 1 = enable
[30:24]	**CLKDIV**	**Bridge Sense clock divider for 400KHz** ADC_CLK /(2*400K)
[23:20]	**Reserved**	**Reserved**
[19:16]	**LDOSET**	**BS set LDO output voltage** 0000 = 1.5 0001 = 1.8 0010 = 2.1 0011 = 2.4 0100 = 2.7 0101 = 3.0 0110 = 3.3 0111 = 3.6 1000 = 3.9 1001 = 4.2 1010 = 4.5
[15]	**Reserved**	Reserved

[14]	**DIVEN**	**BS LDO Voltage Divider Enable** 1 = enable 0 = disable
[13]	**DISCHRG**	**BS LDO Discharge** 1 = enable 0 = disable
[12]	**LDOEN**	**BS LDO(Bridge Bias) Enable** 1 = enable 0 = disable
[11]	**Reserved**	**Reserved**
[10:9]	**BGCCDIVF**	**BS Bandgap Lf chopper frequency selection** 00 = 1024 01 = 512 10 = 256 11 = 128
[8]	**BCHOPPH**	**BS Bandgap chopper phase** When chopper is off 0 = chopper switches in default state 1 = invert chopper switches
[7:4]	**BTRIM**	**BS Bandgap TRIM** 0000 = bandgap +0 0001 = bandgap+(1*12)mV 0010 = bandgap+(2*12)mV 0011 = bandgap+(3*12)mV 0100 = bandgap+(4*12)mV 0101 = bandgap+(5*12)mV 0110 = bandgap+(6*12)mV 0111 = bandgap+(7*12)mV 1000 = bandgap-(8*12)mV 1001 = bandgap-(7*12)mV 1010 = bandgap-(6*12)mV 1011 = bandgap-(5*12)mV 1100 = bandgap-(4*12)mV 1101 = bandgap-(3*12)mV 1110 = bandgap-(2*12)mV 1111 = bandgap-(1*12)mV
[3]	**BCHOPLFEN**	**BS Bandgap low frequency chopper enable** 1 = enable 0 = disable
[2]	**BCHOPEN**	**BS Bandgap chopper enable** 1 = enable 0 = disable

[1]	**BFLTCHRG**	**BS Bandgap filter charge** 1 = quick charge external capacitor 0 = high impedance mode (low noise)
[0]	**BGEN**	**BS Bandgap enable** 1 = enable 0 = disable

Bridge Sense Instrumentation Amplifier Control Register (BS_INSTRAMP)

Register	Offset	R/W	Description	Reset Value
BS_INSTRAMP	BS_BA+0x04	R/W	Bridge Sense Instrumentation Amplifier Control Register	0x000c_0c00

31	30	29	28	27	26	25	24
Reserved	SCHOPPH	SCHOPF		CHOPF		CHOPOPH	CHOPNPH
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CHOPPPH	CHOPPDIS	CHOPNDIS	OFFSETTRIM	SYSCHOPEN	CHOPEN	OFFSETP	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
OFFSETP				OFFSETPEN	OFFSETNEN	OFFSETN	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OFFSETN				GAIN			EN

Bits	Description	
[31]	**Reserved**	**Reserved**
[30]	**SCHOPPH**	**BS Instrumentation Amplifer System chopper phase** When system chopper is off 0 = system chopper switches in default state 1 = invert system chopper switches
[29:28]	**SCHOPF**	**BS Instrumentation Amplifer System chopper frequency** 00 = 40Hz 01 = 80Hz 10 = 20Hz 11 = 10Hz
[27:26]	**CHOPF**	**BS Instrumentation Amplifer chopper frequency selection** 00 = 100kHz 01 = 200kHz 10 = 50kHz 11 = 25kHz
[25]	**CHOPOPH**	**BS Instrumentation Amplifer chopper phase output opamp** When chopper is off 0 = chopper switches in output opamp in default state 1 = invert chopper switches in output opamp
[24]	**CHOPPPH**	**BS Instrumentation Amplifer chopper phase positive input opamp** When chopper is off or when OFFSETTRIM bit is high 0 = chopper switches in positive input opamp in default state 1 = invert chopper switches in positive input opamp

[23]	**CHOPNPH**	**BS Instrumentation Amplifer chopper phase negative input opamp** When chopper is off or when OFFSETTRIM bit is high 0 = chopper switches in negative input opamp in default state 1 = invert chopper switches in negative input opamp
[22]	**CHOPPDIS**	**BS Instrumentation Amplifer chopper positive disable** When OFFSETTRIM bit is high 1 = disable chopper in positive input opamp 0 = Enable chopper in positive input opamp
[21]	**CHOPNDIS**	**BS Instrumentation Amplifer chopper negative disable** When OFFSETTRIM bit is high 1 = disable chopper in negative input opamp 0 = Enable chopper in negative input opamp
[20]	**OFFSETTRIM**	**BS Instrumentation Amplifer offset trim mode** 1= enable offset trim mode 0= disable offset trim mode
[19]	**SYSCHOPEN**	**BS Instrumentation Amplifer system chopper enable** 1= enable system chopper 0= disable system chopper
[18]	**CHOPEN**	**BS Instrumentation Amplifer chopper enable** 1= enable choppers in the three opamps 0= disable choppers in the three opamps
[17:12]	**OFFSETP**	**BS Instrumentation Amplifer offset positive input opamp** bit[17]=sign bit[16:12]=magnitude
[11]	**OFFSETPEN**	**BS Instrumentation Amplifer offset positive input opamp enable** 1= use value of bits [17:12] to change offset of positive input opamp 0= don’t use bits [17:12]
[10]	**OFFSETNEN**	**BS Instrumentation Amplifer offset negetvie input opamp enable** 1= use value of bits [9:4] to change offset of positive input opamp 0= don’t use bits [9:4]
[9:4]	**OFFSETN**	**BS Instrumentation Amplifer offset negetive input opamp** bit[9]=sign bit[8:4]=magnitude
[3:1]	**GAIN**	**BS Instrumentation Amplifer GAIN** 000 = 1 001 = 2 010 = 4 011 = 8 100 = 16 101 = 32 110 = 64 111 = 128

[0]	**EN**	**BS Instrumentation Amplifer enable** 1 = Enable 0 = Disable

5.20 Audio Class D Speaker Driver (DPWM)

5.20.1 Functional Description

The ISD91200 includes a differential Class D (PWM) speaker driver capable of delivering 0.5W into an 8Ω load at 5V supply voltage. The driver works by up-sampling and modulating a PCM input to differentially drive the SPK+ and SPK- pins. The speaker driver operates from its own independent supply VCCSPK and VSSSPK. This supply should be well decoupled as peak currents from speaker driver are large.

5.20.2 Features

- Differential Audio PWM Output (DPWM).
- Direct connection of speaker
- 0.5W drive capability into 8Ω load at 5.5V.
- Configurable up-sampling to support sample rates from 8~48kHz.
- Programmable volume control from -108dB to +36dB in 0.5 dB step.
- Programmable biquad filter to support multiple sample rates from 8~48kHz.
- PDMA data channel for streaming of PCM audio data.

5.20.3 Block Diagram

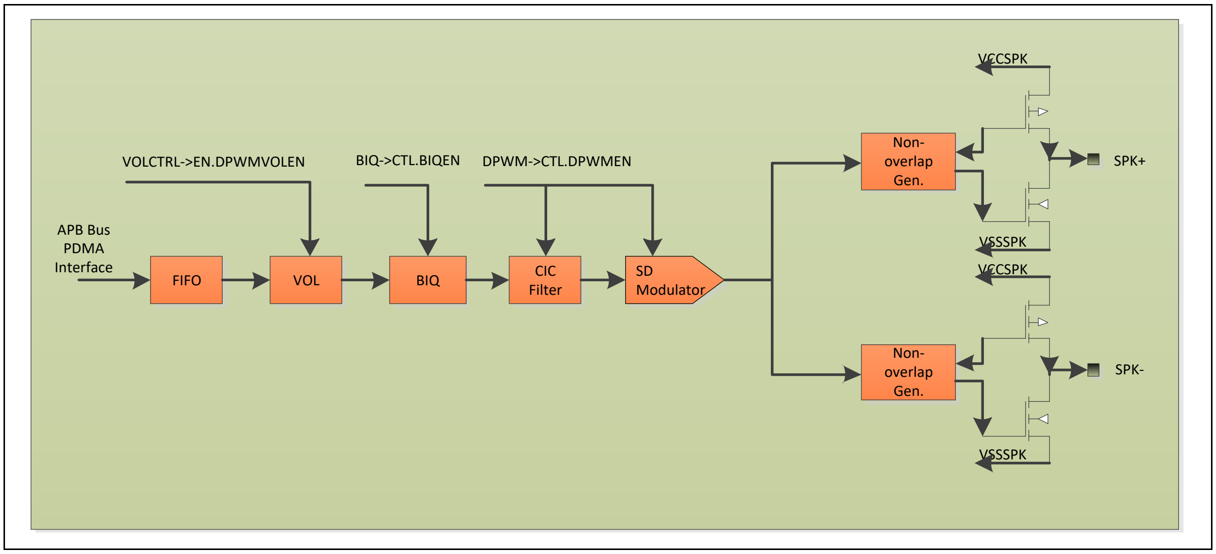


Figure 5-92 DPWM Block Diagram

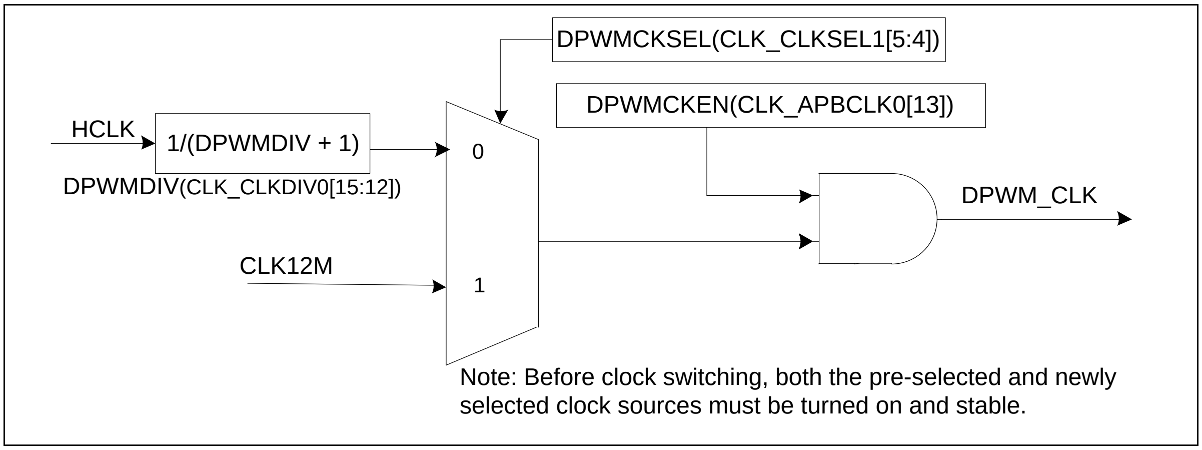
5.20.4 Operation

The DPWM 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 up-sampling Cascaded Integrator Comb (CIC) filter with an up-sampling ratio of 64. The signal is then modulated and sent to the driver stage through a non-overlap circuit. Master clock rate of the Delta-Sigma modulator is controlled by DPWM_CLK. This clock can be HCLK/(DPWMDIV+1) or HXT (refer to CLK_CLKSEL1). Ultimate SNR (Signal-to-Noise Ratio) is determined by the time resolution of the master clock.

5.20.4.1 DPWM switch frequecy

HCLK	DPWM_CLK= HCLK/DPWMDIV	Switch frequency
49.152MHz	24.576MHz	614.4KHz
	12.288MHz	307.2KHz
	8.192MHz	204.8KHz
	6.144MHz	153.6KHz

5.20.4.2 DPWM Clock Generator


DPWMCKSEL(CLK_CLKSEL1[5:4])
DPWMCKEN(CLK_APBCLK0[13])
HCLK
1/(DPWMDIV + 1)
DPWMDIV(CLK_CLKDIV0[15:12])
0
1
CLK12M
DPWM_CLK

Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

5.20.4.3 Determining Sample Rate

The sample rate at which the DPWM block consumes audio data is given by:

$$Fs = DWPM\_CLK \div ZOHDIV \div 64 \div BIQ\_CTL.DPWMPUSR$$

Where HCLK is the master CPU clock rate and DPWM_ZOHDIV is the divider control register. A table of common audio sample rates is provided below.

Table 5-22 DPWM Sample Rates for Various HCLK

	DPWM_CLK = 24.576MHz			
Fs (sample rate)	ZOHDIV (clock divider) 0x40070010	BIQ Enable	BIQ_CTL.DPWMPUSR	DPWM OSR
48KHz	8	off	1x	64
	4	on	2x	128
	2	on	4x	256
32KHz	12	off	1x	64
	6	on	2x	128
	4	on	3x	192

	3	on	4x	256
16KHz	24	off	1x	64
	12	on	2x	128
	6	on	4x	256
	4	on	6x	384
8KHz	48	off	1x	64
	24	on	2x	128
	12	on	4x	256
	8	on	6x	384

5.20.4.4 Configuring Speaker Driver

To operate the speaker driver the following configuration is recommended:

- Enable DPWM clock source (CLK_APBCLK0.DPWMCKEN, CLK_CLKSEL1.DPWMCKSEL).
- Reset DPWM IP block. (SYS_IPRST1.DPWMRST)
- Select sample rate based on current DPWM_CLK frequency.
- Setup PDMA channel to provide data to DPWM.
- Enable PDMA Request.
- Enable Driver.

To operate the speaker driver the following configuration is recommended with BIQ and Volume control.

- Enable DPWM clock source (APBCLK0.DPWMCKEN, CLKSEL1.DPWMSEL).
- Reset DPWM IP block. (IPRST1.DPWMRST,IPRST1.BIQ_RST).
- Enable BIQ clock source (APBCLK0.BIQALCEN).
- Enable BIQ on DPWM path (BIQ_CTL.PATHSEL, BIQ_CTL.BIQEN, BIQ_CTL.DPWMPUSR, BIQ_CTL.STAGE, BIQ_CTL.HPFON).
- Set BIQ coefficient(BIQ_COEFF)
- Setup DPWM volume control (VOLCTRL_EN.DPWMVOLEN,VOLCTRL_DPWMVAL).
  (note: setup BIQ_CTL.DLCOEFF =1 and BIQ enable BIQ_CTL.BIQEN =1 for BIQ operation).
- Select sample rate based on current DPWM_CLK frequency and set ZOHDIV
- Setup PDMA channel to provide data to DPWM.
- Enable PDMA Request.
- Enable DPWM IP (DPWM_CTL.DPWMEN).

5.20.4.5 Peripheral DMA Request

Normal use of the DPWM is with PDMA. In this mode DPWM requests PDMA service whenever there is space in FIFO. PDMA channel will copy data from a streaming buffer to the DPWM and alert the CPU when buffer is empty. In this way an entire buffer of data can be sent to DPWM without any CPU intervention.

5.20.5 DPWM Register Map

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

Register	Offset	R/W	Description	Reset Value
DPWM Base Address: **DPWM_BA = 0x400E_0000**				
DPWM_CTL	DPWM_BA+0x00	R/W	DPWM Control Register	0x0000_0000
DPWM_STS	DPWM_BA+0x04	R	DPWM DATA FIFO Status Register	0x0000_0002
DPWM_DMACTL	DPWM_BA+0x08	R/W	DPWM PDMA Control Register	0x0000_0000
DPWM_DATA	DPWM_BA+0x0C	W	DPWM DATA FIFO Input	0x0000_0000
DPWM_ZOHDIV	DPWM_BA+0x10	R/W	DPWM Zero Order Hold Division Register	0x0000_0006

5.20.6 DPWM Register Description

DPWM Control Register (DPWM_CTL)

Register	Offset	R/W	Description	Reset Value
DPWM_CTL	DPWM_BA+0x00	R/W	DPWM 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**
RXTH				**RXTHIE**	**Reserved**	**Reserved**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DPWMDRVEN	**DPWMEN**	**Reserved**		**DEADTIME**	**Reserved**	**FIFOWIDTH**	

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:12]	**RXTH**	**DPWM FIFO Threshold** If the valid data count of the DPWM FIFO buffer is less than or equal to RXTH setting, the RXTHIF bit will set to 1, else the RXTHIF bit will be cleared to 0.
[11]	**RXTHIE**	**DPWM FIFO Threshold Interrupt** 0 = DPWM FIFO threshold interrupt Disabled 1 = DPWM FIFO threshold interrupt Enabled.
[10]	**Reserved**	**Reserved**
[9:8]	**Reserved**	**Reserved**
[7]	**DWPMDRVEN**	**DPWM Driver Enable** 0 = Disable DPWM Driver. 1 = Enable DPWM Diver.
[6]	**DPWMEN**	**DPWM Enable** 0 = Disable DPWM. 1 = Enable DPWM
[5:4]	**Reserved**	Reserved.
[3]	**DEADTIME**	**DPWM Driver DEADTIME Control.** Enabling this bit will insert an additional clock cycle deadtime into the switching of PMOS and NMOS driver transistors.
[2]	**Reserved**	Reserved.

[1:0]	**FIFOWIDTH**	**DPWM FIFO DATA WIDTH SELETION From PDMA** 00 = PDMA MSB 24bits PWDATA[31:8] 01 = PDMA 16 bits PWDATA[15:0] 10 = PDMA 8bits PWDATA[7:0] 11 = PDMA 24bits PWDATA[23:0]

DPWM FIFO Status Register (DPWM_STS)

Register	Offset	R/W	Description	Reset Value
DPWM_STS	DPWM_BA+0x04	R	DPWM DATA FIFO Status Register	0x0000_0002

31	30	29	28	27	26	25	24
Reserved	**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**
FIFOPTR				**Reserved**	**RXTHIF**	**EMPTY**	**FULL**

Bits	Description	
[31]	**Reserved**	Reserved
[30:8]	**Reserved**	Reserved.
[7:4]	**FIFOPTR**	**DPWM FIFO Pointer (Read Only)** The FULL bit and FIFOPOINTER indicates the field that the valid data count within the DPWM FIFO buffer. The Maximum value shown in FIFO_POINTER is 15. When the using level of DPWM FIFO Buffer equal to 16, The FULL bit is set to 1.
[3]	**Reserved**	Reserved.
[2]	**RXTHIF**	**DPWM FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the DPWM FIFO buffer is larger than the setting value of RXTH. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of RXTH.
[1]	**EMPTY**	**FIFO Empty** 0 = FIFO is not empty. 1 = FIFO is empty.
[0]	**FULL**	**FIFO Full** 0 = FIFO is not full. 1 = FIFO is full.

DPWM PDMA Control Register (DPWM_DMACTL)

Register	Offset	R/W	Description	Reset Value
DPWM_DMACTL	DPWM_BA+0x08	R/W	DPWM 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							**DMAEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**DMAEN**	**Enable DPWM DMA Interface** 0 = Disable PDMA. No requests will be made to PDMA controller. 1 = Enable PDMA. Block will request data from PDMA controller whenever FIFO is not empty.

DPWM FIFO Input (DPWM_DATA)

Register	Offset	R/W	Description	Reset Value
DPWM_DATA	DPWM_BA+0x0C	W	DPWM DATA FIFO Input	0x0000_0000

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

Bits	Description	
[31:0]	**INDATA**	**DPWM FIFO Audio Data Input** A write to this register pushes data onto the DPWM FIFO and increments the write pointer. This is the address that PDMA writes audio data to. Note: Each FIFO write is 32 bits effective including PDMA.

DPWM ZOH Division (DPWM_ZOHDIV)

Register	Offset	R/W	Description	Reset Value
DPWM_ZOHDIV	DPWM_BA+0x10	R/W	DPWM Zero Order Hold Division Register	0x0000_0006

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**
ZOHDIV							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**ZOHDIV**	**DPWM Zero Order Hold, Down-sampling Divisor** The input sample rate of the DPWM is set by DPWM_CLK frequency and the divisor set in this register by the following formula: Fs = DPWM_CLK /(ZOHdiv *64* BIQ_CTL.DPWMPUSR). Default is 6, which gives a sample rate of 16kHz up-sample 256 for a 24.576MHz DPWM_CLK and BIQ_CTL.DPWMPUSR is 4. ZOH_DIV must be greater than 2

5.21 Analog Functional Blocks

5.21.1 Overview

The ISD91200 contains a variety of analog functional blocks that facilitate audio processing, enable analog GPIO functions (current source, relaxation oscillator, and comparator), adjust and measure internal oscillator and provide voltage regulation. These blocks are controlled by registers in the analog block address space. This section describes these functions and registers.

5.21.2 Features

- VMID reference voltage generation.
- Current source generation for AGPIO (Analog enabled GPIO).
- LDO control for GPIOA[7:0] power domain and external device use.
- Operational Amplifier
- Comparator
- Microphone Bias generator.
- Analog Multiplexer.
- Programmable Gain Amplifier (PGA) for Audio Path
- HIRC Frequency Control.
- CapSense Relaxation Oscillator.
- Oscillator Frequency Measurement block.

5.21.3 Register Map

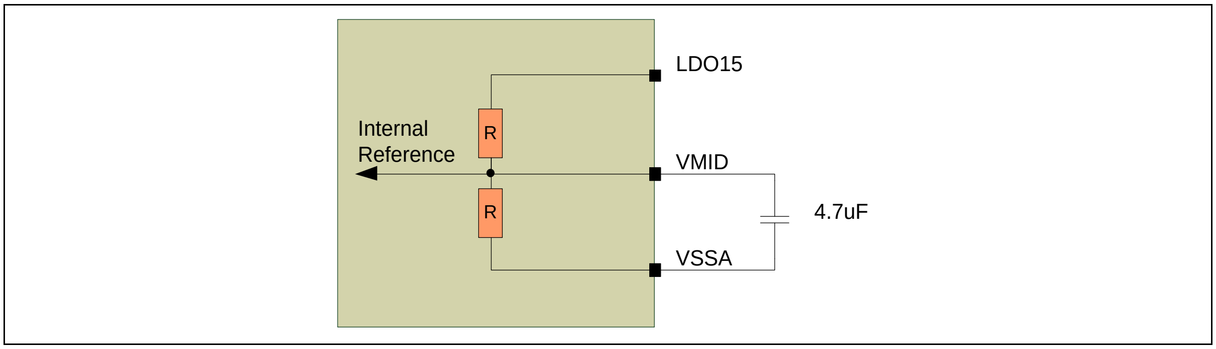
R: read only, **W**: write only, **R/W**: read/write

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_0007
ANA_LDOSEL	ANA_BA+0x20	R/W	LDO Voltage Select Register	0x0000_0000
ANA_LDOPD	ANA_BA+0x24	R/W	LDO Power Down Register	0x0000_0001
ANA_MICBSEL	ANA_BA+0x28	R/W	Microphone Bias Voltage Level Selection	0x0000_0000
ANA_MICBEN	ANA_BA+0x2C	R/W	Microphone Bias Enable	0x0000_0001
ANA_TRIM	ANA_BA+0x84	R/W	Oscillator Trim Register	0x0000_XXXX
ANA_FQMMCTL	ANA_BA+0x94	R/W	Frequency Measurement Control Register	0x0000_0001
ANA_FQMMCNT	ANA_BA+0x98	R	Frequency Measurement Count Register	0x0000_0000
ANA_FQMMCYC	ANA_BA+0x9C	R/W	Frequency Measurement Cycle Register	0x0000_0000

5.21.4 VMID Reference Voltage Generation

The analog path and blocks require a low noise, mid-rail, Voltage reference for operation, the VMID 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 VMID pin. A 4.7µF capacitor should be placed on this pin and returned to analog ground (VSSA) as shown in .

Before using the SDADC, PGA or other analog blocks, the VMID 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.



Note: LDO15 is IC internal signal

Figure 5-93 VMID Reference Generation

VMID Control Register (ANA_VMID)

Register	Offset	R/W	Description	Reset Value
ANA_VMID	ANA_BA+0x00	R/W	VMID Reference Control Register	0x0000_0007

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					**PDHIRES**	**PDLORES**	**PULLDOWN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDHIRES**	**Power Down High (360kΩ) Resistance Reference** 0= Connect the High Resistance reference to VMID. Use this setting for minimum power consumption. 1= The High Resistance reference is disconnected from VMID. Default power down and reset condition.
[1]	**PDLORES**	**Power Down Low (4.8kΩ) Resistance Reference** 0= Connect the Low Resistance reference to VMID. Use this setting for fast power up of VMID. Can be turned off after 50ms to save power. 1= The Low Resistance reference is disconnected from VMID. Default power down and reset condition.
[0]	**PULLDOWN**	**VMID Pulldown** 0= Release VMID pin for reference operation. 1= Pull VMID pin to ground. Default power down and reset condition.

5.21.5 LDO Power Domain Control

The ISD91200 provides a Low Dropout Regulator (LDO) that provides power to the I/O domain of GPIOA[7:0]. Using this regulator device can operate from a 5V supply rail and generate a 1.5-3.3V regulated supply to operate the GPIOA[7:0] domain and external loads up to 30mA. The supply pin for the LDO is the VDDBS pin which should be connected to VCCD. If the LDO is not used, both VDDBS and VD33 should be tied to VCCD. Upon POR or reset the default condition of the LDO is off, meaning supply will be high impedance. Software must configure the LDO before GPIOA[7:0] is usable (unless VD33=VCCD).

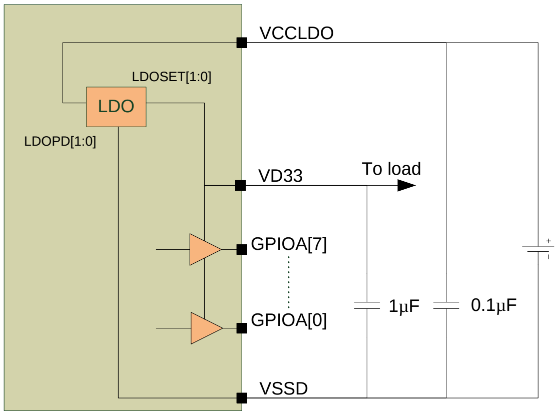


Figure 5-94 LDO Power Domain

LDO Voltage Control Register (ANA_LDOSEL)

Register	Offset	R/W	Description	Reset Value
ANA_LDOSEL	ANA_BA+0x20	R/W	LDO Voltage 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					**LDOSEL**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**LDOSEL**	**Select LDO Output Voltage** Note that maximum I/O pad operation speed only specified for voltage >2.4V. 0= 1.8V 1= 2.4V 2= 2.5V 3= 2.7V 4=3.0V 5=3.3V 6=1.5V 7=1.7V

LDO Power Down Register (ANA_LDOPD)

Register	Offset	R/W	Description	Reset Value
ANA_LDOPD	ANA_BA+0x24	R/W	LDO Power Down 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						DISCHAR	PD

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**DISCHAR**	**Discharge** 0 = Don’t discharge VD33. 1 = Switch discharge resistor to VD33.
[0]	**PD**	**Power Down LDO** When powered down no current delivered to VD33. 0= Enable LDO. 1= Power Down.

5.21.6 Microphone Bias (Replaced by Bridge Sense ADC)

Microphone Bias Enable Register (ANA_MICBEN)

Register	Offset	R/W	Description	Reset Value
ANA_MICBEN	ANA_BA+0x2C	R/W	Microphone Bias Enable	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							**PD**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PD**	**Power Down Microphone Bias** 0= Enable Microphone Bias 1= Power Down Microphone Bias **(Should be power down for I91200B series)** Note: MICBIAS output needs VMID enable together.

Microphone Bias Voltage Level Selection Register (ANA_MICBSEL)

Register	Offset	R/W	Description	Reset Value
ANA_MICBSEL	ANA_BA+0x28	R/W	Microphone Bias Voltage Level Selection	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					**LVL**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**LVL**	LVL controls the voltage output of the MICBIAS generator, voltages are encoded as following: 0 - 1.5V 1 - 1.8V 2 - 1.95V 3 - 2.1V 4 - 2.25V 5 - 2.4V 6 - 2.55V 7 - 2.7 **Note:** MICBIAS voltage should be at least 300mV lower than VCCA.

5.21.7 Oscillator Frequency Measurement and Control

The ISD91200 provides a functional unit that can be used to measure PCLK frequency given a reference frequency such as the 32.768kHz crystal or an I2S frame synchronization signal. This is simply a special purpose timer/counter as shown in Figure 5-95 Oscillator Frequency Measurement Block Diagram.

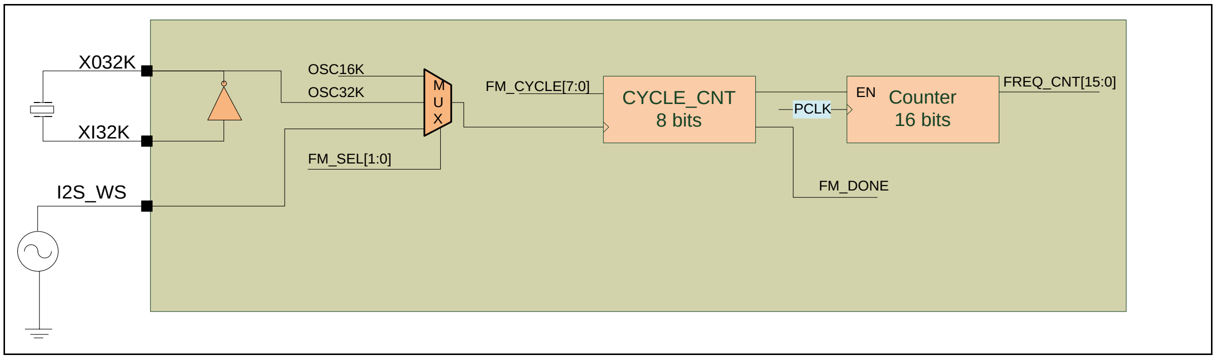


Figure 5-95 Oscillator Frequency Measurement Block Diagram

The block can be used to trim/measure the internal high frequency oscillator to the reference frequency of the 32.768kHz oscillator or an external reference frequency fed in on the I2S frame sync input. With this the internal clock can be set at arbitrary frequencies, other than those trimmed at manufacturing, or can be periodically trimmed to account for temperature variation. The block can also be used to measure the 10kHz oscillator frequency relative to the internal master oscillator.

An example of use would be to measure the internal oscillator with reference to the 32768Hz crystal. To do this:

```
CLK_APBCLK0.ANACKEN   = 1;              /* Turn on analog peripheral clock */
ANA_FQMMCTL.CLKSEL = 1;   // Select reference source as 32kHz XTAL input
ANA_FQMMCTL.CYCLESEL = DRVOSC_NUM_CYCLES-1;
ANA_FQMMCTL. FQMMEN = TRUE;
while( (ANA_FQMMCTL.MMSTS != 1) && (Timeout++ < 0x100000));
        if(        Timeout >= 0x100000)
                   return(E_DRVOSC_MEAS_TIMEOUT);
Freq = ANA_FQMMCNT;
ANA_FQMMCTL.FQMMEN = FALSE;
Freq = Freq*32768 /DRVOSC_NUM_CYCLES;
```

Oscillator Trim Register (ANA_TRIM)

Register	Offset	R/W	Description	Reset Value
ANA_TRIM	ANA_BA+0x84	R/W	Oscillator Trim Register	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**
COARSE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OSCTRIM							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**COARSE**	**COARSE** Current COARSE range setting of the oscillator. Read Only
[7:0]	**OSCTRIM**	**Oscillator Trim** Reads current oscillator trim setting. Read Only.

Frequency Measurement Control Register (ANA_FQMMCTL)

Register	Offset	R/W	Description	Reset Value
ANA_FQMMCTL	ANA_BA+0x94	R/W	Frequency Measurement Control Register	0x0000_0001

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

Bits	Description	
[31]	**FQMMEN**	**FQMMEN** 0 = Disable/Reset block. 1 = Start Frequency Measurement.
[30:24]	**Reserved**	Reserved.
[23:16]	**CYCLESEL**	**Frequency Measurement Cycles** Number of reference clock periods plus one to measure target clock (PCLK). For example if reference clock is OSC32K (T is 30.5175us), set CYCLESEL to 7, then measurement period would be 30.5175*(7+1), 244.1us.
[15:3]	**Reserved**	Reserved.
[2]	**MMSTS**	**Measurement Done** 0 = Measurement Ongoing. 1 = Measurement Complete.
[1:0]	**CLKSEL**	**Reference Clock Source** 00b: OSC10k, 01b: OSC32K (default), 1xb: I2S_WS – can be GPIOA[4,8,12] according to SYS_GPA_MFP register, configure I2S in SLAVE mode to enable.

Frequency Measurement Count (ANA_FQMMCNT)

Register	Offset	R/W	Description	Reset Value
ANA_FQMMCNT	ANA_BA+0x98	R	Frequency Measurement 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**
FQMMCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FQMMCNT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FQMMCNT**	**Frequency Measurement Count** When MMSTS = 1 and FQMMEN = 1, this is number of PCLK periods counted for frequency measurement. The frequency will be PCLK = FQMMCNT * Fref /(CYCLESEL+1) Hz. Maximum resolution of measurement is Fref /(CYCLESEL+1)*2 Hz

Frequency Measurement Cycle (ANA_FQMMCYC)

Register	Offset	R/W	Description	Reset Value
ANA_FQMMCYC	ANA_BA+0x9C	R/W	Frequency Measurement Cycle Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**FQMMCYC**	**Frequency Measurement Cycles** Number of reference clock periods plus one to measure target clock (PCLK). For example if reference clock is OSC32K (T = 30.5175µs), FQMMCYC = 7, then measurement period would be 30.5175*(7+1) = 244.1µs. This address access same register as ANA_FQMMCTL but allows access to more bits of register.

5.22 Automatic Level Control (ALC)

5.22.1 Overview and Features

The SDADC audio signal digital path is supported by digital automatic level control function. The ALC monitors the output of the SDADC biquad output when that filter is enabled in the SDADC path, or the output of the SINC filter otherwise. The SDADC output is fed into a peak detector, which updates the measured peak value whenever the absolute value of the input signal is higher than the current measured peak. The measured peak gradually decays to zero unless a new peak is detected, allowing for an accurate measurement of the signal envelope. Based on a comparison between the measured peak value and the target value, the ALC block adjusts the gain of audio signal.

Please note this gain is digital gain. Different from the trantional PGA gain which resides in analog domain and is typically adjusted by changing resistor value.

The ALC is enabled by setting ALCEN. The ALC shares a clock source with the Biquad filter so CLK_APBCLK0.BQALCKEN must be set to operate ALC. The BIQ result also multiply the ALC result, so BIQ should work together. The ALC has two functional modes, which is set by MODESEL.

- Normal mode (MODESEL = LOW)
- Peak Limiter mode (MODESEL = HIGH)

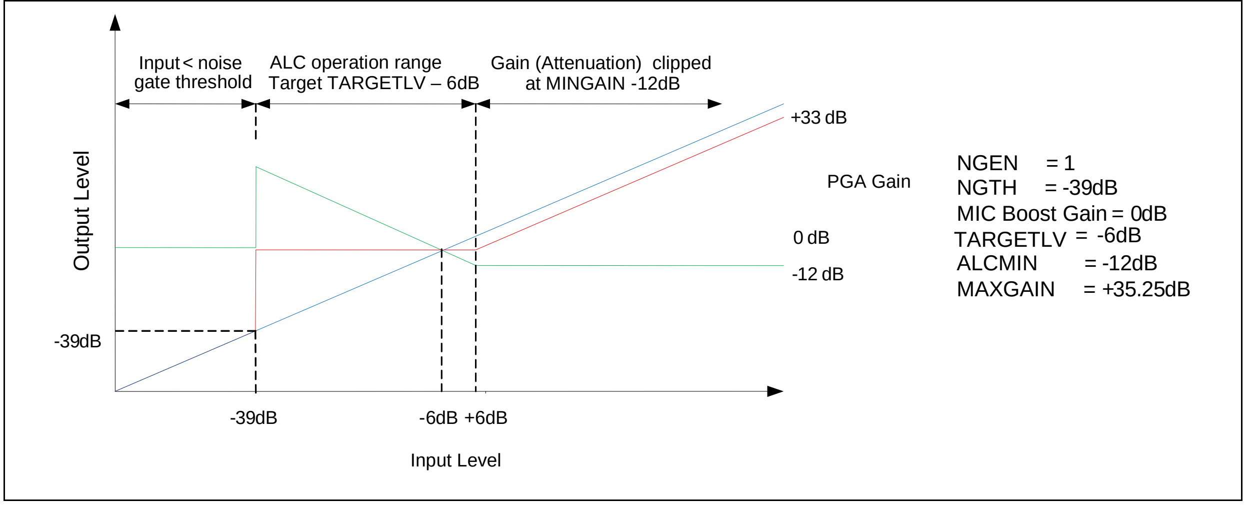


Figure 5-97 ALC Response Graph

The registers listed in the following sections allow configuration of ALC operation with respect to:

- ALC target level
- Gain increment and decrement rates
- Minimum and maximum PGA gain values for ALC operating range
- Hold time before gain increments in response to input signal
- Inhibition of gain increment during noise inputs
- Limiter mode operation

The operating range of the ALC is set by MAXGAIN and MINGAIN bits such that the PGA gain generated by the ALC is constrained to be between the programmed minimum and maximum levels. When the ALC is enabled, the PGA gain setting from PGASEL has no effect.

In Normal mode, the MAXGAIN bits set the maximum level for the PGA but in the Limiter mode MAXGAIN has no effect because the maximum level is set by the initial PGA gain setting upon enabling of the ALC.

5.22.1.1 Normal Mode

Normal mode is selected when MODESEL is set LOW and the ALC is enabled by setting ALCEN HIGH. This block adjusts the PGA gain setting up and down in response to the input level. A peak detector circuit measures the envelope of the input signal and compares it to the target level set by TARGETLV. The ALC increases the gain when the measured envelope is less than (target – 1.5dB) and decreases the gain when the measured envelope is greater than the target. The following waveform illustrates the behavior of the ALC.

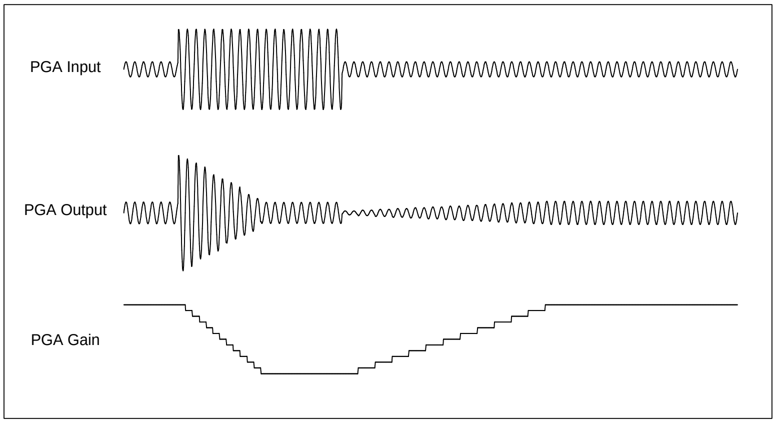


Figure 5-98 ALC Normal Mode Operation

5.22.1.2 ALC Hold Time (Normal mode Only)

The hold parameter HOLDTIME configures the time between detection of the input signal envelope being below the target range and the actual gain increase.

Input signals with different characteristics (e.g., voice vs. music) may require different settings for this parameter for optimal performance. Increasing the ALC hold time prevents the ALC from reacting too quickly to brief periods of silence such as those that may appear in music recordings; having a shorter hold time, on the other hand, may be useful in voice applications where a faster reaction time helps to adjust the volume setting for speakers with different volumes. The waveform below shows the operation of the HOLDTIME parameter.

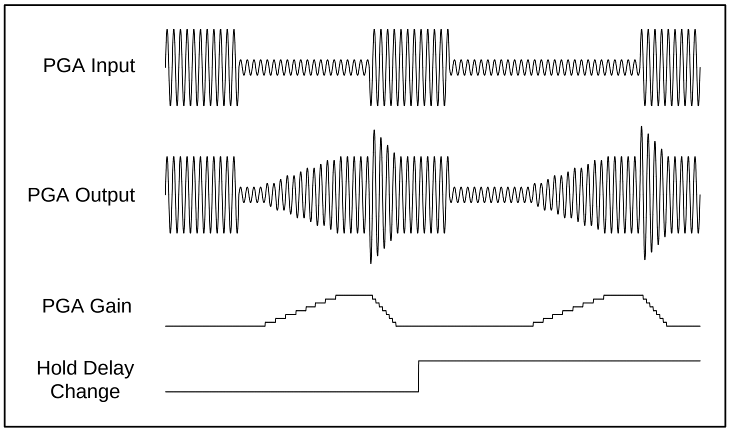


Figure 5-99 ALC Hold Time

5.22.1.3 Peak Limiter Mode

Peak Limiter mode is selected when MODESEL is set to HIGH and the ALC is enabled by setting ALCEN. In limiter mode, the PGA gain is constrained to be less than or equal to the gain setting at the time the limiter mode is enabled. In addition, attack and decay times are faster in limiter mode than in normal mode as indicated by the different lookup tables for these parameters for limiter mode. The following waveform illustrates the behavior of the ALC in Limiter mode in response to changes in various ALC parameters.

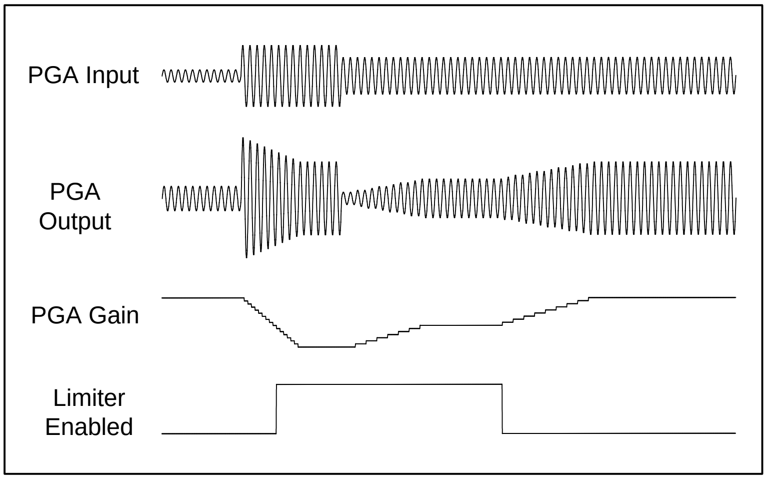


Figure 5-100 ALC Limiter Mode Operations

When the input signal exceeds 87.5% of full scale, the ALC block ramps down the PGA gain at the maximum attack rate (ATKSEL=0000) regardless of the mode and attack rate settings until the SDADC output level has been reduced below the threshold. This limits SDADC clipping if there is a sudden increase in the input signal level.

5.22.1.4 Attack Time

When the absolute value of the SDADC output exceeds the level set by the ALC threshold, TARGETLV, attack mode is initiated at a rate controlled by the attack rate register ATKSEL. The peak detector in the ALC block loads the SDADC output value when the absolute value of the SDADC output exceeds the current measured peak; otherwise, the peak decays towards zero, until a new peak has been identified. This sequence is continuously running. If the peak is ever below the target threshold, then there is no gain decrease at the next attack timer time; if it is ever above the target-1.5dB, then there is no gain increase at the next decay timer time.

5.22.1.5 Decay Times

The decay time DECAYSEL is the time constant used when the gain is increasing. In limiter mode, the time constants are faster than in ALC mode.

5.22.1.6 Noise gate (normal mode only)

A noise gate is used when there is no input signal or the noise level is below the noise gate threshold. The noise gate is enabled by setting NGEN to HIGH. It does not remove noise from the signal. The noise gate threshold NGTH is set to a desired level so when there is no signal or a very quiet signal (pause), which is composed mostly of noise, the ALC holds the gain constant instead of amplifying the signal towards the target threshold. The noise gate only operates in conjunction with the ALC (ALCEN HIGH) and ONLY in Normal mode. The noise gate flag is asserted when

(Signal at SDADC – ALC gain) < NGTH (dB)

Levels at the extremes of the range may cause inappropriate operation, so care should be taken when setting up the function.

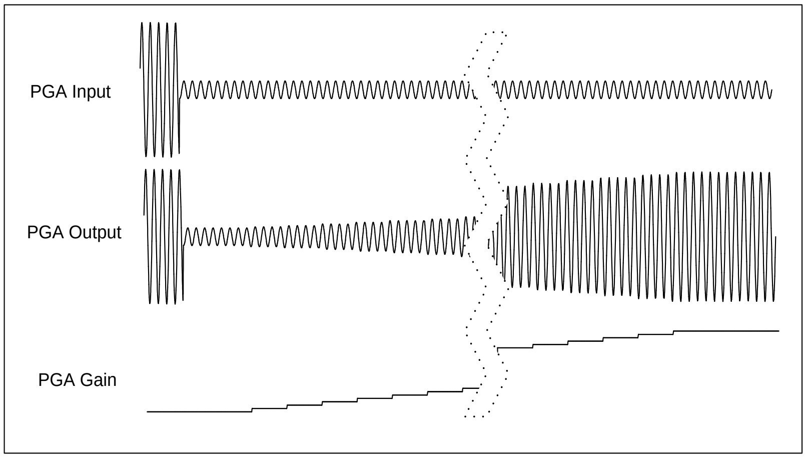


Figure 5-101 ALC Operation with Noise Gate disabled

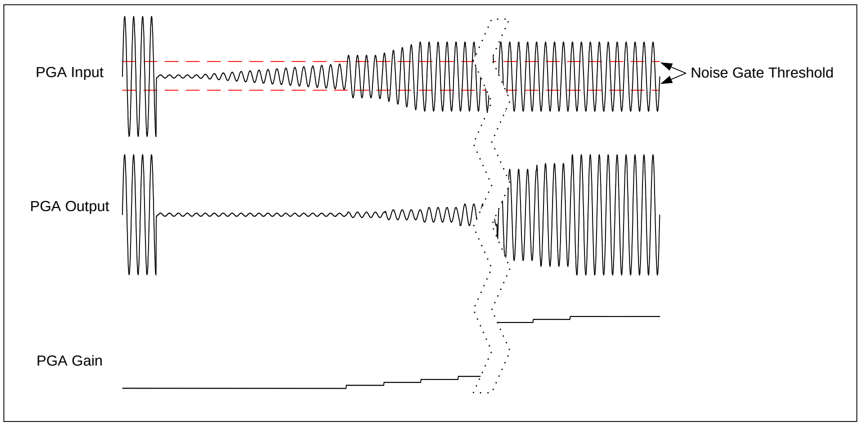


Figure 5-102 ALC Operation with Noise Gate Enabled

5.22.1.7 Zero Crossing

The PGA gain comes from either the ALC block when it is enabled or from the PGA gain register setting when the ALC is disabled. Zero crossing detection may be enabled to cause PGA gain changes to occur only at an input zero crossing. Enabling zero crossing detection limits clicks and pops that may occur if the gain changes while the input signal has a high volume.

There are two zero crossing detection enables:

- Register ZCEN – is only relevant when the ALC is enabled.

If the zero crossing function is enabled (using either register), the zero cross timeout function may take effect. If the zero crossing flag does not change polarity within 0.25 seconds of a PGA gain update (either via ALC update or PGA gain register update), then the gain will update. This backup system prevents the gain from locking up if the input signal has a small swing and a DC offset that prevents the zero crossing flag from toggling.

5.22.2 Register Map

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

Register	Offset	R/W	Description	Reset Value
ALC Base Address: **ALC_BA = 0x400B_0090**				
ALC_CTL	ALC_BA+0x00	R/W	ALC Control Register	0x0000_0000
ALC_GAIN	ALC_BA+0x04	R/W	ALC GAIN Control Register	0x6fdc_0010
ALC_STS	ALC_BA+0x08	R/W	ALC Status Register	0x0100_0000
ALC_INTCTL	ALC_BA+0x0C	R/W	ALC Interrupt Control Register	0x0000_0000

5.22.3 Register Description

ALC Control Register (ALC_CTRL)

Register	Offset	R/W	Description	Reset Value
ALC_CTL	ALC_BA+0x00	R/W	ALC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PKLIMEN	PKSEL	NGPKSEL	ALCEN	MAXGAIN			MINGAIN
23	22	21	20	19	18	17	16
MINGAIN		ALCRANGESEL	HOLDTIME				TARGETLV
15	14	13	12	11	10	9	8
TARGETLV			MODESEL	DECAYSEL			
7	6	5	4	3	2	1	0
ATKSEL				NGEN	NGTHBST		

Bits	Description	
[31]	**PKLIMEN**	**ALC Peak Limiter Enable** Default is “0”, Please set as “1”
[30]	**PKSEL**	**ALC Gain Peak Detector Select** 0 = use absolute peak value for ALC training (default). 1 = use peak-to-peak value for ALC training.
[29]	**NGPKSEL**	**ALC Noise Gate Peak Detector Select** 0 = use peak-to-peak value for noise gate threshold determination (default). 1 = use absolute peak value for noise gate threshold determination.
[28]	**ALCEN**	**ALC Select** 0 = ALC disabled (default). 1 = ALC enabled.
[27:25]	**MAXGAIN**	**ALC Maximum Gain** 0 = -6.75 dB. 1 = -0.75 dB. 2 = +5.25 dB. 3 = +11.25 dB. 4 = +17.25 dB. 5 = +23.25 dB. 6 = +29.25 dB. 7 = +35.25 dB.

[24:22]	**MINGAIN**	**ALC Minimum Gain** 0 = -12 dB. 1 = -6 dB. 2 = 0 dB. 3 = 6 dB. 4 = 12 dB. 5 = 18 dB. 6 = 24 dB. 7 = 30 dB.
[21]	**ALCRANGESEL**	**ALC Target range selection** 0 = ALC target range -28.5~ -6dB 1 = ALC target range -22.5 ~-1.5dB
[20:17]	**HOLDTIME**	**ALC Hold Time** (Value: 0~10). Hold Time = (2^HOLDTIME) ms.
[16:13]	**TARGETLV**	**ALC Target Level** 0 = -28.5 dB. 1 = -27 dB. 2 = -25.5 dB. 3 = -24 dB. 4 = -22.5 dB. 5 = -21 dB. 6 = -19.5 dB. 7 = -18 dB. 8 = -16.5 dB. 9 = -15 dB. 10 = -13.5 dB. 11 = -12 dB. 12 = -10.5 dB. 13 = -9 dB. 14 = -7.5 dB. 15 = -6 dB.
[12]	**MODESEL**	**ALC Mode** 0 = ALC normal operation mode. 1 = ALC limiter mode.
[11:8]	**DECAYSEL**	**ALC Decay Time** **(Value: 0~10)** When MODESEL = 0, Range: 500us to 512ms. When MODESEL = 1,Range: 125us to 128ms (Both ALC time doubles with every step).
[7:4]	**ATKSEL**	**ALC Attack Time** **(Value: 0~10)** When MODESEL = 0, Range: 125us to 128ms. When MODESEL = 1, Range: 31us to 32ms (time doubles with every step).

[3]	**NGEN**	**Noise Gate Enable** 0 = Noise gate disabled. 1 = Noise gate enabled.
[2:0]	**NGTHBST**	**Noise Gate Threshold** 000 --- -39dB 001--- -45dB 010---- -51dB 011 --- -57dB 100 --- -63dB 101 --- -69dB 110 --- -75dB 111 --- -81dB

ALC Gain Control Register (ALC_GAIN)

Register	Offset	R/W	Description	Reset Value
ALC_GAIN	ALC_BA+0x04	R/W	ALC GAIN Control Register	0x6fdc_0010

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

Bits	Description	
[31:16]	**PKLIMIT**	**ALC Peak Limiter Threshold** Full scale - 0x7fff Default value is 0x6fdc - 87.5% of full scale
[15:8]	**Reserved**	Reserved.
[7]	**INITGAINEN**	**ALC Update Initial Gain** 0 = ALC PGA GAIN load automatic calculating gain. 1 = ALC PGA GAIN load ALCINIT_GAIN
[6]	**ZCEN**	**ALC Zero Crossing Enable** 0 = zero crossing disabled. 1 = zero crossing enabled when update gain.
[5:0]	**INITGAIN**	**ALC Initial Gain** Set ALC initial gain. Selects the PGA gain setting from -12dB to 35.25dB in 0.75dB step size. 0x00 is lowest gain setting at -12dB and 0x3F is largest gain at 35.25dB

ALC Status Register (ALC_STATUS)

Register	Offset	R/W	Description	Reset Value
ALC_STS	ALC_BA+0x08	R/W	ALC Status Register	0x0100_0000

31	30	29	28	27	26	25	24
Reserved						**ALCGAIN**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ALCGAIN				**PEAKVAL**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PEAKVAL					**P2PVAL**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P2PVAL						**NOISEF**	**CLIPF**

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:20]	**ALCGAIN**	**ALC GAIN** Current ADC gain setting
[19:11]	**PEAKVAL**	**Peak Value** 9 MSBs of measured absolute peak value
[10:2]	**P2PVAL**	**Peak-to-peak Value** 9 MSBs of measured peak-to-peak value
[1]	**NOISEF**	**Noise Flag** Asserted when signal level is detected to be below NGTHBST
[0]	**CLIPF**	**Clipping Flag** Asserted when signal level is detected to be above 87.5% of full scale

ALC Interrupt Register (ALC_INT)

Register	Offset	R/W	Description	Reset Value
ALC_INTCTL	ALC_BA+0x0C	R/W	ALC Interrupt Control Register	0x0000_0000

31	30	29	28	27	26	25	24
ALCINT	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**GMINIF**	**GMAXIF**	**GDECIF**	**GINCIF**	**NGIF**	**PLMTIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**GMINIE**	**GMAXIE**	**GDECIE**	**GINCIE**	**NGIE**	**PLMTIE**

Bits	Description	
[31]	**ALCIF**	**ALC Interrupt flag** This interrupt flag asserts whenever the interrupt is enabled and the PGA gain is updated, either through an ALC change with the ALC enabled or through a PGA gain write with the ALC disabled. Write a 1 to this register to clear.
[30:14]	**Reserved**	Reserved.
[13]	**GMINIF**	**GAIN less than minimum GAIN interrupt flag.**
[12]	**GMAXIF**	**GAIN more than maximum GAIN interrupt flag.**
[11]	**GDECIF**	**GAIN Decrease interrupt flag**
[10]	**GINCIF**	**GAIN Increase interrupt flag**
[9]	**NGIF**	**ALC noise gating interrupt flag**
[8]	**PLMTIF**	**ALC Peak limiting Interrupt flag**
[7:6]	**Reserved**	Reserved.
[5]	**GMINIE**	**GAIN less than minimum GAIN interrupt enable control**
[4]	**GMAXIE**	**GAIN more than maximum GAIN interrupt enable control**

[3]	**GDECIE**	**GAIN Decrease interrupt enable control**
[2]	**GINCIE**	**GAIN Increase interrupt enable control**
[1]	**NGIE**	**ALC noise gating interrupt enable control**
[0]	**PLMTIE**	**ALC Peak limiting Interrupt enable control**

5.23 Capacitive Sensing Scan (CSCAN) and Operational Amplifiers

5.23.1 Overview and Features

Capacitive Sensing Scanner has the ability to set up a capacitive sensing on up to 16 GPIO pins. The block can do a single measurement or be set up to scan a defined set of GPIO before interrupting the CPU. The measurement can be done rapidly against a high precision clock (HIRC) in active mode or can be done slowly against a low frequency clock which can be done at low current in STOP or SPD power modes. The analog portion of the circuit consists of a relaxation oscillator producing a triangle wave on the parasitic capacitance attached to the GPIO.

5.23.2 Features

- Support single mode or scan mode for pin cap sensing, maximum 16 pins supported.
- Low power consumption (<10uA) touch wake up from STOP or SPD mode.
- Current source generation for AGPIO (Analog enabled GPIO)
- 16 kinds of dummy delay time (maximum 3840 clocks setting in DUR_CNT) for additional duration of periodical wakeup.

5.23.3 Operation

To operate the CSCAN with scan mode for multi pin capacitive sensing, the below steps is the sequence.

- Set CTRL.PD=0 to power up CSCAN engien
- Select the cscan timebase clock by setting SLOW_CLK
- Set AGPIO.AGPIO for expected GPIO pin to analog mode.
- Set CYCCNT.MASK for all the pins which will have capacitive sensing.
- Set CYCCNT.CYCLE_CNT for numbers of cycle to time a capacitive sensing.
- Select scan mode by setting CTRL.MODE0=1
- Set CTRL.MODE1 for interrupt with DUR_CNT delay or not. MODE1=0 for no delay, interrupt happens after finish all selected pins sensing. MODE1=1 for delay with DUR_CNT setting time.
- Set CTRL.CURRENT for controlling the bias current of relaxation comparators.
- Set CTRL.INT_EN to enable the IP interrupt
- Enable NVIC CSCAN interrupt
- Start the capacitive sensing with setting CTRL.EN

After interrupt happens, user program needs to read back the counter value of each pin capacitive sensing which sotred in SBRAM for touch algorithm judgement then trigger next loop of scan. The address is fixed for each pin, no matter pin is used for capacitive sensing or not.

SBRAM_BASE (= 0x400F0000)	**PB0**	**PB1**	**PB2**	**~**	**PB13**	**PB14**	**PB15**
Offset (unit: byte)	0x0	0x2	0x4	~	0x1A	0x1C	0x1E

5.23.4 Operational Amplifier

The ISD91200 contains two fully integrated Operational Amplifiers. These OPAs can be used for signal amplification according to specific user requirements. These blocks are controlled by registers in the analog block address space. This section describes these functions and registers

The internal Operational Amplifiers are fully under the control of internal registers, OPA0C0, OPA0C1, OPA1C0, OPA1C1 and OPA1C2. These registers control enable/disable function, input path selection and gain control.

The following diagram and table illustrate the OPA0 switch control setting and the corresponding connections.

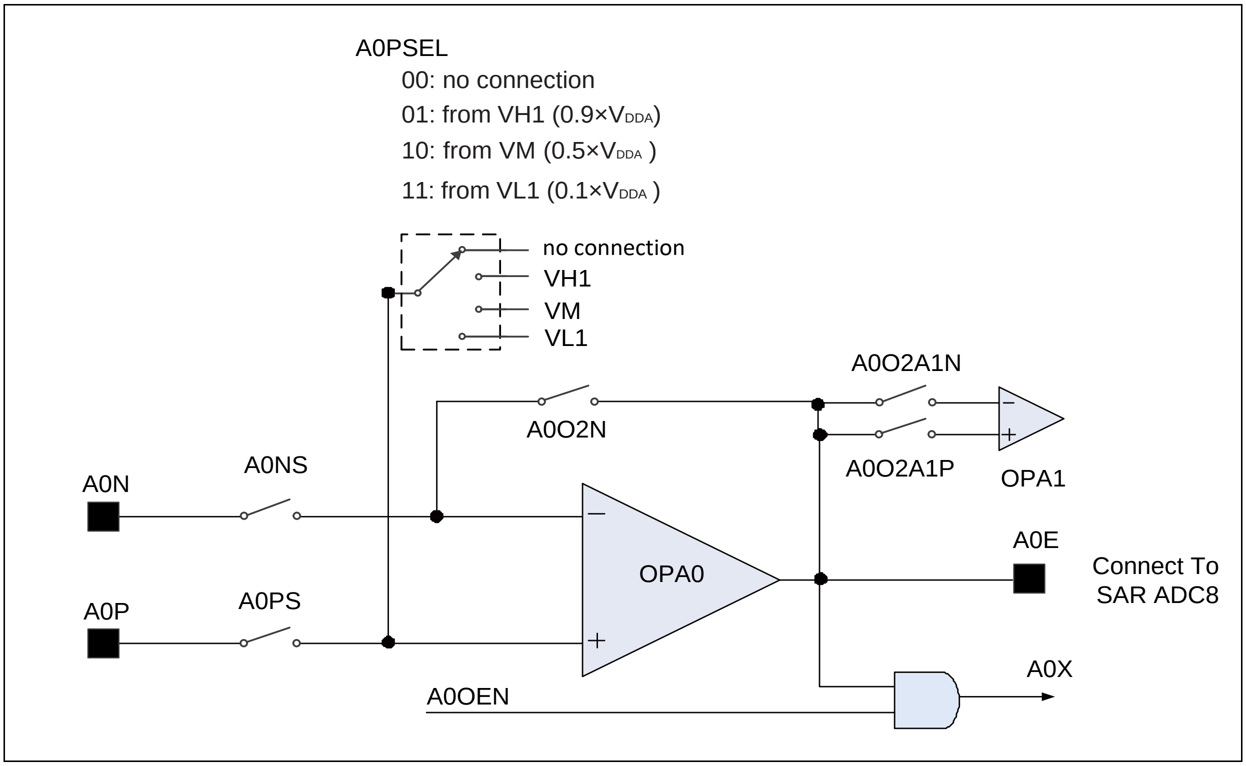


Figure 5-103 Operational Amplifier 0 Switch Control

The following diagram and table illustrate the OPA1 switch control setting and the corresponding connections. Note the PAGEN switch control selections will force some switches to be controlled by hardware automatically.

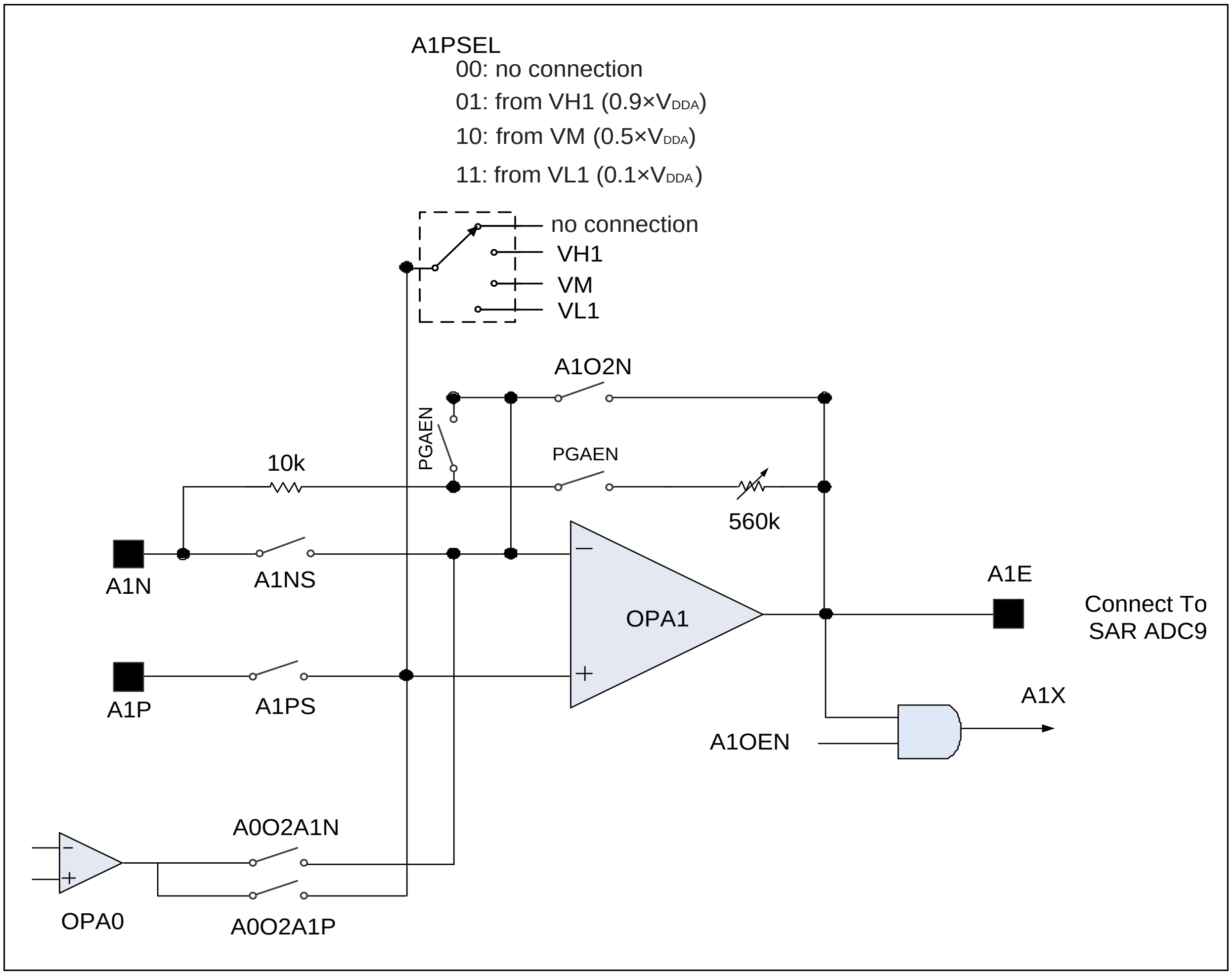


Figure 5-104 Operational Amplifier 1 Switch Control

The following diagram is OPAs bias setting, If the OPBIASEN is set to “1”, that will turn on the resistor DC path, which will generate bias voltage for OPAs

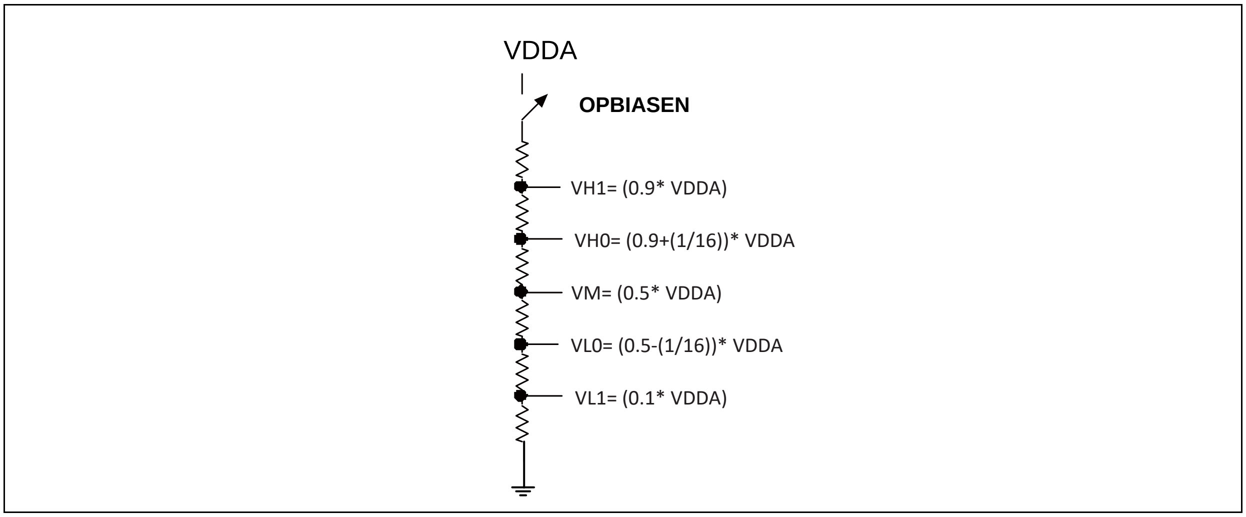


Figure 5-105 Operational Amplifier bias Switch Control

5.23.5 Comparator

The ISD91200 contains two analog comparators are contained within the devices. These functions offer flexibility via their register controlled features such as power-down, interrupt etc. Sharing their pins with normal I/O pins, the comparators do not waste precious I/O pins if there functions are otherwise unused.

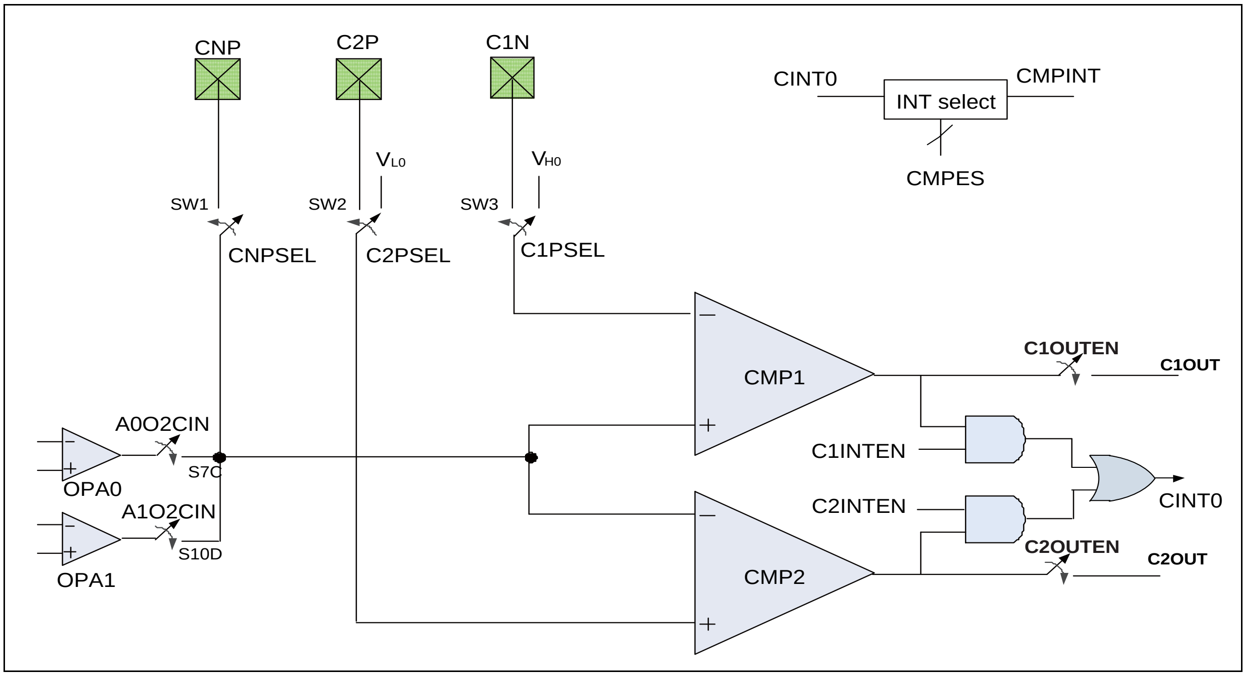


Figure 5-106 Comparator Switch Control

5.23.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
CSCAN Base Address: **CSCAN_BA = 0x400D_0000**				
CSCAN_CTRL	CSCAN_BA+0x00	R/W	CSCAN Control Register	0x8000_0000
CSCAN_CYCCNT	CSCAN_BA +0x04	R/W	CSCAN Cycle Count Control Register	0x0000_0000
CSCAN_COUNT	CSCAN_BA +0x08	R/W	CSCAN Count Status Register	0x0000_0000
CSCAN_INT	CSCAN_BA +0x0C	R/W	CSCAN Interrupt Register	0x0000_0000
CSCAN_AGPIO	CSCAN_BA+0x10	R/W	CSCAN Analog GPIO function Register	0x0000_0000
CSCAN_OPACTL	CSCAN_BA+0x14	R/W	Operational Amplifier Control Register	0x0000_0000
CSCAN_CMPCTL	CSCAN_BA+0x18	R/W	Comparator Control Register	0x0000_0000

5.23.7 Register Description

CSCAN Control Register (CSCAN_CTRL)

Register	Offset	R/W	Description	Reset Value
CSCAN_CTRL	CSCAN_BA+0x00	R/W	CSCAN Control Register	0x8000_0000

31	30	29	28	27	26	25	24
PD	EN	Reserved	Reserved	DUR_CNT			
23	22	21	20	19	18	17	16
MODE1	MODE0	SLOW_CLK	INT_EN	Reserved		CURRENT	
15	14	13	12	11	10	9	8
SEL							
7	6	5	4	3	2	1	0
SEL							

Bits	Description	
[31]	**PD**	**Power Down** 0: Enable analog circuit 1: Power down analog circuit and block.
[30]	**EN**	**CSCAN Enable** Write 1 to start. Reset by hardware when operation finished.
[27:24]	**DUR_CNT**	**CSCAN Duration Count** This counter is used to set a wakeup time after a capacitive sensing scan is complete. It is in units of low frewquency clock period (either LXT or LIRC clock) and gives delay of 160, 320, 480,640, 800, 960, 1120, 1280, 1440,1600, 1920, 2240, 2560, 2880,3200 3840 periods for settings 0,..,15.
[23]	**MODE1**	**CSCAN Mode1** 0 = Interrupt when scan finished 1 = Interrupt when DUR_CNT delay occurs.
[22]	**MODE0**	**CSCAN Mode0** 0 = Single shot Capacitive sense 1 = Scans each channel set in SCAN_MASK and stores in RAM.
[21]	**SLOW_CLK**	**CSCAN Slow Clock** 0 = Timebase clock is HIRC. 1 = Timebase clock is LIRC (XTAL32K_EN = 0) or XTAL (XTAL32K_EN = 1) **Notes:** In low speed mode, for CYCLE_CNT <5, the minimum frequency of oscillation of a CAPSENSE GPIO must be > Fclk/2. Where Fclk is the frequency of LXT or LIRC depending which is selected as reference.

[20]	**INT_EN**	**CSCAN Enable Interrupt** 0 = Interrupt disabled. 1 = Interrupt enabled.
[19:18]	**Reserved**	Keep with 0
[17:16]	**CURRENT**	**CSCAN Oscillator current** Controls the analog bais current of the capacitive relaxation oscillator. 0:300nA 1:450nA 2:600nA 3:1200nA
[15:0]	**SEL**	**CSCAN Select** In single mode selects the channel (GPIOB[15:0]) to perform measurement on.

CSCAN Cycle Count Control Register (CSCAN_CYCCNT)

Register	Offset	R/W	Description	Reset Value
CSCAN_CYCCNT	CSCAN_BA +0x04	R/W	CSCAN Cycle Count Control Register	0x0000_0000

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

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:16]</td><td><b>MASK</b></td><td><b>Scan Mask Register</b>
If MASK[n] is set then GPIOB[n] is included in scan of capacitive sensing.</td></tr>
<tr><td>[15:4]</td><td><b>Reserved</b></td><td>Reserved</td></tr>
<tr><td>[3:0]</td><td><b>CYCLE_CNT</b></td><td><b>CSCAN Cycle Count</b>
Number of cycles to time a CapSense even over 4 bit value decoded to 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 768, 1024, 1536, 2048, 2560, 3072
<table>
<tr><td><b>CYCLE_CNT</b></td><td>0x0</td><td>0x1</td><td>0x2</td><td>0x3</td><td>0x4</td><td>0x5</td><td>0x6</td><td>0x7</td></tr>
<tr><td><b>Cycles</b></td><td>1</td><td>2</td><td>4</td><td>8</td><td>16</td><td>32</td><td>64</td><td>128</td></tr>
<tr><td><b>CYCLE_CNT</b></td><td>0x8</td><td>0x9</td><td>0xA</td><td>0xB</td><td>0xC</td><td>0xD</td><td>0xE</td><td>0xF</td></tr>
<tr><td><b>Cycles</b></td><td>256</td><td>512</td><td>768</td><td>1024</td><td>1536</td><td>2048</td><td>2560</td><td>3072</td></tr>
</table>
<b>Notes:</b>
<ul>
<li>It is recommended to use CYCLE_CNT >= 5 (32 cycles or more) when SLOW_CLK = 1. Inappropriate CSCAN_CTRL.CURRENT settings with short scan cyle (CYCLE_CNT <= 4) might disrupt Capsense Interrupt and power mode wake up.</li>
<li>It is recommended to check result of CYCLE_CNT > 2 (4 cycles) when SLOW_CLK = 0 to avoid overflow the scan counter.</li>
</ul></td></tr>
</table>

CSCAN Count Register (CSCAN_COUNT)

Register	Offset	R/W	Description	Reset Value
CSCAN_COUNT	CSCAN_BA +0x08	R/W	CSCAN Count 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**
COUNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COUNT							

Bits	Description	
[31:16]	**Reserved**	**Reserved**
[15:0]	**COUNT**	**CSCAN Count** Count result of single scan.

CSCAN Interrupt Register (CSCAN_INT)

Register	Offset	R/W	Description	Reset Value
CSCAN_INT	CSCAN_BA +0x0C	R/W	CSCAN Interrupt 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							**INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**INT**	**CSCAN Interrupt active** Write ‘1’ to clear.

CSCAN Analog GPIO Register (CSCAN_AGPIO)

Register	Offset	R/W	Description	Reset Value
CSCAN_AGPIO	CSCAN_BA+0x10	R/W	CSCAN Analog GPIO function 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**
AGPIO							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AGPIO							

Bits	Description	
[31:16]	**Reserved**	**Scan Mask Register** If MASK[n] is set then GPIOB[n] is included in scan of capacitive sensing.
[15:0]	**AGPIO**	**CSCAN AGPIO** If bit set to 1 then corresponding GPIOB[n] is forced to an analog mode where digital input, output and pullup is disabled. Can be used to set pad into analog mode for CapSensing, SAR ADC and OPAMP functions.

Operational Amplifier Control Register (CSCAN_OPACTRL)

Register	Offset	R/W	Description	Reset Value
CSCAN_OPACTL	CSCAN_BA+0x14	R/W	Operational Amplifier Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					A1O2CIN	A0O2CIN	LPWREN
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
A0O2A1N	A0O2A1P	Reserved	VREFEN	PGAEN	PGA_GAIN		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
A1X	A1O2N	A1PSEL		A1PS	A1NS	A1OEN	A1EN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
A0X	A0O2N	A0PSEL		A0PS	A0NS	A0OEN	A0EN

Bits	Description	
[31:25]	**Reserved**	Reserved.
[26]	**A1O2CIN**	**OPA1 output to comparator input control bit** 0= disable 1= enable
[25]	**A0O2CIN**	**OPA0 output to comparator input control bit** 0= disable 1= enable
[24]	**LPWREN**	**Enable Opamps in STOP/SPD modes** 0 = disable. 1 = enable.
[23]	**A0O2A1N**	**OPA0 Output to OPA0 Inverting Input Control Bit** 0 = disable. 1 = enable.
[22]	**A0O2A1P**	**OPA0 Output to OPA1 Non-inverting Input Control Bit** 0 = disable. 1 = enable.
[21]	**Reserved**	Reserved
[20]	**VREFEN**	**Enable OPA and Comparator Reference Voltage Generator** 0 = disable. 1 = enable.
[19]	**PGAEN**	**OPA1 PGA Gain Enable Control Bits** 0 = disable. 1 = Enable.

[18:16]	**PGA**	**OPA1 Gain Control Bits** 000 = 1. 001 = 8. 010 = 16. 011 = 24. 100 = 32. 101 = 40. 110 = 48. 111 = 56.
[15]	**A1X**	**Operational amplifier 1 output; positive logic** This bit is read only
[14]	**A102N**	**OPA1 Output to OPA1 Inverting Input Control Bit** 0 = disable. 1 = enable.
[13:12]	**A1PSEL**	**OPA1 Non-inverting Input Selection Bit** 00 = no connection. 01 = from VH1 (0.9×VDDA). 10 = from VM (0.5×VDDA). 11 =: from VL1 (0.1×VDDA).
[11]	**A1PS**	**A1P Pin to OPA1 Non-inverting Input Control Bit** 0 = no connection. 1 = from A0P pin.
[10]	**A1NS**	**A1N Pin to OPA1 Inverting Input Control Bit** 0 = no connection. 1 = from A0N pin.
[9]	**A1OEN**	**OPA1 Output Enable or Disable Control Bit** 0 = disable. 1 = enable.
[8]	**A1EN**	**OPA1 Enable or Disable Control Bit** 0 = disable. 1 = enable.
[7]	**A0X**	**Operational amplifier 0 output; positive logic** This bit is read only
[6]	**A0O2N**	**OPA0 Output to OPA0 Inverting Input Control Bit** 0 = disable. 1 = enable.
[5:4]	**A0PSEL**	**OPA0 Non-inverting Input Selection Bit** 00 = no connection. 01 = from VH1 (0.9×VDDA). 10 = from VM (0.5×VDDA). 11 =: from VL1 (0.1×VDDA).
[3]	**A0PS**	**A0P Pin to OPA0 Non-inverting Input Control Bit** 0 = no connection. 1 = from A0P pin.

[2]	**A0NS**	**A0N Pin to OPA0 Inverting Input Control Bit** 0 = no connection. 1 = from A0N pin.
[1]	**A0OEN**	**OPA0 Output Enable or Disable Control Bit** 0 = disable. 1 = enable.
[0]	**A0EN**	**OPA0 Enable or Disable Control Bit** 0 = disable. 1 = enable.

Comparator Control Register (CSCAN_CMPCTL)

Register	Offset	R/W	Description	Reset Value
CSCAN_CMPCTL	CSCAN_BA+0x18	R/W	Comparator Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							LPWREN
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						C2OUT	C1OUT
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPES		Reserved	Reserved	C2INTEN	C2OUTEN	C2PSEL	CMP2EN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNPSEL	Reserved	Reserved	CMP_INT	C1INTEN	C1OUTEN	C1NSEL	CMP1EN

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**LPWREN**	**Comparator Low power mode enable** If ‘1’ comparator will remain enabled in STOP/SPD power modes.
[23:18]	**Reserved**	Reserved.
[17]	**C2OUT**	**Comparator 2 Output.** Real time readback of comparator 2.
[16]	**C1OUT**	**Comparator 1 Output.** Real time readback of comparator 1.
[15:14]	**CMPES**	**Interrupt edge control bits** 00=disable 01= rising edge trigger 10= falling edge trigger 11= dual edge trigger
[13:12]	**Reserved**	Reserved.
[11]	**C2INTEN**:	**Comparator 2 interrupt control** 0= disable 1= enable
[10]	**C2OUTEN**	**Comparator 2 output pin control bit** 0= disable 1= enable
[9]	**C2PSEL**:	**Comparator 2 inverting input control** 0= from VL0 1= from C2P pin

[8]	**CMP2EN**:	**Comparator 2 enable or disable control** 0= disable 1= enable
[7]	**CNPSEL**:	**Comparator non-inverting input control** 0= from OPA output 1= from CNP pin
[6:5]	**Reserved**	**Reserved**
[4]	**CMP_INT**	**Comparator Interrupt.** **Set by harddware.** **Write 1 to clear.**
[3]	**C1INTEN**:	**Comparator 1 interrupt control** 0= disable 1= enable
[2]	**C1OUTEN**	**Comparator 1 output pin control bit** 0= disable 1= enable
[1]	**C1NSEL**	**Comparator 1 inverting input control** 0= from VH0 1= from C1N pin
[0]	**CMP1EN**	**Comparator 1 enable or disable control** 0= disable 1= enable

5.24 Biquad Filter (BIQ)

5.24.1 Overview and Features

A coefficient programmable 6-stage Biquad filter (12th-Order IIR filter) is available which can be used on either SDADC path or DPWM path to further reduce unwanted noise or filter the signal. 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.BIQEN). 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_CTL.PATHSEL register bit determines which path the BIQ is going to use. The default value is 0 which is the microphone ADC path, by setting this bit 1, the BIQ will be used in DPWM path.

If the BIQ is intended to be used in DPWM path, the BIQ can up sample the data rate by programming BIQ_CTL.DPWMPUSR register which has default value at 1.

If the BIQ is intended to be used in ADC path, 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. Setup the BIQ operation sample rate by program DPWMPUSR or SRDIV register bits if necessary.
4. Decide the SDADC or DPWM path to be used for the BIQ by programming PATHSEL, and turn off PRGCOEFF bit (if it was turned on in step #2).
5. Setup BIQ stage (BIQ_CTL.STAGE), set “1” BIQ with 5 stages, set “0” BIQ with 6 stages.
6. Setup high pass filter on or off (BIQ_CTL.HPFON. If HPF is ON, the BIQ Stage automatically set 6 stages, one for HPF).
7. Turn on BIQ_CTL.BIQEN, BIQ will start filter function.

Configuring Coefficient

1. BIQ function work on 6 stages, set BIQ_CTL.STAGE=0, BIQ function will call 30 coefficients from BIQ_BA+0x0 ~0x074
2. BIQ function work on 5 stages, set BIQ_CTL.STAGE=1, BIQ function will call 25 coefficients from BIQ_BA+0x00 ~0x060
3. If High pass filter is on (BIQ_CTL.HPFON=1), BIQ function automatically set 6 stages, one for HPF. 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

5.24.2 Register Map

R: read only, **W:** write only, **R/W:** both read and 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
BIQ_STS	BIQ_BA+0x084	R/W	BIQ status Register	0x8000_0000

5.24.3 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							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
COEFFDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
COEFFDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COEFFDAT							

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			SRDIV				
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SRDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				STAGE	DPWMPUSR		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRGCOEFF	SDADCWNSR			DLCOEFF	PATHSEL	HPFON	BIQEN

Bits	Description	
[31:12]	**Reserved**	Reserved
[28:16]	**SRDIV**	**SR Divider**
[15:12]	**Reserved**	Reserved
[11]	**STAGE**	**BIQ Stage Number Control** 0 = 6 stage. 1 = 5 stage.
[10:8]	**DPWMPUSR**	**DPWM Path Up Sample Rate (From SRDIV Result)** 0001 --- up 1x (no up sample) 0010 --- up 2x 0011 --- up 3x 0100 --- up 4x 0110 --- up 6x Others reserved
[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 is on.
[6:4]	**SDADCWNSR**	**SDADC Down Sample** 001--- 1x (no down sample) 010 --- 2x 011 --- 3x 100 --- 4x 11 0--- 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]	**PATHSEL**	**AC Path Selection for BIQ** 0 = used in SDADC path. 1 = used in DPWM path.
[1]	**HPFON**	**High Pass Filter On** 0 = disable high pass filter. 1 = enable high pass filter. **Note :** If this register is on, BIQ only 5 stage left.for user. SDADC path sixth stage coefficient is for HPF filter coefficient. DPWM path first stage coefficient is for HPF filter coefficient.
[0]	**BIQEN**	**BIQ Filter Start to Run** 0 = BIQ filter is not processing. 1 = BIQ filter is on.

BIQ Status Register (BIQ_STATUS)

Register	Offset	R/W	Description	Reset Value
BIQ_STS	BIQ_BA+0x084	R/W	BIQ status Register	0x8000_0000

31	30	29	28	27	26	25	24
RAMINITF	**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]	**RAMINITF**	**Coefficient Ram Initial Default Done Flag** 0 = initial default value done. 1 = still working on.
[30:0]	**Reserved**	Reserved

5.25 Successive Approximation Analog-to-Digital Convertor (SARADC)

5.25.1 Overview and Features

- Analog input voltage range: 0~VREF
- Up to 12 single-end analog input channels
- Three operating modes
  - Single mode: A/D conversion is performed one time on a specified channel.
  - Single-cycle scan mode: A/D conversion is performed one cycle on all specified channels following the sequence from the lowest numbered channel to the highest numbered channel.
  - Continuous scan mode: A/D converter continuously performs Single cycle scan mode until software stops A/D conversion.
- An A/D conversion can be started by:
  - Writing 1 to SWTRG bit through software
  - External pin SARADC_TRIG (or called STADC in section)
- Conversion results are held in data registers for each channel with valid and overrun indicators
- Conversion result can be compared with specified value; can generate interrupt when conversion result matches the compare register setting.

5.25.2 Block Diagram

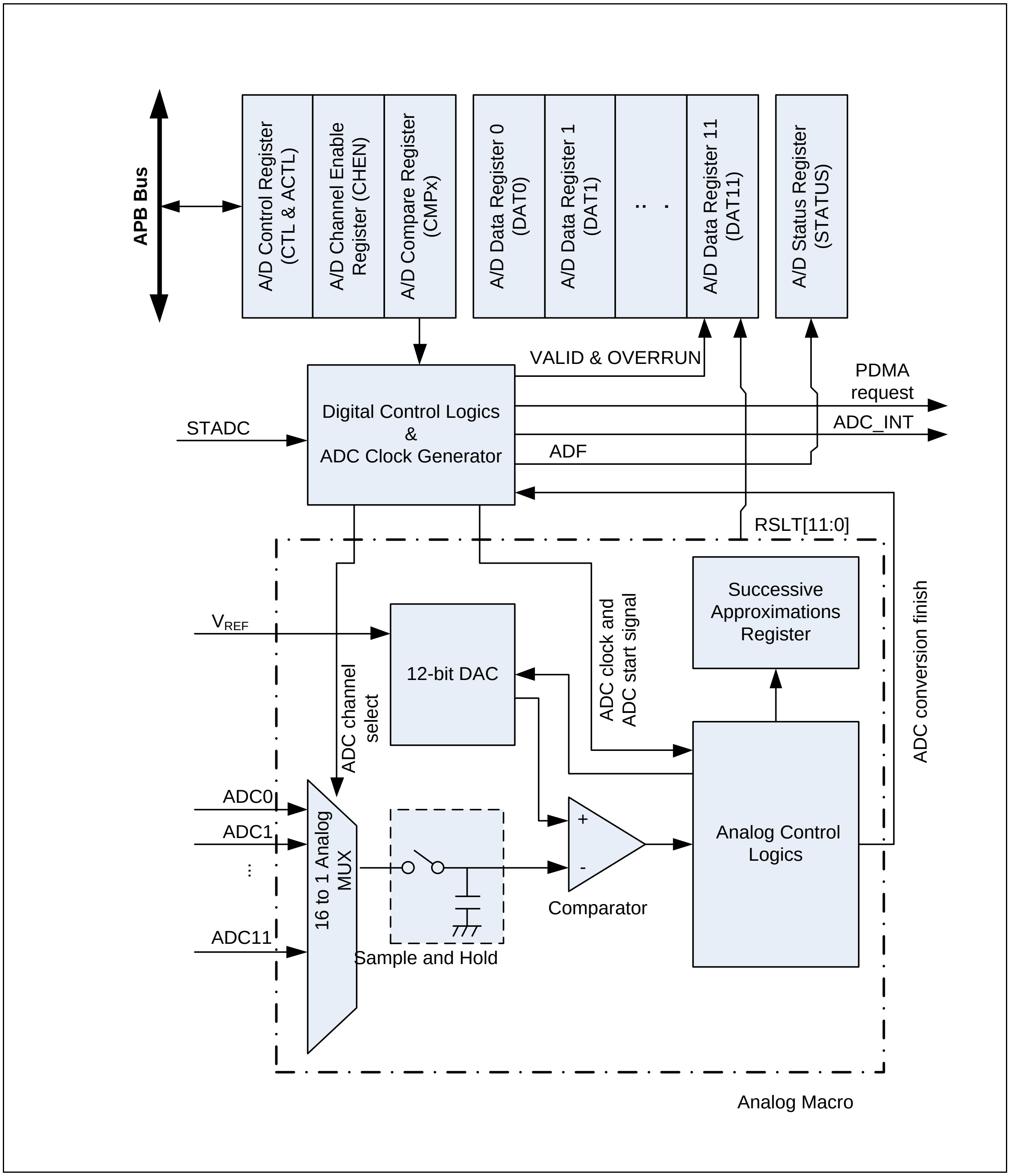

APB Bus
A/D Control Register (CTL & ACTL)
A/D Channel Enable Register (CHEN)
A/D Compare Register (CMPx)
A/D Data Register 0 (DAT0)
A/D Data Register 1 (DAT1)
. . .
A/D Data Register 11 (DAT11)
A/D Status Register (STATUS)
VALID & OVERRUN
PDMA request
ADC_INT
ADF
STADC
Digital Control Logics & ADC Clock Generator
RSLT[11:0]
Successive Approximations Register
VREF
12-bit DAC
ADC channel select
ADC clock and ADC start signal
ADC conversion finish
ADC0
ADC1
...
ADC11
16 to 1 Analog MUX
Sample and Hold
+
-
Comparator
Analog Control Logics
Analog Macro

Figure 5-107 SARADC Block Diagram

5.25.3 Function description

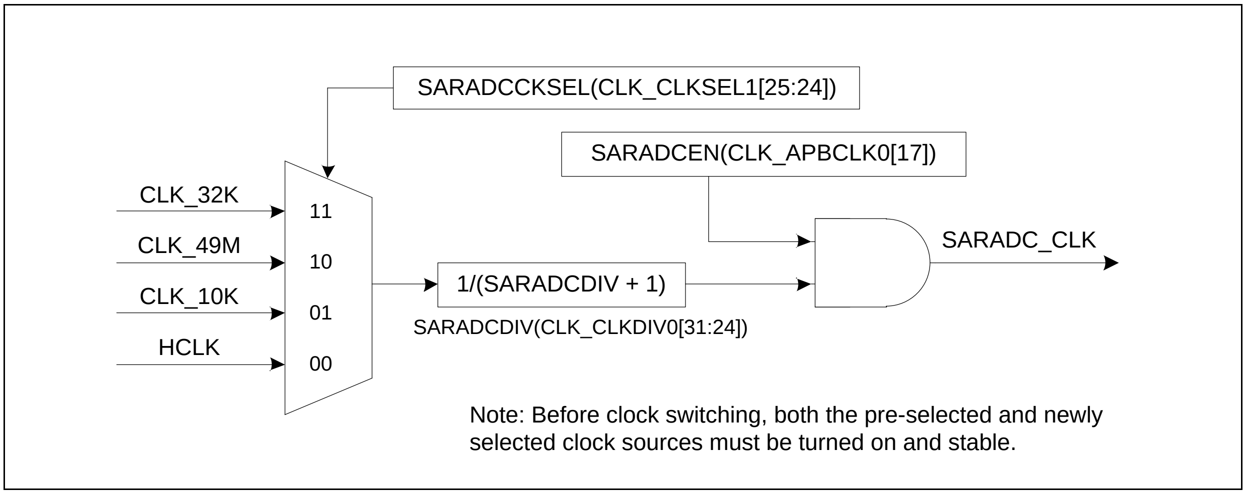
The A/D converter operates by successive approximation with 12-bit resolution. The SARADC had three operation modes: single mode, single-cycle scan mode and continuous scan mode. When changing the operation mode or analog input channel, to prevent incorrect operation, software must clear SWTRG bit to “0” in CTL register.

5.25.3.1 SARADC Clock Generator

The SARADC engine has four clock sources selected by 2 bits SARADCSEL, the SARADC clock divided by 8 bits prescaler with the formula.

The SARADC clock frequency = (SARADC clock source frequency)/(SARADCDIV+1).

Where the 8 bits SARADCDIV is located in register CLKDIV0[31:24].


SARADCCKSEL(CLK_CLKSEL1[25:24])
SARADCEN(CLK_APBCLK0[17])
CLK_32K
CLK_49M
CLK_10K
HCLK
11
10
01
00
1/(SARADCDIV + 1)
SARADCDIV(CLK_CLKDIV0[31:24])
SARADC_CLK

Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 5-108 SARADC Clock Generator

5.25.3.2 Single Mode

In single mode, A/D conversion is performed only once on the specified single channel. The operations are as follows:

1. A/D conversion will be started when the SWTRG bit of CTL is set to 1 by software.
2. When A/D conversion is finished, the result is stored in the A/D data register corresponding to the channel.
3. The ADEF bit of STATUS register will be set to 1. If the ADCIE bit of CTL register is set to 1, the SARADC interrupt will be asserted.
4. 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 idle state.

Note: If software enables more than one channel in single mode, the channel with the smallest number will be selected and the other enabled channels will be ignored.

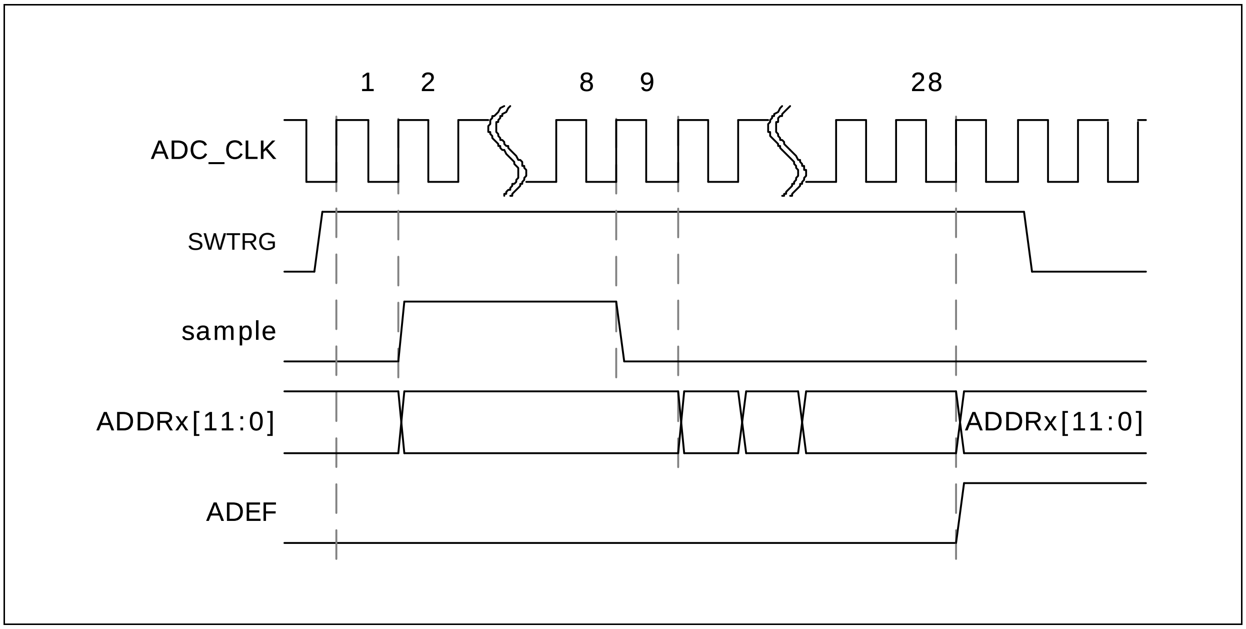


Figure 5-109 Single Mode Conversion Timing Diagram

5.25.3.3 Single-Cycle Scan Mode

In single-cycle scan mode, A/D conversion will sample and convert the specified channels once in the sequence from the smallest number enabled channel to the largest number enabled channel.

1. When the SWTRG bit of CTL is set to 1 by software or external trigger input, A/D conversion starts on the channel with the smallest number.
2. When A/D conversion for each enabled channel is completed, the result is sequentially transferred to the A/D data register corresponding to each channel.
3. When the conversions of all the enabled channels are completed, the ADEF bit in STATUS is set to 1. If the SARADC interrupt function is enabled, the SARADC interrupt occurs.
4. After A/D conversion ends, the SWTRG bit is automatically cleared to 0 and the A/D converter enters idle state. If SWTRG is cleared to 0 before all enabled SARADC channels conversion done, SARADC controller will finish current conversion and save the result to the DATx(ADDRx) of the current conversion channel.

An example timing diagram for single-cycle scan on enabled channels (0, 2, 3 and 7) is shown below:

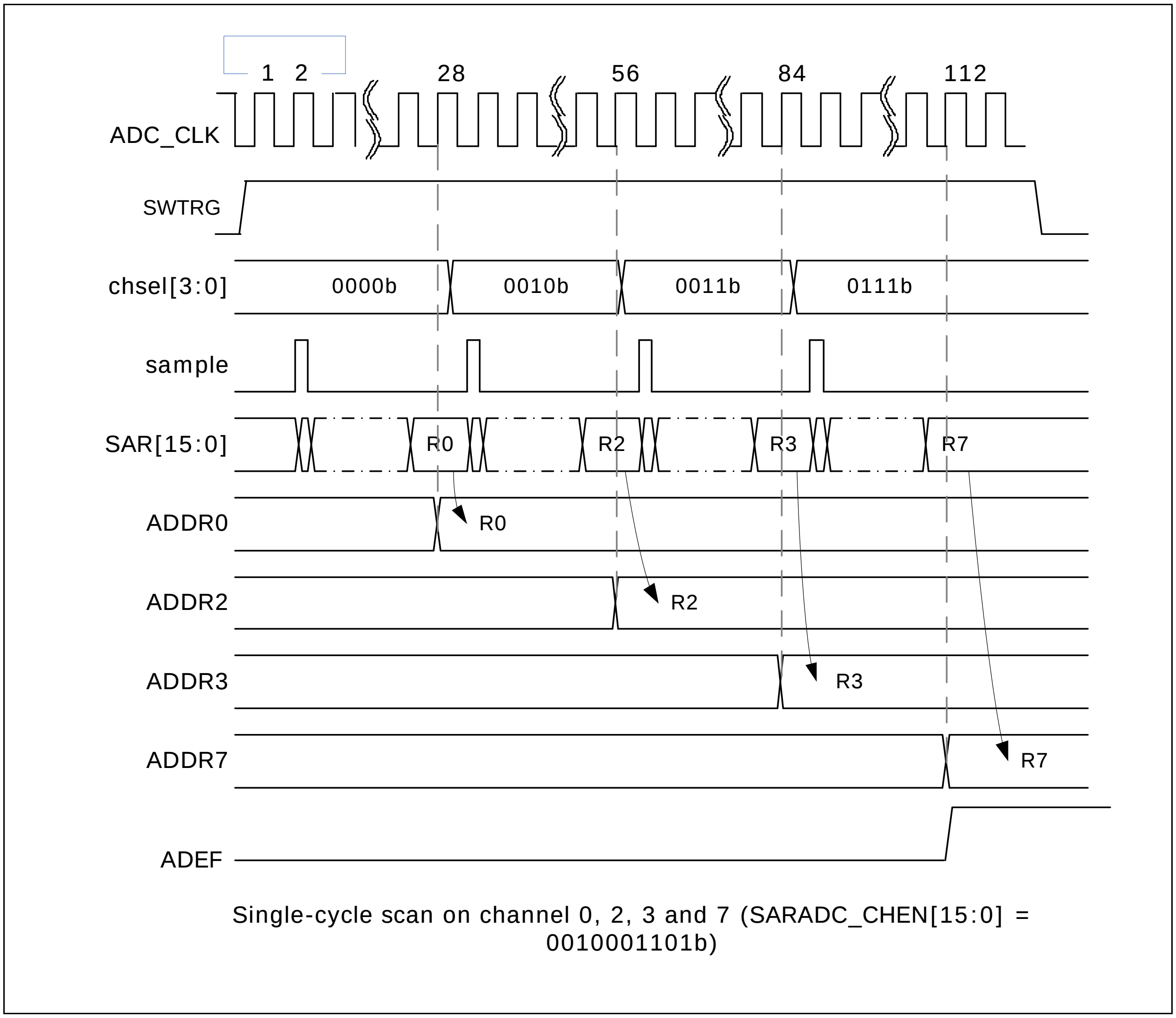


Figure 5-110 Single-Cycle Scan on Enabled Channels Timing Diagram

5.25.3.4 Continuous Scan Mode

In continuous scan mode, A/D conversion is performed sequentially on the specified channels that enabled by CHEN bits in CHEN register (maximum 12 channels for SARADC). The operations are as follows:

1. When the ADST bit in CTL is set to 1 by software, A/D conversion starts on the channel with the smallest number.
2. When A/D conversion for each enabled channel is completed, the result of each enabled channel is stored in the A/D data register corresponding to each enabled channel.
3. When A/D converter completes the conversions of all enabled channels sequentially, the ADEF bit (STATUS[0]) will be set to 1. If the SARADC interrupt function is enabled, the SARADC interrupt occurs. The conversion of the enabled channel with the smallest number will start again if software has not cleared the ADST bit.
4. As long as the ADST bit remains at 1, the step 2 ~ 3 will be repeated. When ADST is cleared to 0, SARADC controller will stop conversion.

An example timing diagram for continuous scan on enabled channels (0, 2, 3 and 7) is shown below:

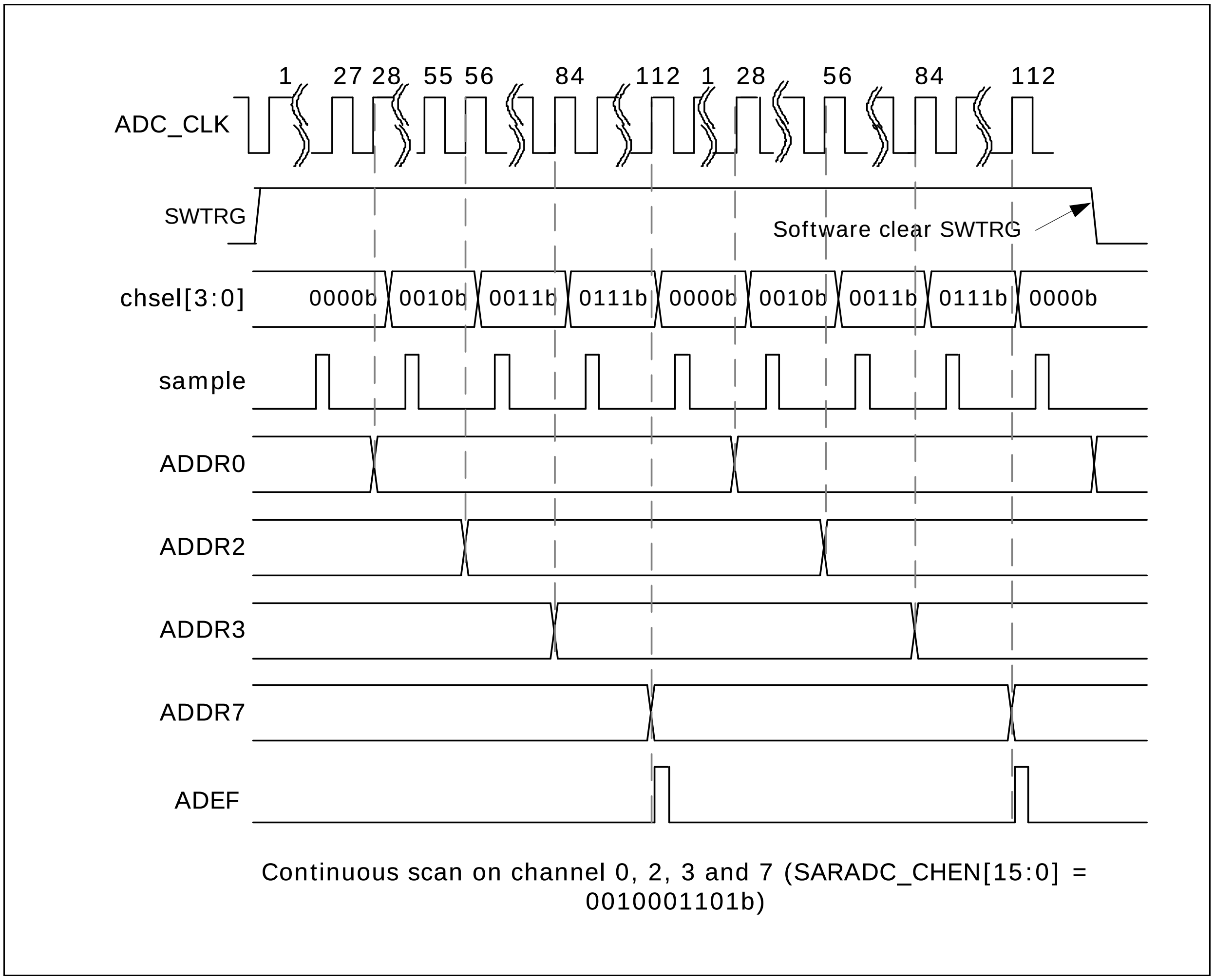


Figure 5-111 Continuous Scan on Enabled Channels Timing Diagram

5.25.3.5 External trigger Input Sampling and A/D Conversion Time

In Single-cycle scan mode, A/D conversion can be triggered by external pin request. When the CTL.HWTRGEN is set to high to enable SARADC external trigger function. setting the HWTRGSEL[1] bits to 0b is to select external trigger input from the STADC pin. Software can set HWTRGCOND[1:0] to select trigger condition is falling/rising edge or low/high level. If level trigger condition is selected, the STADC pin must be kept at defined state at least 8 PCLKs. The ADST bit will be set to 1 at the 9th PCLK and start to conversion. Conversion is continuous if external trigger input is kept at active state in level trigger mode. It is stopped only when external condition trigger condition disappears. If edge trigger condition is selected, the high and low state must be kept at least 4 PLCKs. Pulse that is shorter than this specification will be ignored.

5.25.3.6 Conversion Result Monitor by Compare Function

SARADC controller provide two sets of compare register SARADC_CMP0 and SARADC_CMP1, to monitor maximum two specified channels conversion result from A/D conversion controller, refer Figure… Software can select which channel to be monitored by set CMPCH(SARADC_CMPx[5:0]) and CMPCOND bit is used to check conversion result is less than specify value or greater than (equal to) value specified in CMPDAT[11:0]. When the conversion of the channel specified by CMPCH is completed, the comparing action will be triggered one time automatically. When the compare result meets the setting, compare match counter will increase 1, otherwise, the compare match counter will be cleared to 0. When counter value reach the setting of (CMPMCNT+1) then ADCMPF bit will be set to 1, if ADCMPIE bit is set then an ADC_INT interrupt request is generated. Software can use it to monitor the external analog input pin voltage transition in scan mode without imposing a load on software. Detailed logics diagram is shown below:

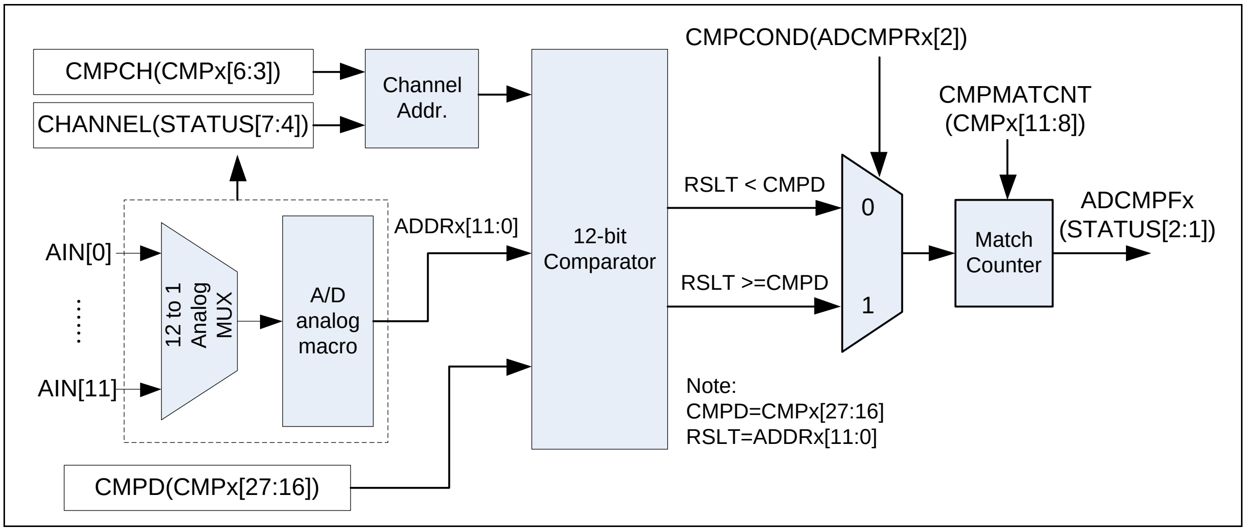


Figure 5-112 A/D Conversion Result Monitor Logics Diagram

5.25.3.7 *Interrupt Sources*

There are three interrupt sources of SARADC interrupt. When an SARADC operation mode finishes its conversion, the A/D conversion end flag, ADEF, will be set to 1. . The ADCMPF0 and ADCMPF1 are the compare flags of compare function. When the conversion result meets the settings of ADCMPR0/1, the corresponding flag will be set to 1. When one of the flags, ADEF, ADCMPF0 and ADCMPF1, is set to 1 and the corresponding interrupt enable bit, ADCIE of CTL and ADCMPIE of SARADC_CMP0/SARADC_CMP1, is set to 1, the SARADC interrupt will be asserted. Software can clear the flag to revoke the interrupt request.

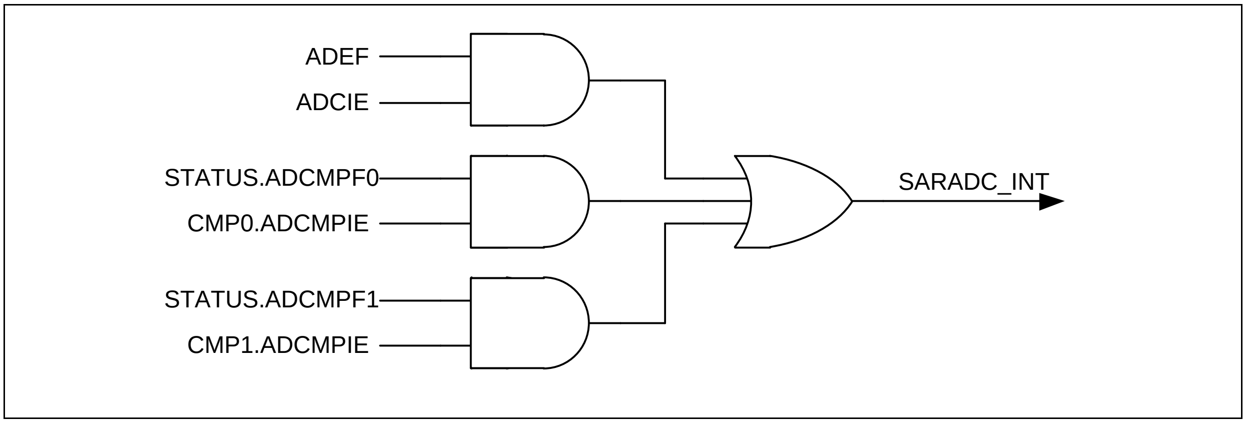


Figure 5-113 A/D Controller Interrupt

5.25.3.8 *Peripheral DMA Request*

When A/D conversion is finished, the conversion result will be loaded into DATx register and VALID bit will be set to 1. If the PTEN bit of CTL is set, SARADC controller will generate a request to PDMA. User can use PDMA to transfer the conversion results to a user-specified memory space without CPU's intervention. The source address of PDMA operation is fixed at the address of register PDMADAT no matter what channels was selected. When PDMA is transferring the conversion result, SARADC will continue converting the next selected channel if the operation mode of SARADC is single scan mode or continuous scan mode. User can monitor current PDMA transfer data through reading PDMADAT register. If SARADC completes the conversion of a selected channel and the last conversion result of the same channel has not been transferred by PDMA, overrun OV bit of the corresponding channel will be set and the last SARADC conversion result will be overwritten by the new SARADC conversion result. PDMA will transfer the latest data of selected channels to the user-specified destination address.

5.25.4 Register Map

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

Register	Offset	R/W	Description	Reset Value
SARADC Base Address: **SARADC_BA = 0x4006_0000**				
SARADC_DAT0	SARADC_BA+0x00	R	SAR ADC Data Register 0	0x0000_0000
SARADC_DAT1	SARADC_BA+0x04	R	SAR ADC Data Register 1	0x0000_0000
SARADC_DAT2	SARADC_BA+0x08	R	SAR ADC Data Register 2	0x0000_0000
SARADC_DAT3	SARADC_BA+0x0c	R	SAR ADC Data Register 3	0x0000_0000
SARADC_DAT4	SARADC_BA+0x10	R	SAR ADC Data Register 4	0x0000_0000
SARADC_DAT5	SARADC_BA+0x14	R	SAR ADC Data Register 5	0x0000_0000
SARADC_DAT6	SARADC_BA+0x18	R	SAR ADC Data Register 6	0x0000_0000
SARADC_DAT7	SARADC_BA+0x1c	R	SAR ADC Data Register 7	0x0000_0000
SARADC_DAT8	SARADC_BA+0x20	R	SAR ADC Data Register 8	0x0000_0000
SARADC_DAT9	SARADC_BA+0x24	R	SAR ADC Data Register 9	0x0000_0000
SARADC_DAT10	SARADC_BA+0x28	R	SAR ADC Data Register 10	0x0000_0000
SARADC_DAT11	SARADC_BA+0x2c	R	SAR ADC Data Register 11	0x0000_0000
SARADC_STATUS	SARADC_BA+0x40	R/W	SAR ADC status Register	0x0000_0000
SARADC_PDMADAT	SARADC_BA+0x50	R	SAR ADC PDMA Current Transfer Data	0x0000_0000
SARADC_ACTL	SARADC_BA+0x5C	R/W	SAR ADC analog control register	0x0000_0000
SARADC_CTL	SARADC_BA+0x60	R/W	SAR ADC Control Register	0x0000_0000
SARADC_CHEN	SARADC_BA+0x64	R/W	SAR ADC Channel Enable Register	0x0000_0000
SARADC_CMP0	SARADC_BA+0x68	R/W	SAR ADC Compare Register 0	0x0000_0000
SARADC_CMP1	SARADC_BA+0x6C	R/W	SAR ADC Compare Register 1	0x0000_0000

5.25.5 Register description

SAR ADC Data Register (SARADC_DATx)

Register	Offset	R/W	Description	Reset Value
SARADC_DAT0	SARADC_BA+0x00	R	SAR ADC Data Register 0	0x0000_0000
SARADC_DAT1	SARADC_BA+0x04	R	SAR ADC Data Register 1	0x0000_0000
SARADC_DAT2	SARADC_BA+0x08	R	SAR ADC Data Register 2	0x0000_0000
SARADC_DAT3	SARADC_BA+0x0c	R	SAR ADC Data Register 3	0x0000_0000
SARADC_DAT4	SARADC_BA+0x10	R	SAR ADC Data Register 4	0x0000_0000
SARADC_DAT5	SARADC_BA+0x14	R	SAR ADC Data Register 5	0x0000_0000
SARADC_DAT6	SARADC_BA+0x18	R	SAR ADC Data Register 6	0x0000_0000
SARADC_DAT7	SARADC_BA+0x1c	R	SAR ADC Data Register 7	0x0000_0000
SARADC_DAT8	SARADC_BA+0x20	R	SAR ADC Data Register 8	0x0000_0000
SARADC_DAT9	SARADC_BA+0x24	R	SAR ADC Data Register 9	0x0000_0000
SARADC_DAT10	SARADC_BA+0x28	R	SAR ADC Data Register 10	0x0000_0000
SARADC_DAT11	SARADC_BA+0x2c	R	SAR ADC Data Register 11	0x0000_0000

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**
Reserved				RESULT			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RESULT							

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**Valid Flag (Read Only)** 0 = Data in RESULT[11:0] bits is not valid. 1 = Data in RESULT[11:0] bits is valid. **Note:** This bit is set to 1 when corresponding channel analog input conversion is completed and cleared by hardware after DAT register is read.

[16]	**OV**	**Overrun Flag (Read Only)** 0 = Data in RESULT[11:0] is recent conversion result. 1 = Data in RESULT[11:0] is overwritten. **Note:** If converted data in RESULT[11:0] has not been read before new conversion result is loaded to this register, OV is set to 1 and previous conversion result is gone. It is cleared by hardware after DAT register is read.
[15:12]	**Reserved**	Reserved.
[11:0]	**RESULT**	**A/D Conversion Result** This field contains conversion result of SARADC. 12-bit SARADC conversion result with unsigned format.

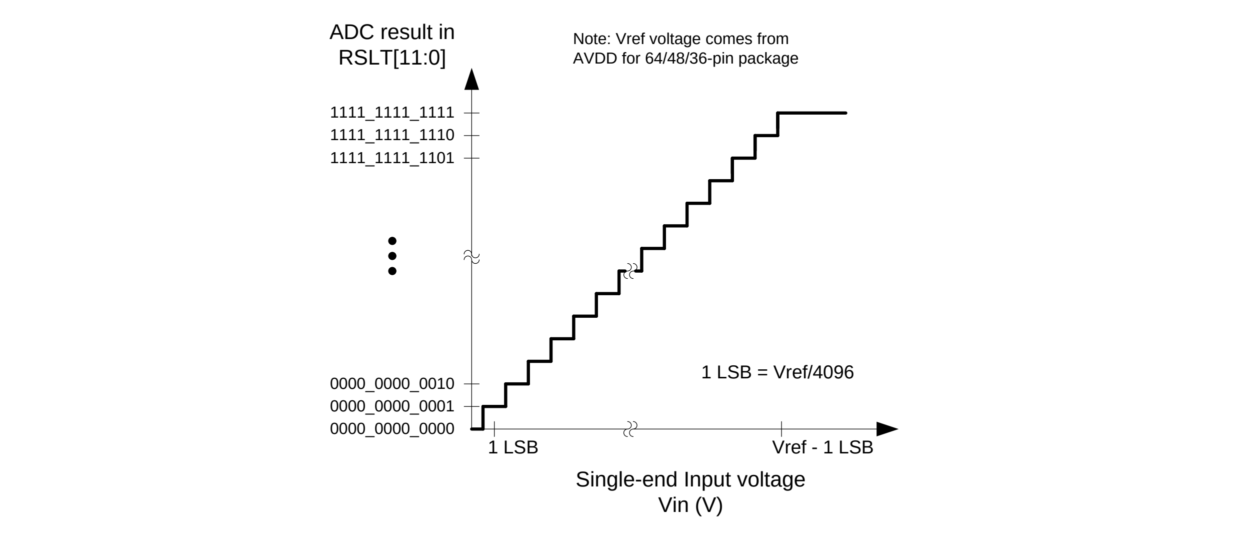


Figure 5-114 Conversion Result Mapping Diagram of Single-end Input

SAR ADC Status Register (SARADC_STATUS)

Register	Offset	R/W	Description	Reset Value
SARADC_STATUS	SARADC_BA+0x40	R/W	SAR ADC status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
VALID							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VALID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHANNEL				BUSY	ADCMPF1	ADCMPF0	ADEF

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:8]	**VALID**	**Data Valid Flag (Read Only)** It is a mirror of VALID bit in DATx.
[7:4]	**CHANNEL**	**Current Conversion Channel (Read Only)** This field reflects the current conversion channel when BUSY = 1. When BUSY = 0, it shows the number of the next converted channel.
[3]	**BUSY**	**BUSY/IDLE (Read Only)** 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion. This bit is mirror of as SWTRG bit in CTL.
[2]	**ADCMPF1**	**Compare Flag** When the selected channel A/D conversion result meets setting condition in SARADC_CMP1 then this bit is set to 1. And it is cleared by writing 1 to self. 0 = Conversion result in DAT register does not meet CMP1 register. 1 = Conversion result in DAT register meets CMP1 register.
[1]	**ADCMPF0**	**Compare Flag** When the selected channel A/D conversion result meets setting condition in SARADC_CMP0 then this bit is set to 1. And it is cleared by writing 1 to self. 0 = Conversion result in DAT register does not meet CMP0 register. 1 = Conversion result in DAT register meets CMP0 register.
[0]	**ADEF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. ADEF is set to 1 at these two conditions: 1. When A/D conversion ends in Single mode. 2. When A/D conversion ends on all specified channels in Scan mode. **Note:** This bit can be cleared by writing ‘1’ to it.

SAR ADC PDMA Current Transfer Data Register (SARADC_PDMADAT)

Register	Offset	R/W	Description	Reset Value
SARADC_PDMADAT	SARADC_BA+0x50	R	SAR ADC PDMA Current Transfer Data	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:0]	**DATA**	**SAR ADC PDMA Current Transfer Data Register (Read Only)** When PDMA transferring, read this register can monitor current PDMA transfer data. Current PDMA transfer data is the content of DAT0 ~ DAT11. If PDMA word length = 32, data including Reserved bits, OV bit and VALID bit is moved If PDMA word length = 16, only AD conversion result is moved.

SAR ADC Analog Control Register (SARADC_ACTL)

Register	Offset	R/W	Description	Reset Value
SARADC_ACTL	SARADC_BA+0x5C	R/W	SAR ADC analog control register	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	reserved
[18]	**SAR_VREF**	**VREF selection** 0 -- select VCCA as VREF 1 -- select MICBIAS as VREF (MICBIAS shold be power down in I91200B series)
[17]	**Reserved**	Reserved
[16]	**Reserved**	Reserved
[15:1]	**Reserved**	Reserved
[0]	**SAR_SE_MODE**	Have to be 1

SAR ADC Control Register (SARADC_CTL)

Register	Offset	R/W	Description	Reset Value
SARADC_CTL	SARADC_BA+0x60	R/W	SAR 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				**SWTRG**	**Reserved**	**PDMAEN**	**HWTRGEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HWTRGCOND		**HWTRGSEL**		**OPMODE**		**ADCIE**	**ADCEN**

Bits	Description	
[31:12]	**Reserved**	Reserved
[11]	**SWTRG**	**A/D Conversion Start** 0 = Conversion stops and A/D converter enter idle state. 1 = Conversion starts. SWTRG bit can be set to 1 from three sources: software, external pin STADC. SWTRG will be cleared to 0 by hardware automatically at the ends of single mode and single-cycle scan mode. In continuous scan mode, A/D conversion is continuously performed until software writes 0 to this bit or chip reset.
[10]	**Reserved**	Reserved
[9]	**PDMAEN**	**PDMA Transfer Enable Bit** 0 = PDMA data transfer Disabled. 1 = PDMA data transfer in DAT 0~11 Enabled. When A/D conversion is completed, the converted data is loaded into DAT 0~11, software can enable this bit to generate a PDMA data transfer request. When PDMA=1, software must set ADCIE=0 to disable interrupt.
[8]	**HWTRGEN**	**Hardware Trigger Enable Bit** Enable or disable triggering of A/D conversion by hardware (external STADC pin or PWM Center-aligned trigger). 0 = Disabled. 1 = Enabled. SARADC hardware trigger function is only supported in single-cycle scan mode. If hardware trigger mode, the SWTRG bit can be set to 1 by the selected hardware trigger source.

[7:6]	**HWTRGCOND**	**External Trigger Condition** These two bits decide external pin STADC trigger event is level or edge. The signal must be kept at stable state at least 8 PCLKs for level trigger and 4 PCLKs at high and low state for edge trigger. 00 = Low level. 01 = High level. 10 = Falling edge. 11 = Rising edge.
[5:4]	**HWTRGSEL**	**Hardware Trigger Source Selection** 00 = A/D conversion is started by external STADC pin. Others = Reserved. Software should disable TRGEN and SWTRG before change HWTRGSEL.
[3:2]	**OPMODE**	**A/D Converter Operation Mode** 00 = Single conversion. 01 = Reserved. 10 = Single-cycle scan. 11 = Continuous scan. When changing the operation mode, software should disable SWTRG bit firstly.
[1]	**ADCIE**	**A/D Interrupt Enable Bit** 0 = A/D interrupt function Disabled. 1 = A/D interrupt function Enabled. A/D conversion end interrupt request is generated if ADCIE bit is set to 1.
[0]	**ADCEN**	**A/D Converter Enable Bit** 0 = Disabled. 1 = Enabled. Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit for saving power consumption.

SAR ADC Channel Enable Register (SARADC_CHEN)

Register	Offset	R/W	Description	Reset Value
SARADC_CHEN	SARADC_BA+0x64	R/W	SAR ADC Channel 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**
CHEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHEN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CHEN**	**Analog Input Channel Enable Bit** Set CHEN[11:0] to enable the corresponding analog input channel 11 ~ 0. 0 = SARADC input channel Disabled. 1 = SARADC input channel Enabled. Note: Keep 0 for [15:12]

SAR ADC Compare Register 0/1 (SARADC_CMP0/SARADC_CMP1)

Register	Offset	R/W	Description	Reset Value
SARADC_CMP0	SARADC_BA+0x68	R/W	SAR ADC Compare Register 0	0x0000_0000
SARADC_CMP1	SARADC_BA+0x6C	R/W	SAR ADC Compare Register 1	0x0000_0000

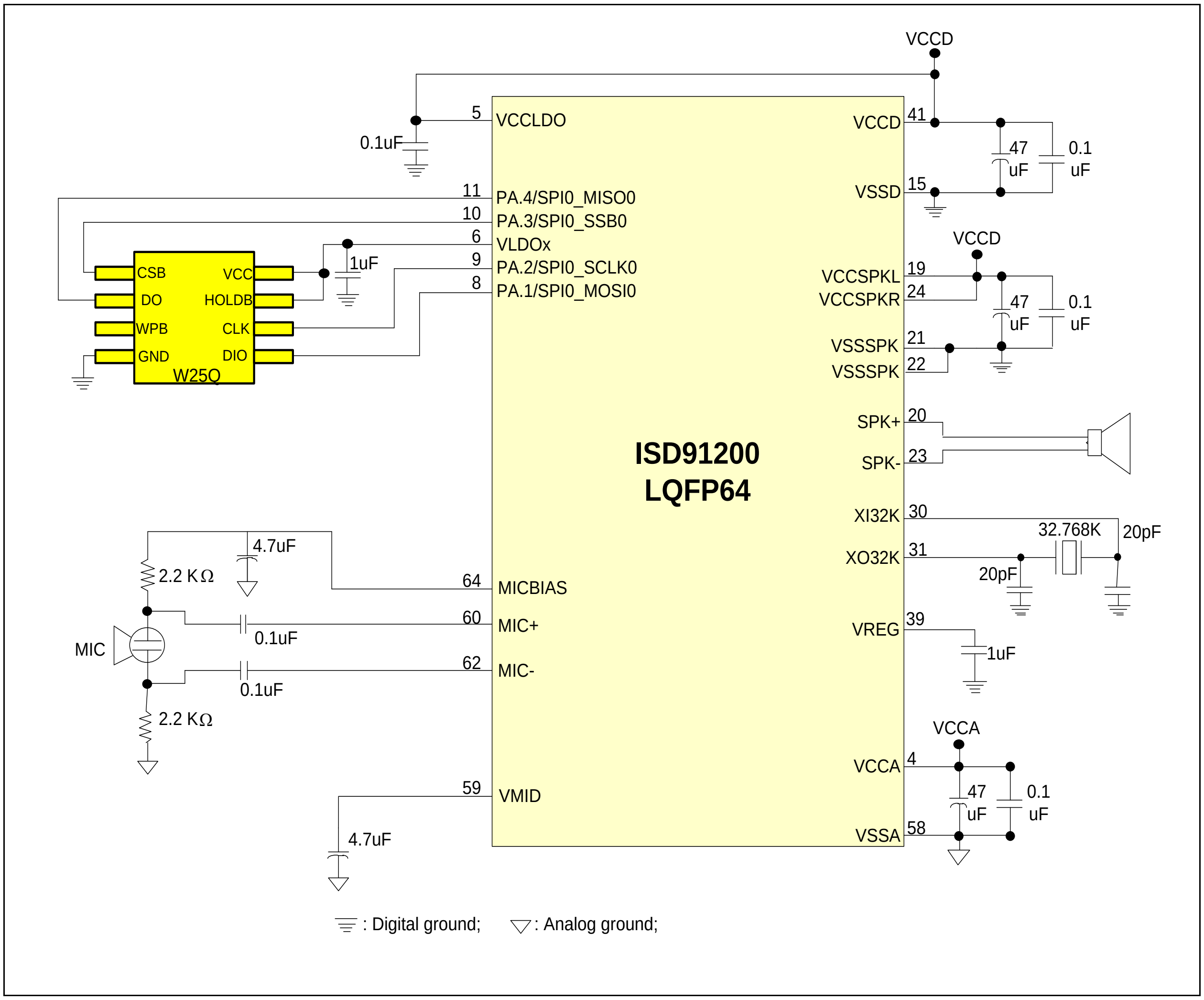
31	30	29	28	27	26	25	24
Reserved				**CMPDAT**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CMPMCNT**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CMPCH**				**CMPCOND**	**ADCMPIE**	**ADCMPEN**

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**CMPDAT**	**Comparison Data** The 12-bit data is used to compare with conversion result of specified channel. When ADCFMbit is set to 0, SARADC comparator compares CMPDAT with conversion result with unsigned format. CMPDAT should be filled in unsigned format. When ADCFMbit is set to 1, SARADC comparator compares CMPDAT with conversion result with 2'complement format. CMPDAT should be filled in 2'complement format.
[15:12]	**Reserved**	Reserved.
[11:8]	**CMPMCNT**	**Compare Match Count** When the specified A/D channel analog conversion result matches the compare condition defined by CMPCOND[2], the internal match counter will increase 1. When the internal counter reaches the value to (CMPMCNT +1), the ADCMPFx bit will be set.
[7]	**Reserved**	Reserved.

[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. 1100 = Reserved. 1101 = Reserved. 1110 = Reserved. 1111 = Reserved
[2]	**CMPCOND**	**Compare Condition** 0 = Set the compare condition as that when a 12-bit A/D conversion result is less than the 12-bit CMPDAT (CMPx[27:16]), the internal match counter will increase one. 1 = Set the compare condition as that when a 12-bit A/D conversion result is greater or equal to the 12-bit CMPD (CMPx[27:16]), the internal match counter will increase one. **Note:** When the internal counter reaches the value to (CMPMCNT +1), the ADCMPFx bit will be set.
[1]	**ADCMPIE**	**Compare Interrupt Enable Bit** 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled. **Note:** If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMCNT, ADCMPF bit will be asserted, in the meanwhile, if ADCMPIE is set to 1, a compare interrupt request is generated.
[0]	**ADCMPEN**	**Compare Enable Bit** 0 = Compare function Disabled. 1 = Compare function Enabled. **Note:** Set this bit to 1 to enable SARADC controller to compare CMPDAT[11:0] with specified channel conversion result when converted data is loaded into DAT register.

6 APPLICATION DIAGRAM

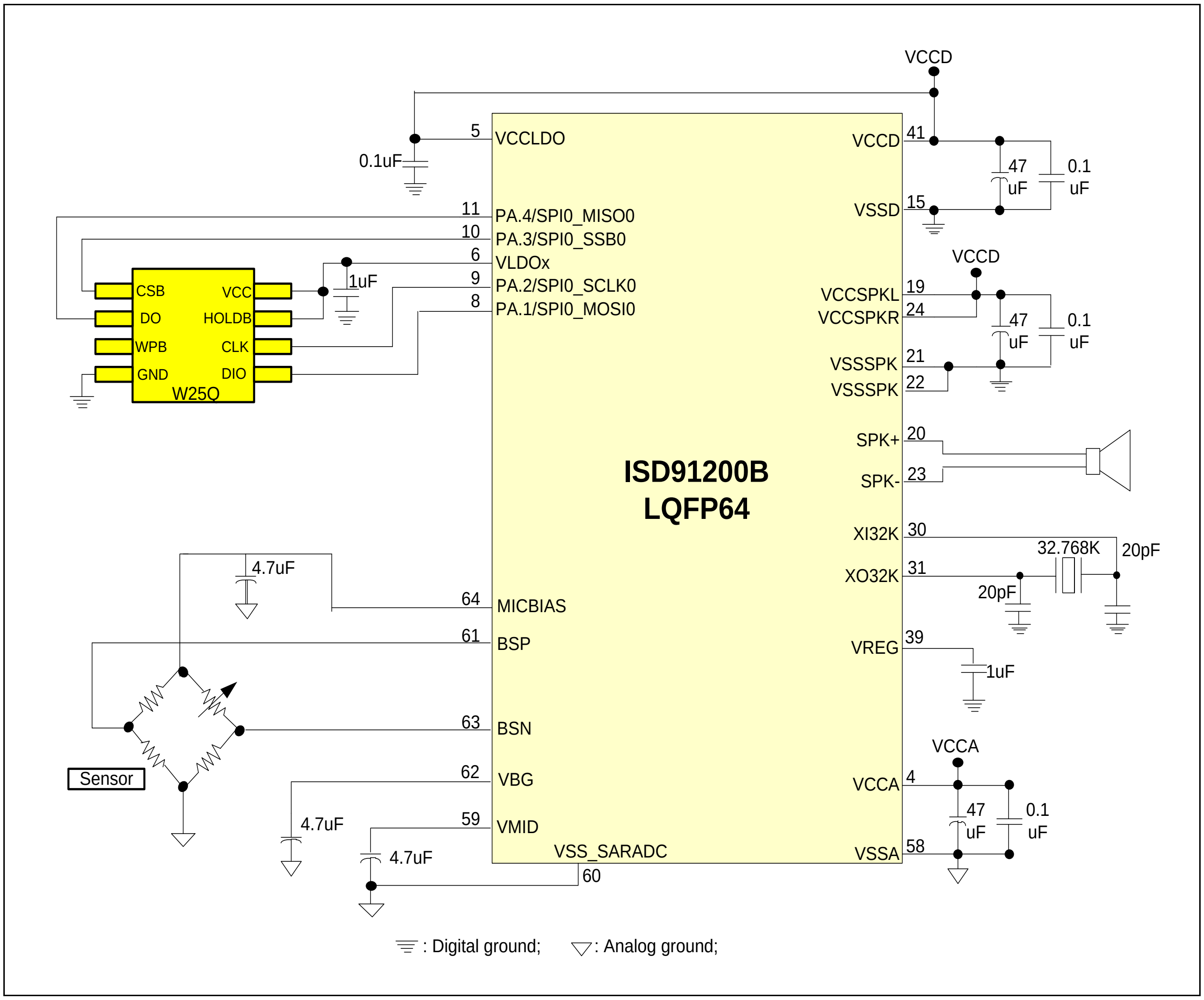
6.1 ISD91200 series (Non Bridge Sense)



Note:

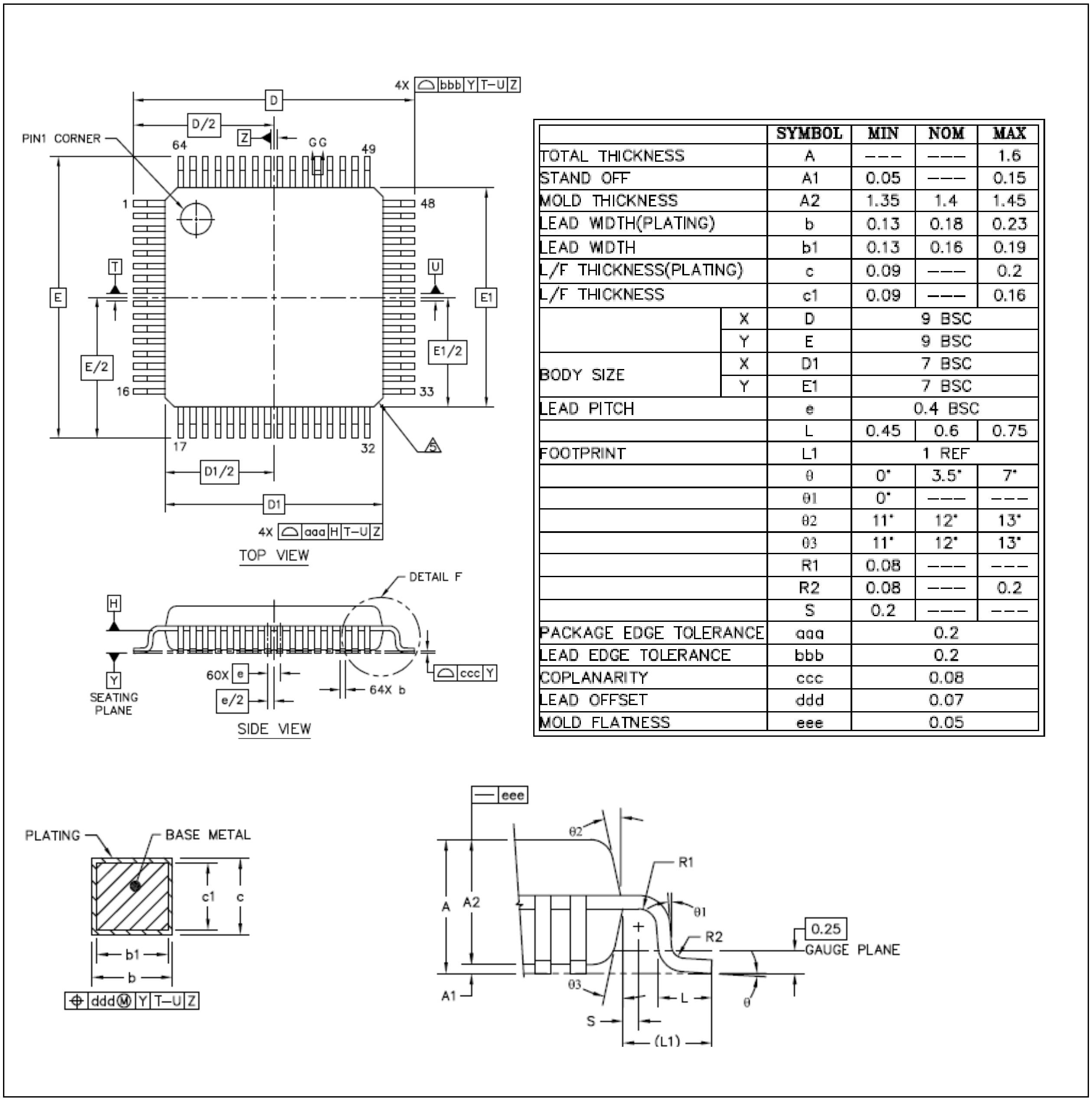
1. For SPI flash quad mode access, disconnect HOLDB & WPB from VDDB, and then connect HOLDB to PA.0 and WPB to PA.5.
2. Cannot use AMIC and DMIC at the same time, only one can be used. DMIC_CLK and DMIC_DAT can be set with different GPIO share pins.
3. DMIC operating voltage may be different as ISD91200 series. Customer have to care the voltage and data high/low level for ISD91200 recognition.
4. No MIC function in ISD91200P series
5. No MIC & SPK function in ISD91200G series

6.2 ISD91200B series (Bridge Sense)



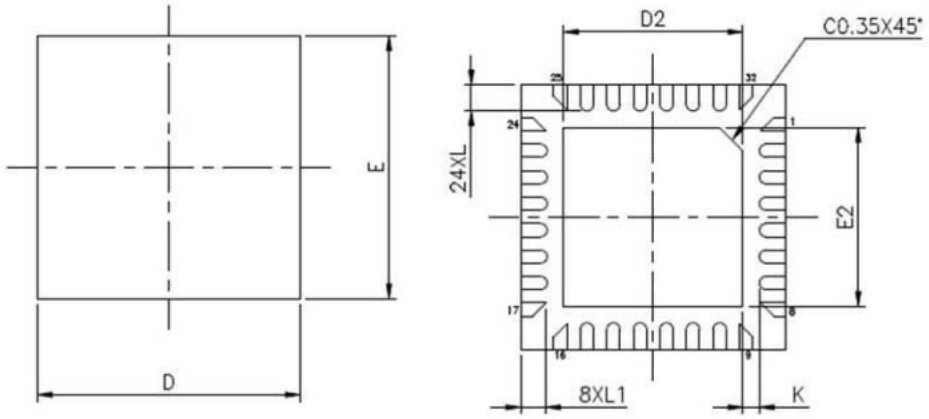
7 PACKAGE DIMENSIONS

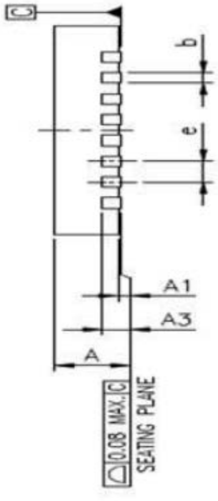
7.1 64L LQFP (7x7x1.4mm footprint 2.0mm)



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.13	0.18	0.23
LEAD WIDTH		b1	0.13	0.16	0.19
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
BODY SIZE	X	D		9 BSC	
	Y	E		9 BSC	
	X	D1		7 BSC	
	Y	E1		7 BSC	
LEAD PITCH		e		0.4 BSC	
		L	0.45	0.6	0.75
FOOTPRINT		L1		1 REF	
		θ	0°	3.5°	7°
		θ1	0°	---	---
		θ2	11°	12°	13°
		θ3	11°	12°	13°
		R1	0.08	---	---
		R2	0.08	---	0.2
		S	0.2	---	---
PACKAGE EDGE TOLERANCE		aaa		0.2	
LEAD EDGE TOLERANCE		bbb		0.2	
COPLANARITY		ccc		0.08	
LEAD OFFSET		ddd		0.07	
MOLD FLATNESS		eee		0.05	

7.2 QFN 32L 4X4 mm2, Thickness: 0.8mm (Max), Pitch: 0.40mm

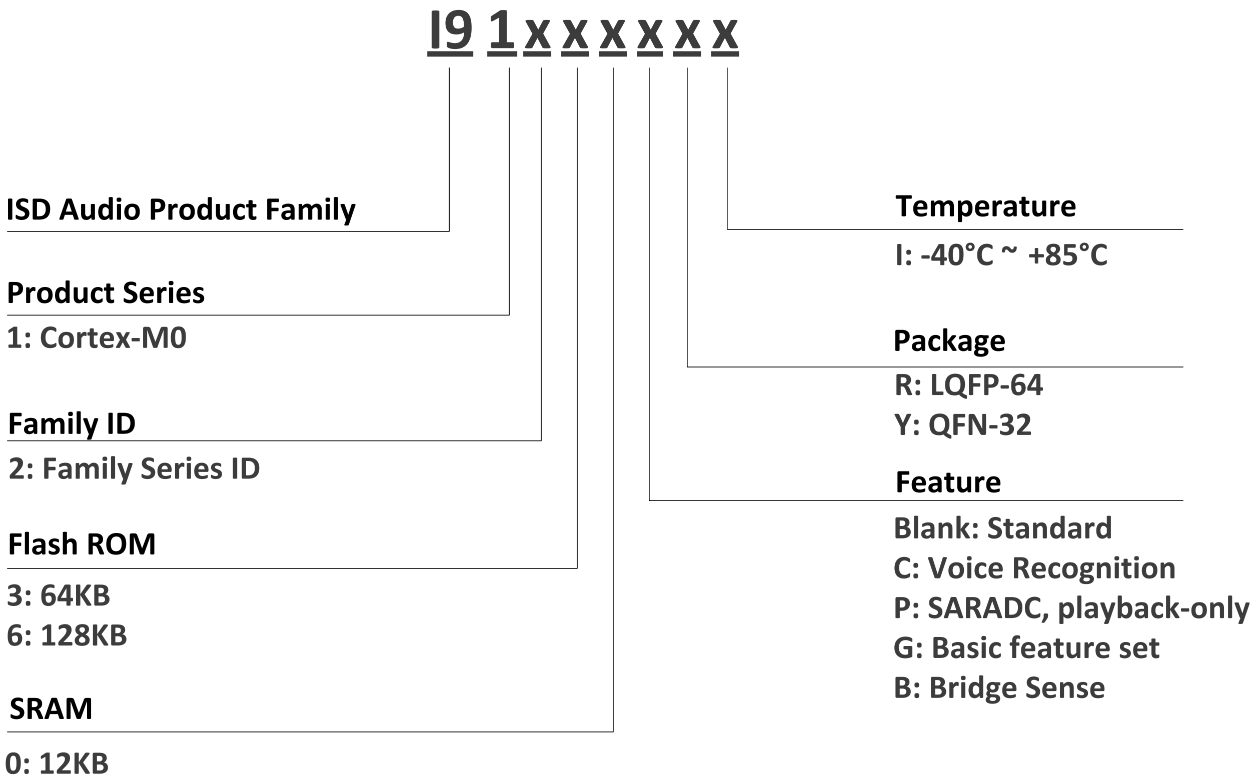




	Symbol	Min	Nom	Max
Total Thickness	A	0.7	0.75	0.8
Stand Off	A1	0.0	0.02	0.05
L/F Thickness	A3	0.2 REF.		
Lead Width	b	0.15	0.20	0.25
Body Size	D	4.00 BSC		
	E	4.00 BSC		
Lead Pitch	e	0.40 BSC		
Lead Length	L	0.35	0.4	0.45
	K	0.2	--	--

Symbol	L/F SIZE	Dimension in mm		
		MIN.	NOR.	MAX.
D2	114X114 mm	2.6	2.7	2.75
E2		2.6	2.7	2.75

8 ORDERING INFORMATION



9 REVISION HISTORY

VERSION	DATE	PAGE/ CHAP.	DESCRIPTION
V0.2	Sep 1, 2017	-	Preliminary Release.
V0.21	Sep 5, 2017		Add CSCAN feature and operation, fix Table & Figure reference errors
V0.22	Sep 7, 2017		Add SAR_VREF bit and description
V0.23	Sep 26, 2017		- Rename flash sector size as page size - Power distruibution diagram revised - OPA diagram revised
V2.0	Oct 17, 2017		Add figure number in SARADC, change clock source naming, formal release
V2.1	Nov 13, 2017		SPI0 and some typo corrected
V2.2	Dec 14, 2017		- Add analog pin function in pin description - Add more part feature in ordering information
V2.3	Jan 8, 2018		- Modify SARADC maximum SPS to 700K - Remove the DINBYPS - Add description for DINEDGE - Add DMIC circuit in Application Diagram
V2.4	Sep 16, 2019		- Changed cover title - Changed header title
V2.5	Mar. 03, 2020		- Add section 7.2 for Bridge Sense function - Section 3.1 Pin Configuration and section 3.2 Pin Description are revised - Revise the section 4 Block Diagram
V2.6	Apr. 10, 2020		- Change pin name MICBIAS to BSLDO for I91200B series - Add bridge sense option in AUDIOPATHSEL - Revise SDADC_DAT from 16 bits to 32 bits - SDADC_CTL revision - Add note for **ANA_MICBEN** power down in I91200B series
V2.7	Apr. 23, 2020		- Revise CONFIG0 register description - Revise BODCTL description - Revise SDADC Clock Generator figure and SD_CLK, CLKDIV & DSR description - Revise Table number
V2.8	June 07, 2020		- Change BISTEN to reserved (not for user) - Revise CLKDIV0 description - More SDADC FIFO notes
V2.9	Mar. 4, 2023		- Added “Package is Halogen-free, RoHS-compliant and TSCA-compliant” in section 2. - Updated format

Important Notice

Nuvoton products are not designed, intended, authorized or warranted for use as components in systems or equipment intended for surgical implantation, atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, or for other applications intended to support or sustain life. Furthermore, Nuvoton products are not intended for applications wherein failure of Nuvoton products could result or lead to a situation where personal injury, death or severe property or environmental damage could occur.

Nuvoton customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nuvoton for any damages resulting from such improper use or sales.

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.