ISD ARM® Cortex®-M0 SoC

ISD9100 Series

Technical Reference Manual

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

Table of Contents-

1 GENERAL DESCRIPTION

The ISD9100 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 ISD9100 embeds a Cortex®-M0 core running up to 49 MHz with 68/100/145K-byte 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, Analog Comparator, Low Voltage Detector and Brown-out detector.

The ISD9100 comes equipped with a rich set of power saving modes including a Deep Power Down (DPD) mode drawing less than 1μA. A micro-power 16KHz 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 10 μA.

For audio functionality the ISD9100 includes a Sigma-Delta ADC with 92dB SNR performance coupled with a Programmable Gain Amplifier (PGA) capable of a maximum gain of 61dB to enable direct connection of a microphone. Audio output is provided by a Differential Class D amplifier (DPWM) that can deliver 1W[1] of power to an 8Ω speaker.

The ISD9100 provides eight 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.

[1] We suggest implementing thermal protection by utilizing the Temperature Alarm; for details please refer to Temperature Alarm in Design Guide.

2 FEATURES

- Core
  - ARM® Cortex®-M0 core runs up to 50MHz.
  - 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 2.4V to 5.5V.
  - Power management Unit (PMU) providing four levels of power control.
  - Deep Power Down (DPD) mode with sub micro-amp leakage (<1μA).
  - Wakeup from Deep Power Down via dedicated WAKEUP pin or timed operation from internal low power 16KHz oscillator.
  - Standby mode with limited RAM retention and RTC operation (<10μA).
  - 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
  - 145K bytes Flash EPROM for program code and data storage.
  - 4KB of flash can be configured as boot sector for ISP loader.
  - Support In-system program (ISP) and In-circuit program (ICP) application code update
  - 1K 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
  - One high speed and two low speed oscillators providing flexible selection for different applications. No external components necessary.
  - Built-in trimmable oscillator with range of 16-50MHz. Factory trimmed 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) 16KHz oscillator for watchdog and wakeup from power-down or sleep operation.
  - External 32kHz crystal input for RTC function and low power system operation.
- GPIO
  - Four I/O modes:
    - Quasi bi-direction
    - Push-Pull output
    - Open-Drain output
    - Input only with high impendence
  - TTL/Schmitt trigger input selectable.
  - I/O pin can be configured as interrupt source with edge/level setting.
  - Switchable pull-up.
- Audio Analog to Digital converter
  - Sigma Delta ADC with configurable decimation filter and 16 bit output.
  - 92dB 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.

- Differential Audio PWM Output (DPWM)
  - Direct connection of speaker
  - 1W drive capability into 8Ω load.
  - High efficiency 88%
  - Configurable up-sampling to support sample rates from 8-32kHz.
  - 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
  - Built-in up to two 16-bit PWM generators provide two PWM outputs or one 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
  - 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
  - Master up to 20 Mbps / Slave up to 10 Mbps.
  - Support MICROWIRE/SPI master/slave mode (SSP)
  - Full duplex synchronous serial data transfer
  - Variable length of transfer data from 1 to 32 bits
  - MSB or LSB first data transfer
  - 2 slave/device select lines when used in master mode.
  - Hardware CRC calculation module available for CRC calculation of data stream.
  - DMA support for burst transfers.

- 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.
  - I2C-bus controller supports multiple address recognition.

- I^2S
  - 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^2S 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

- Brown-out detector
  - With 8 levels: 2.1V, 2.2V, 2.4V, 2.5V, 2.625V, 2.8V, 3.0V, and 4.6V
  - 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 for output voltage of 1.8V, 2.4V, 3.0V and 3.3V
  - Eight GPIO (GPIOA<7:0>) operate from LDO voltage domain allowing direct interface to, for example, 3V SPI Flash.
  - Can be bypassed and voltage domain supplied directly from system power.

- Additional Features
  - Over temperature alarm. Can generate interrupt if device exceeds safe operating temperature.
  - Temperature proportional voltage source which can be routed to ADC for temperature measurements.
  - Digital Microphone interface.

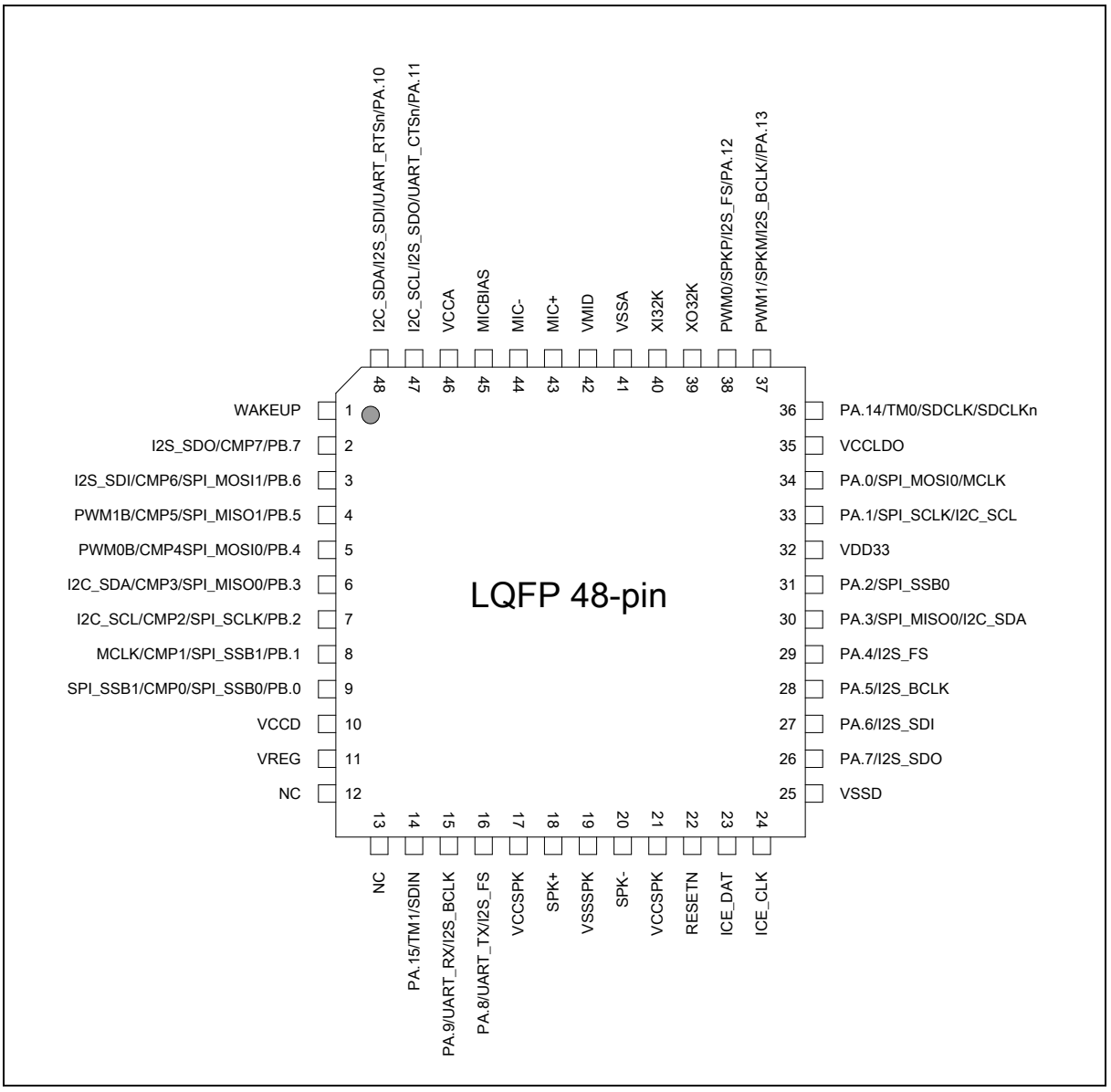
- Operating Temperature: -40C~85C

- Package:
  - LQFP 48-pin
  - QFN 33-pin
  - Package is Halogen-free, RoHS-compliant and TSCA-compliant

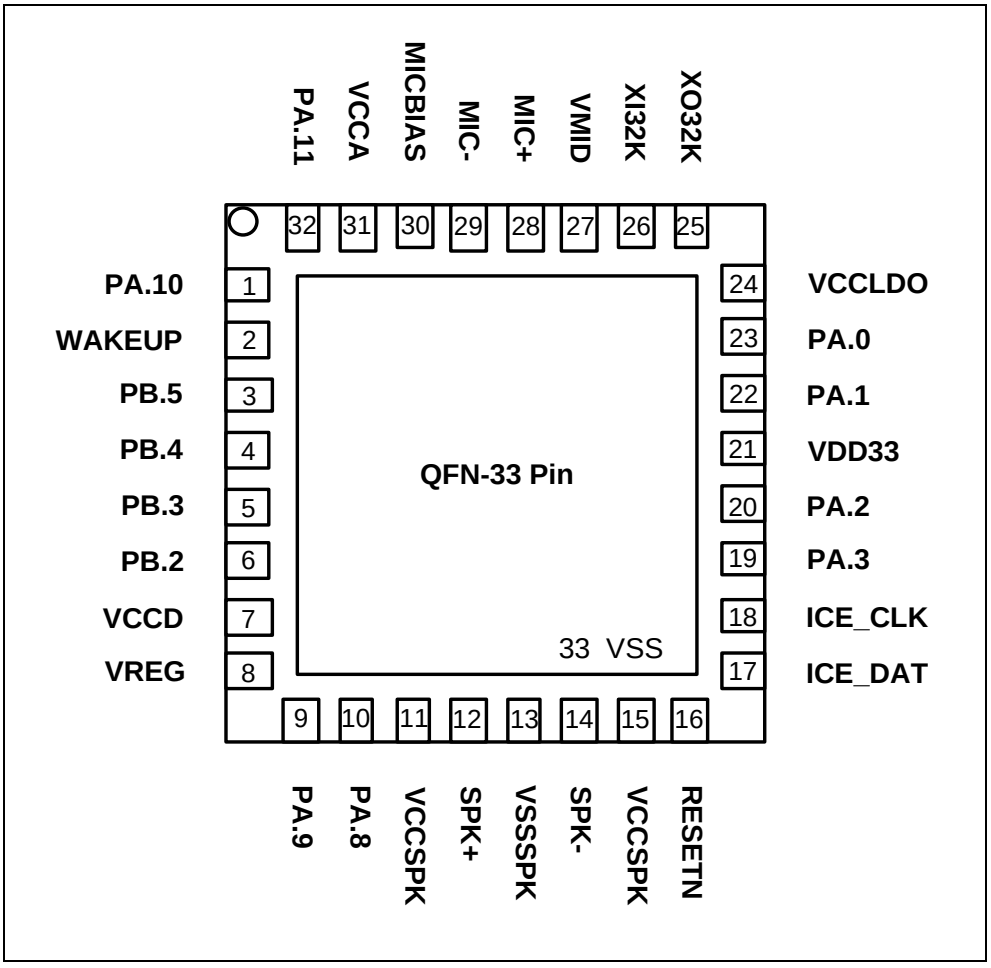
3 PART INFORMATION AND PIN CONFIGURATION

3.1 Pin Configuration

3.1.1 ISD9100 LQFP 48 pin



3.1.2 ISD9100 QFN 33 pin



3.1.3 Pin Description

The ISD9100 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.

Pin No.		Pin Name	Pin Type	Alt CFG	Description
LQFP 48	QFN 33				
1	2	WAKEUP	**I**		Pull low to wake part from deep power down
2	-	PB.7	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 7
		I2S_SDO	**O**	1	Serial Data Output for I2S interface
		CMP7	**AIO**	2	Configure as relaxation oscillator for capacitive touch sensing
3	-	PB.6	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 6
		I2S_SDI	**I**	1	Serial Data Input for I2S interface
		CMP6	**AIO**	2	Configure as relaxation oscillator for capacitive touch sensing
		SPI_MOSI1	**O**	3	Master Out, Slave In channel 1 for SPI interface
4	3	PB.5	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 5
		PWM1B	**O**	1	PWM channel 1 complementary output pin
		CMP5	**AIO**	2	Configure as relaxation oscillator for capacitive touch sensing
		SPI_MISO1	**I**	3	Master In, Slave Out channel 1 for SPI interface
5	4	PB.4	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 4
		PWM0B	**O**	1	PWM channel 0 complementary output pin
		CMP4	**AIO**	2	Configure as relaxation oscillator for capacitive touch sensing
		SPI_MOSI0	**O**	3	Master Out, Slave In channel 0 for SPI interface
6	5	PB.3	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 3
		I2C_SDA	**I/O**	1	Serial Data, I2C interface
		CMP3	**AIO**	2	Configure as relaxation oscillator for capacitive touch sensing
		SPI_MISO0	**I**	3	Master In, Slave Out channel 0 for SPI interface
7	6	PB.2	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 2
		I2C_SCL	**I/O**	1	Serial Clock, I2C interface
		CMP2	**AIO**	2	Configure as relaxation oscillator for capacitive touch sensing
		SPI_SCLK	**I/O**	3	Serial Clock for SPI interface
8	-	PB.1	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 1. Triggers external interrupt 1 (EINT1/IRQ3)
		MCLK	**O**	1	Master clock output for synchronizing external device
		CMP1	**AIO**	2	Configure as relaxation oscillator for capacitive touch sensing

Pin No.		Pin Name	Pin Type	Alt CFG	Description
LQFP 48	QFN 33				
		SPI_SSB1	**O**	3	Slave Select Bar 1 for SPI interface
9	-	PB.0	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 0. Triggers external interrupt 0 (EINT0/IRQ2)
		SPI_SSB1	**O**	1	Slave Select Bar 1 for SPI interface
		CMP0	**AIO**	2	Configure as relaxation oscillator for capacitive touch sensing
		SPI_SSB0	**I/O**	3	Slave Select Bar 0 for SPI interface
10	7	VCCD	**P**		Main Digital Supply for Chip. Supplies all IO except analog, Speaker Driver and PA<7:0>
11	8	VREG	**P**		Logic regulator output decoupling pin. A 1µF capacitor returning to VSSD must be placed on this pin.
12	-	NC			Should remain unconnected.
13	-	NC			Should remain unconnected.
14	-	PA.15	**I/O**	0	General purpose input/output pin; Port A, bit 15
		TM1	**I**	1	External input to Timer 1
		SDIN	**I**	2	Sigma Delta bit stream input for digital MIC mode
15	9	PA.9	**I/O**	0	General purpose input/output pin; Port A, bit 9
		UART_RX	**I**	1	Receive channel of UART
		I2S_BCLK	**I/O**	2	Bit Clock for I2S interface
16	10	PA.8	**I/O**	0	General purpose input/output pin; Port A, bit 8
		UART_TX	**O**	1	Transmit channel of UART
		I2S_FS	**I/O**	2	Frame Sync Clock for I2S interface
17	11	VCCSPK	**P**		Power Supply for PWM Speaker Driver
18	12	SPK+	**O**		Positive Speaker Driver Output
19	13	VSSSPK	**P**		Ground for PWM Speaker Driver
20	14	SPK-	**O**		Negative Speaker Driver Output
21	15	VCCSPK	**P**		Power Supply for PWM Speaker Driver
22	16	RESETN	**I**		External reset input. Pull this pin low to reset device to initial state. Has internal weak pull-up.
23	17	ICE_DAT	**I/O**		Serial Wire Debug port data pin. Has internal weak pull-up.
24	18	ICE_CLK	**I**		Serial Wire Debug port clock pin. Has internal weak pull-up.
25	-	VSSD	**P**		Digital Ground.
26	-	PA.7	**I/O**	0	General purpose input/output pin; Port A, bit 7
		I2S_SDO	**O**	1	Serial Data Out for I2S interface

Pin No.		Pin Name	Pin Type	Alt CFG	Description
LQFP 48	QFN 33				
27	-	PA.6	**I/O**	0	General purpose input/output pin; Port A, bit 6
		I2S_SDI	**I**	1	Serial Data In for I2S interface
28	-	PA.5	**I/O**	0	General purpose input/output pin; Port A, bit 5
		I2S_BCLK	**I/O**	1	Bit Clock for I2S interface
29	-	PA.4	**I/O**	0	General purpose input/output pin; Port A, bit 4
		I2S_FS	**I/O**	1	Frame Sync Clock for I2S interface
30	19	PA.3	**I/O**	0	General purpose input/output pin; Port A, bit 3
		SPI_MISO0	**I**	1	Master In, Slave Out channel 0 for SPI interface
		I2C_SDA	**I/O**	2	Serial Data, I2C interface
31	20	PA.2	**I/O**	0	General purpose input/output pin; Port A, bit 2
		SPI_SSB0	**I/O**	1	Slave Select Bar 0 for SPI interface
32	21	VDD33	**P**		LDO Regulator Output. If used, a 1µF capacitor must be placed to ground. If not used then tie to VCCD.
33	22	PA.1	**I/O**	0	General purpose input/output pin; Port A, bit 1
		SPI_SCLK	**I/O**	1	Serial Clock for SPI interface
		I2C_SCL	**I/O**	2	Serial Clock, I2C interface
34	23	PA.0	**I/O**	0	General purpose input/output pin; Port A, bit 2
		SPI_MOSI0	**O**	1	Master Out, Slave In channel 0 for SPI interface
		MCLK	**O**	2	Master clock output.
35	24	VCCLDO	**P**		Power Supply for LDO, should be connected to VCCD
36	-	PA.14	**I/O**	0	General purpose input/output pin; Port A, bit 14
		SDCLK	**O**	1	Clock output for digital microphone mode.
		SDCLKn	**O**	2	Inverse Clock output for digital microphone mode.
37	-	PA.13	**I/O**	0	General purpose input/output pin; Port A, bit 13
		PWM1	**O**	1	PWM1 Output.
		SPKM	**O**	2	Equivalent to SPK-.
		I2S_BCLK	**I/O**	3	Bit Clock for I2S interface
38	-	PA.12	**I/O**	0	General purpose input/output pin; Port A, bit 12
		PWM0	**O**	1	PWM0 Output.
		SPKP	**O**	2	Equivalent to SPK+
		I2S_FS	**I/O**	3	Frame Sync Clock for I2S interface
39	25	XO32K	**O**		32.768kHz Crystal Oscillator Output

Pin No.		Pin Name	Pin Type	Alt CFG	Description
LQFP 48	QFN 33				
40	26	XI32K	**I**		32.768kHz Crystal Oscillator Input. Max Voltage 1.8V
41		VSSA	**AP**		Ground for analog circuitry.
42	27	VMID	**O**		Mid rail reference. Connect 4.7µF to VSSA.
43	28	MIC+	**AI**		Positive microphone input.
44	29	MIC-	**AI**		Negative microphone input.
45	30	MICBIAS	**AO**		Microphone bias output.
46	31	VCCA	**AP**		Analog power supply.
47	32	PA.11	**I/O**	0	General purpose input/output pin; Port A, bit 11
		I2C_SCL	**I/O**	1	Serial Clock, I2C interface
		I2S_SDO	**O**	2	Serial Data Out I2S interface
		UART_CTSn	**I**	3	UART Clear to Send Input.
48	-	PA.10	**I/O**	0	General purpose input/output pin; Port A, bit 10
		I2C_SDA	**I/O**	1	Serial Data, I2C interface
		I2S_SDI	**I**	2	Serial Data In I2S interface
		UART_RTSn	**O**	3	UART Request to Send Output.
-	33	VSS	**P**		Ground for both digital and analog. Center pad underneath.

Note:

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

4 BLOCK DIAGRAM

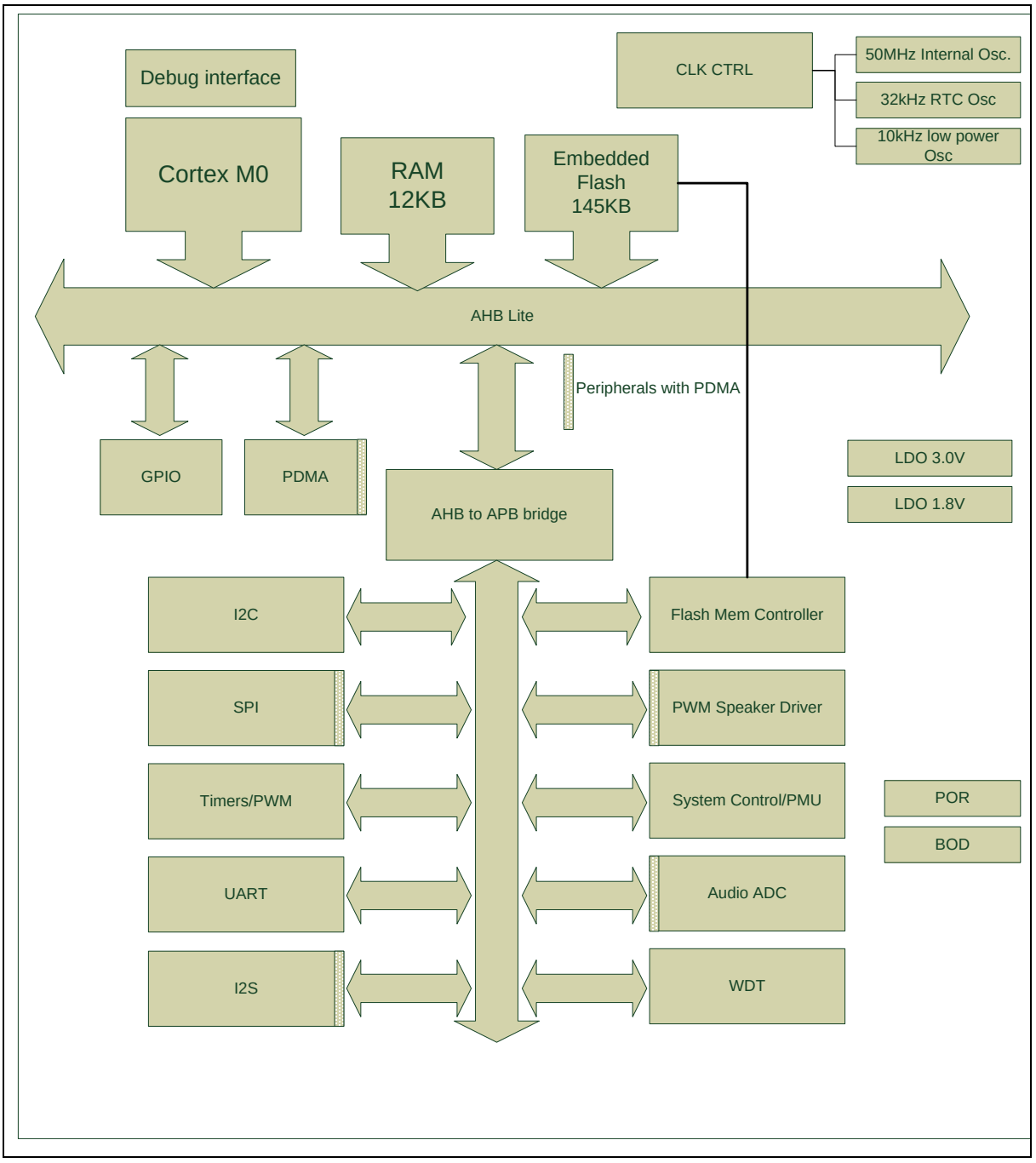


Figure 4-1 ISD9100 Block Diagram

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.

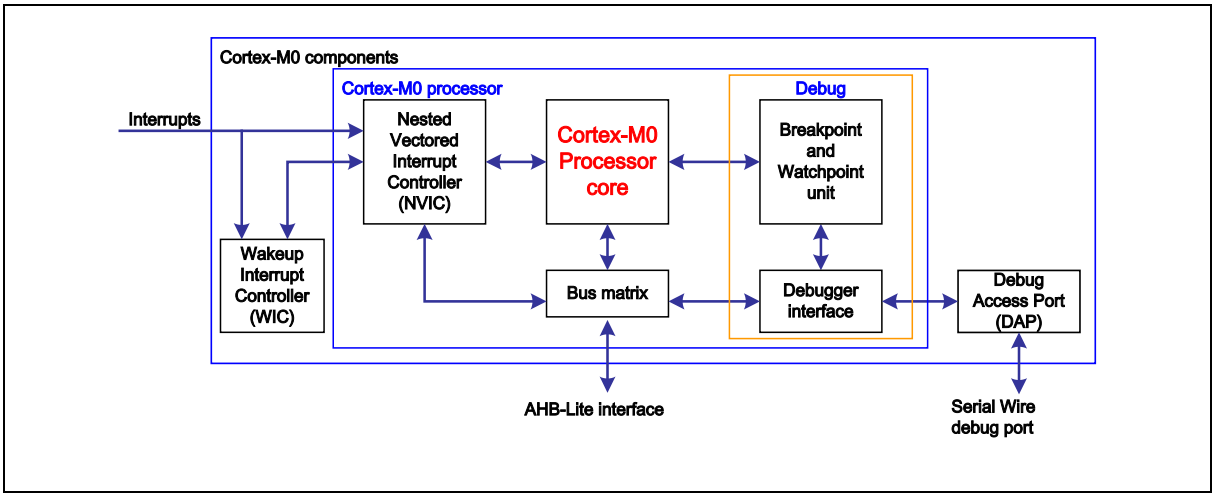


Figure 5-1 Functional Block Diagram

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 (1.8V) supply and will assert if the voltage on this rail drops below reliable logic threshold.

5.2.3 System Power Distribution

The ISD9100 series 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 1.8V power for digital operation.
- VCCLDO supplies the LDO regulator whose output is available on pin VDD33. This supply powers the IO ring for GPIOA<7:0>.
- An internal Standby reference (SB REG) generates a 1.8V 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 VDD33, require external decoupling capacitors which should be located close to the corresponding pin. The following diagram shows the power distribution of this device.

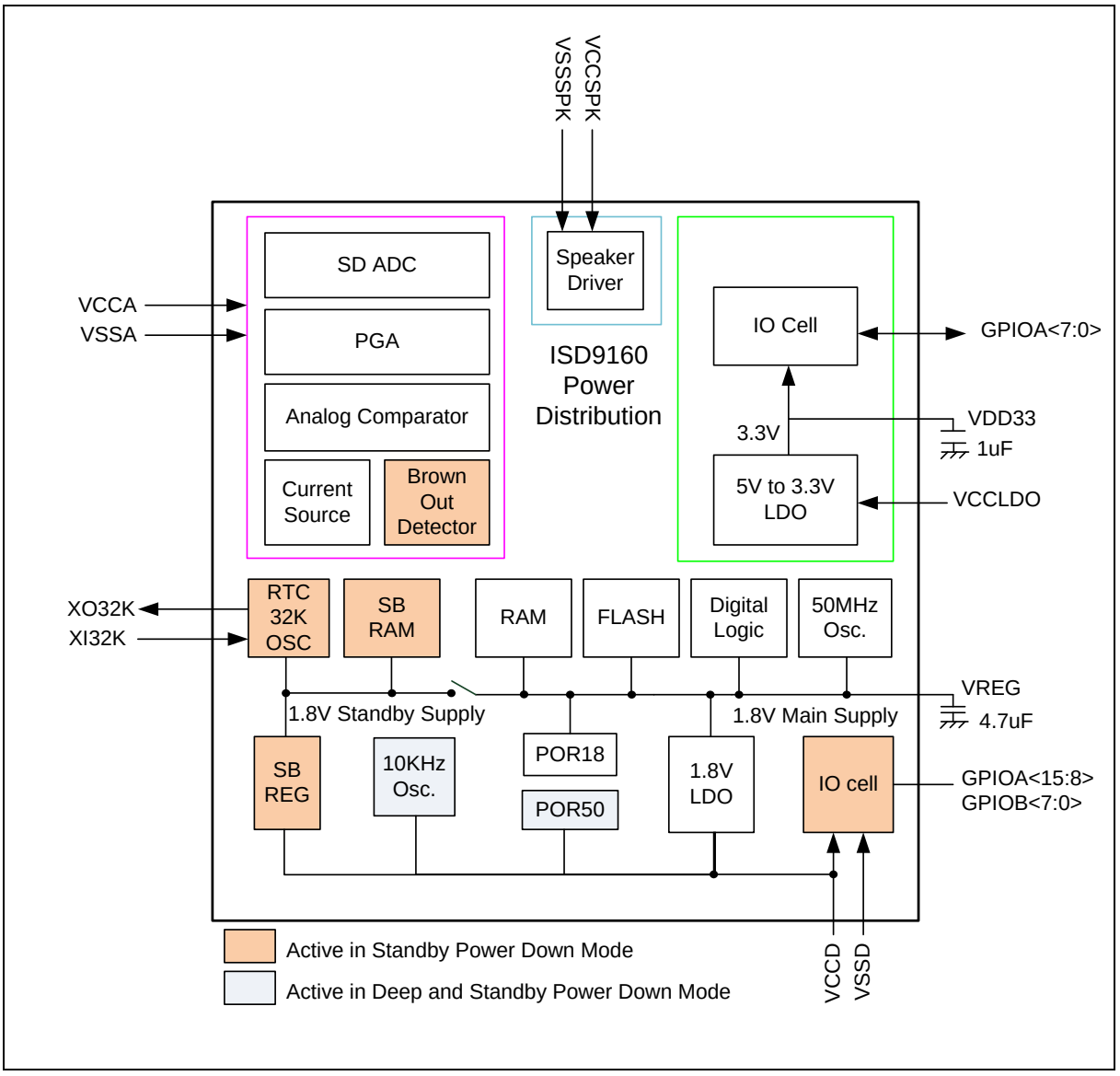


Figure 5-2 ISD9100 Power Distribution Diagram

5.2.4 System Memory Map

The ISD9100 series 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 ISD9100 series supports little-endian data format.

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

Address Space	Token	Modules	Reference
Flash & SRAM Memory Space			
0x0000_0000 – 0x0002_33FF	FLASH_BA	FLASH Memory Space (141KB)	
0x0000_0000 – 0x0002_43FF	FLASH_BA	FLASH Memory Space (145KB)	
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	5.2.5
0x5000_0200 – 0x5000_02FF	CLK_BA	Clock Control Registers	5.3.5
0x5000_0300 – 0x5000_03FF	INT_BA	Interrupt Multiplexer Control Registers	0
0x5000_4000 – 0x5000_7FFF	GPIO_BA	GPIO Control Registers	5.4.3
0x5000_8000 – 0x5000_BFFF	PDMA_BA	SRAM_APB DMA Control Registers	5.15
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash Memory Control Registers	6.3
APB1 Modules Space (0x4000_0000 ~ 0x400F_FFFF)			
0x4000_4000 – 0x4000_7FFF	WDT_BA	Watch-Dog Timer Control Registers	5.11
0x4000_8000 – 0x4000_BFFF	RTC_BA	Real Time Clock (RTC) Control Register	5.8
0x4001_0000 – 0x4001_3FFF	TIMER0_BA	Timer0/Timer1 Control Registers	5.10
0x4002_0000 – 0x4002_3FFF	I2C0_BA	I2C0 Interface Control Registers	5.6
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 Serial Interface Control Registers	5.9
0x4004_0000 – 0x4004_3FFF	PWM_BA	PWM0/1 Control Registers	5.7
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0 Control Registers	5.12
0x4007_0000 – 0x4007_3FFF	DPWM_BA	Differential Audio PWM Speaker Driver	7.2
0x4008_0000 – 0x4008_3FFF	ANA_ BA	Analog Block Control Registers	0
0x4008_4000 – 0x4008_7FFF	BODTALM_BA	Brown Out Detector Control Registers	5.5.1
0x4009_0000 – 0x4009_7FFF	CRC_BA	CRC Block Control Registers	5.14

0x400A_0000 - 0x400A_FFFF	I2S_BA	I2S Interface Control registers	5.13
0x400B_0000 - 0x400B_FFFF	BIQ_BA	Biquad Filter Control Registers	7.6
0x400D_0000 – 0x400D_3FFF	ACMP_BA	Analog Comparator Control Registers	0
0x400E_0000 – 0x400E_FFFF	ADC0_BA	Analog-Digital-Converter (ADC) Registers	7.1
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	5.2.6
0xE000_E100 – 0xE000_ECFF	SCS_BA	External Interrupt Controller Control Registers	5.2.7
0xE000_ED00 – 0xE000_ED8F	SYSINFO_BA	System Control Registers	5.2.8

5.2.5 System Manager Control Registers

Register	Offset	R/W	Description	Reset Value
SYS Base Address: SYS_BA = 0x5000_0000				
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Source Register	0x0000_00XX
SYS_IPRST0	SYS_BA+0x08	R/W	IP Reset Control Resister1	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	IP Reset Control Resister2	0x0000_0000
SYS_PASMTEN	SYS_BA+0x30	R/W	GPIOA input type control register	0x0000_0000
SYS_PBSMTEN	SYS_BA+0x34	R/W	GPIOB input type control register	0x0000_0000
SYS_GPA_MFP	SYS_BA+0x38	R/W	GPIOA multiple function control register	0x0000_0000
SYS_GPB_MFP	SYS_BA+0x3C	R/W	GPIOB multiple function 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 Key Address register	0x0000_0000
SYS_IRCTCTL	SYS_BA+0x110	R/W	Oscillator Frequency Adjustment control register	0xXXXX_XXXX

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_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**
CPURF	**PMURSTF**	**SYSRF**	**Reserved**	**Reserved**	**WDTRF**	**Reserved**	**CORERSTF**

Table 5-2 System Reset Source Register (SYS_RSTSTS, address 0x5000_0004) Bit Description.

Bits	Description	
[31:8]	Reserved	Reserved
[7]	**CPURF**	**Reset Source From CPU** The CPURF flag is set by hardware if software writes SYS_IPRST0.CPURST 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 SYSCTL_AIRCTL.SYSRESTREQ, Application Interrupt and Reset Control Register) in system control registers of Cortex_M0 kernel. This bit is cleared by writing 1 to itself.
[4:3]	Reserved	Reserved
[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]	Reserved	Reserved
[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 Resister1	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

Table 5-3 IP Reset Control Register 1 (SYS_IPRST0 address 0x5000_0008) Bit Description.

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 (see Protected Register Lock Key Register (SYS_REGLCTL)) 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 (see Protected Register Lock Key Register (SYS_REGLCTL)) 0= Normal 1= Reset CHIP

IP Reset Control Register2 (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 Resister2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	ANARST	I2S0RST	EADCRST	Reserved	Reserved	Reserved	Reserved
23	22	21	20	19	18	17	16
Reserved	ACMPRST	Reserved	PWM0RST	CRCRST	BIQRST	Reserved	UART0RST
15	14	13	12	11	10	9	8
Reserved	Reserved	DPWMRST	SPI0RST	Reserved	Reserved	Reserved	I2C0RST
7	6	5	4	3	2	1	0
TMR1RST	TMR0RST	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

Table 5-4 IP Reset Control Register 2 (SYS_IPRST1 address 0x5000_000C) Bit Description.

Bits	Description	
[30]	ANARST	**Analog Block Control Reset** 0 = Normal Operation 1 = Reset
[29]	I2S0RST	**I2S Controller Reset** 0 = Normal Operation 1 = Reset
[28]	EADCRST	**ADC Controller Reset** 0 = Normal Operation 1 = Reset
[22]	ACMPRST	**Analog Comparator Reset** 0 = Normal Operation 1 = Reset
[20]	PWM0RST	**PWM10 controller Reset** 0 = Normal Operation 1 = Reset
[19]	CRCRST	**CRC Generation Block Reset** 0 = Normal Operation 1 = Reset

[18]	**BIQRST**	**Biquad Filter Block Reset** 0 = Normal Operation 1 = Reset
[16]	**UART0RST**	**UART0 Controller Reset** 0 = Normal Operation 1 = Reset
[13]	**DPWMRST**	**DPWM Speaker Driver Reset** 0 = Normal Operation 1 = Reset
[12]	**SPI0RST**	**SPI0 Controller Reset** 0 = Normal Operation 1 = Reset
[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

GPIOA Input Type Control Register (SYS_PASMTEN)

Register	Offset	R/W	Description	Reset Value
SYS_PASMTEN	SYS_BA+0x30	R/W	GPIOA input type control register	0x0000_0000

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

Table 5-5 GPIOA Input Type Control Register (SYS_PASMTEN address 0x5000_0030) Bit Description.

Bits	Description	
[n] **n=16,17..31**	**SMTEN**	**Schmitt Trigger** This register controls whether the GPIO input buffer Schmitt trigger is enabled. 0 = GPIOA[15:0] I/O input Schmitt Trigger disabled 1 = GPIOA[15:0] I/O input Schmitt Trigger enabled
[15:0]	**Reserved**	**Reserved**

GPIOB Input Type Control Register (SYS_PBSMTEN)

Register	Offset	R/W	Description	Reset Value
SYS_PBSMTEN	SYS_BA+0x34	R/W	GPIOB input type control register	0x0000_0000

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

Table 5-6 GPIOB Input Type Control Register (SYS_PBSMTEN address 0x5000_0034) Bit Description.

Bits	Description	
[n] **n=16,17..23**	**SMTEN**	**Schmitt Trigger** This register controls whether the GPIO input buffer Schmitt trigger is enabled. 0= GPIOB(port 0 ~ port 7) I/O input Schmitt Trigger disabled 1= GPIOB(port 0 ~ port 7) I/O input Schmitt Trigger enabled

GPIO 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 function 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	

Table 5-7 GPIOA Alternate Function Register (SYS_GPA_MFP address 0x5000_0038)

Bits	Description	
[31:30]	**PA15MFP**	**Alternate Function Setting For PA15MFP** 00 = GPIO 01 = TM1 10 = SDIN
[29:28]	**PA14MFP**	**Alternate Function Setting For PA14MFP** 00 = GPIO 01 = TM0 10 = SDCLK 11 = SDCLKn
[27:26]	**PA13MFP**	**Alternate Function Setting For PA13MFP** 00 = GPIO 01 = PWM1 10 = SPKM 11 = I2S_BCLK

[25:24]	**PA12MFP**	**Alternate Function Setting For PA12MFP** 00 = GPIO 01 = PWM0 10 = SPKP 11 = I2S_FS
[23:22]	**PA11MFP**	**Alternate Function Setting For PA11MFP** 00 = GPIO 01 = I2C_SCL 10 = I2S_SDO 11 = UART_CTSn
[21:20]	**PA10MFP**	**Alternate Function Setting For PA10MFP** 00 = GPIO 01 = I2C_SDA 10 = I2S_SDI 11 = UART_RTSn
[19:18]	**PA9MFP**	**Alternate Function Setting For PA0MFP** 00 = GPIO 01 = UART_RX 10 = I2S_BCLK
[17:16]	**PA8MFP**	**Alternate Function Setting For PA8MFP** 00 = GPIO 01 = UART_TX 10 = I2S_FS
[15:14]	**PA7MFP**	**Alternate Function Setting For PA7MFP** 00 = GPIO 01 = I2S_SDO
[13:12]	**PA6MFP**	**Alternate Function Setting For PA6MFP** 00 = GPIO 01 = I2S_SDI
[11:10]	**PA5MFP**	**Alternate Function Setting For PA5MFP** 00 = GPIO 01 = I2S_BCLK
[9:8]	**PA4MFP**	**Alternate Function Setting For PA4MFP** 00 = GPIO 01 = I2S_FS

[7:6]	**PA3MFP**	**Alternate Function Setting For PA3MFP** 00 = GPIO 01 = SPI_MISO0 10 = I2C_SDA
[5:4]	**PA2MFP**	**Alternate Function Setting For PA2MFP** 00 = GPIO 01 = SPI_SSB0
[3:2]	**PA1MFP**	**Alternate Function Setting For PA1MFP** 00 = GPIO 01 = SPI_SCLK 10 = I2C_SCL
[1:0]	**PA0MFP**	**Alternate Function Setting For PA0MFP** 00 = GPIO 01 = SPI_MOSI0 10 = MCLK

GPIO 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 function 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
PB7MFP		PB6MFP		PB5MFP		PB4MFP	
7	6	5	4	3	2	1	0
PB3MFP		PB2MFP		PB1MFP		PB0MFP	

Table 5-8 GPIOB Alternate Function Register (SYS_GPB_MFP address 0x5000_003C)

Bits	Description	
[31:16]	Reserved	Reserved
[15:14]	PB7MFP	**Alternate Function Setting For PB7MFP** 00 = GPIO 01 = I2S_SDO 10 = CMP7
[13:12]	PB6MFP	**Alternate Function Setting For PB6MFP** 00 = GPIO 01 = I2S_SDI 10 = CMP6 11 = SPI_MOSI1
[11:10]	PB5MFP	**Alternate Function Setting For PB5MFP** 00 = GPIO 01 = PWM1B 10 = CMP5 11 = SPI_MISO1

[9:8]	**PB4MFP**	**Alternate Function Setting For PB4MFP** 00 = GPIO 01 = PWM0B 10 = CMP4 11 = SPI_MOSI0
[7:6]	**PB3MFP**	**Alternate Function Setting For PB3MFP** 00 = GPIO 01 = I2C_SDA 10 = CMP3 11 = SPI_MISO0
[5:4]	**PB2MFP**	**Alternate Function Setting For PB2MFP** 00 = GPIO 01 = I2C_SCL 10 = CMP2 11 = SPI_SCLK
[3:2]	**PB1MFP**	**Alternate Function Setting For PB1MFP** 00 = GPIO 01 = MCLK 10 = CMP1 11 = SPI_SSB1
[1:0]	**PB0MFP**	**Alternate Function Setting For PB0MFP** 00 = GPIO 01 = SPI_SSB1 10 = CMP0 11 = SPI_SSB0

GPIO	**Power Domain**	**GPAn=01**		**GPAn =10**		**GPAn =11**	
		Function	**Type**	**Function**	**Type**	**Function**	**Type**
GPIOA0	VDD33	SPI_MOSI0	O	MCLK	O		
GPIOA1	VDD33	SPI_SCLK	IO	I2C_SCL	IO		
GPIOA2	VDD33	SPI_SSB0	IO				
GPIOA3	VDD33	SPI_MISO0	I	I2C_SDA	IO		
GPIOA4	VDD33	I2S_FS	IO				
GPIOA5	VDD33	I2S_BCLK	IO				
GPIOA6	VDD33	I2S_SDI	I				

GPIOA7	VDD33	I2S_SDO	O				
GPIOA8	VCCD	UART_TX	O	I2S_FS	IO		
GPIOA9	VCCD	UART_RX	I	I2S_BCLK	IO		
GPIOA10	VCCD	I2C_SDA	IO	I2S_SDI	I	UART_RTSn	O
GPIOA11	VCCD	I2C_SCL	IO	I2S_SDO	O	UART_CTSn	I
GPIOA12	VCCD	PWM0	O	SPKP	O	I2S_FS	IO
GPIOA13	VCCD	PWM1	O	SPKM	O	I2S_BCLK	IO
GPIOA14	VCCD	TM0	I	SDCLK	O	SDCLKn	O
GPIOA15	VCCD	TM1	I	SDIN	I		

GPIO	**Power Domain**	**GPBn=01**		**GPBn =10**		**GPBn =11**	
		Function	**Type**	**Function**	**Type**	**Function**	**Type**
GPIOB0	VCCD	SPI_SSB1	O	CMP0	AIO	SPI_SSB0	IO
GPIOB1	VCCD	MCLK	O	CMP1	AIO	SPI_SSB1	O
GPIOB2	VCCD	I2C_SCL	IO	CMP2	AIO	SPI_SCLK	IO
GPIOB3	VCCD	I2C_SDA	IO	CMP3	AIO	SPI_MISO0	I
GPIOB4	VCCD	PWM0B	O	CMP4	AIO	SPI_MOSI0	O
GPIOB5	VCCD	PWM1B	O	CMP5	AIO	SPI_MISO1	I
GPIOB6	VCCD	I2S_SDI	I	CMP6	AIO	SPI_MOSI1	O
GPIOB7	VCCD	I2S_SDO	O	CMP7	AIO		

Wakeup Pin Control Register (SYS_WKCTL)

The WAKEUP pin of the ISD9100 series is a special purpose pin that can be used to wake the device from a deep power down condition when all other pins of the device are inactive. When the device is active, this register can be used to set the state of the WAKEUP pin. The default state of the pin is as a tri-state input.

Register	Offset	R/W	Description	Reset Value
SYS_WKCTL	SYS_BA+0x54	R/W	WAKEUP pin control register	0x0000_0006

7	6	5	4	3	2	1	0
Reserved				**WKDIN**	**WKPUEN**	**WKOENB**	**WKDOUT**

Table 5-9 Wakeup Pin Control Register (SYS_WKCTL, address 0x5000_0054) Bit Description.

Bits	Description	
[3]	**WKDOUT**	**Wakeup Output State** 0 = drive Low if the corresponding output mode bit is set (default) 1 = drive High if the corresponding output mode bit is set
[2]	**WKOENB**	**Wakeup Pin Output Enable Bar** 0 = drive WKDOUT to pin 1 = tristate (default)
[1]	**WKPUEN**	**Wakeup Pin Pull-up Control** This signal is latched in deep power down and preserved. 0 = pull-up enable 1 = tristate (default)
[0]	**WKDIN**	**State Of Wakeup Pin** Read only.

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 Key Address register	0x0000_0000

7	6	5	4	3	2	1	0
							REGLCTL

Table 5-10 Protected Register Lock Key Register (SYS_REGLCTL address 0x5000_0100) Bit Description.

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 ISD9100 series has an adjustable frequency and is controlled by the SYS_IRCTCTL register. This register contains two oscillator frequency trim values, which one is active depends upon the setting of register CLK_CLKSEL0.HIRCFSEL bit. If this bit is 0, SYS_IRCTCTL[0] is active, if 1 then SYS_IRCTCTL[1] is active. Upon power on reset this register is loaded from flash memory with factory stored values to give oscillator frequencies of 49.152MHz for SYS_IRCTCTL[0] and 32.768MHz for SYS_IRCTCTL[1]. If users wish to change either of the default frequencies it is possible to do so by setting this register. An additional SUPERFINE trim register is also available to interpolate frequencies between the available SYS_IRCTCTL settings (see Table 7-37)

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

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							**RGE1SEL**
23	22	21	20	19	18	17	16
FREQ1SEL							
15	14	13	12	11	10	9	8
Reserved							**RGE0SEL**
7	6	5	4	3	2	1	0
FREQ0SEL							

Table 5-11 Oscillator Frequency Adjustment Control Register (SYS_IRCTCTL, address 0x5000_0110).

Bits	Description	
[24]	**RGE1SEL**	**Range Bit For Oscillator** 0 = high range 1 = low range
[23:16]	**FREQ1SEL**	**8 Bit Trim For Oscillator** FREQ1SEL [7:5] are 8 coarse trim ranges which overlap in frequency. FREQ1SEL [4:0] are 32 fine trim steps of approximately 0.5% resolution.
[8]	**RGE0SEL**	**Range Bit For Oscillator** 0 = high range 1 = low range
[7:0]	**FREQ0SEL**	**8 Bit Trim For Oscillator** FREQ0SEL [7:5] are 8 coarse trim ranges which overlap in frequency. FREQ0SEL [4:0] are 32 fine trim steps of approximately 0.5% resolution.

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_0004
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

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_0004

Table 5-12 SysTick Control and Status Register (SYST_CSR, address 0xE000_E010)

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

Table 5-13 SysTick Reload Value Register (SYST_RVR, address 0xE000_E014)

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

Bits	Description	
[31:24]	Reserved	Reserved
[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

Table 5-14 SysTick Current Value Register (SYST_CVR, address 0xE000_E018)

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

Bits	Description	
[31:24]	Reserved	Reserved
[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 ISD9100 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 5-15 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 5-16 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	**PWM_IRQn**	PWM01	PWM0, PWM1 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	Reserved		
28	12	**UART0_IRQn**	UART0	UART0 interrupt
29	13	Reserved		
30	14	**SPI0_IRQn**	SPI0	SPI0 interrupt
31	15	Reserved		
32	16	Reserved		
33	17	Reserved		
34	18	**I2C0_IRQn**	I2C0	I2C0 interrupt
35	19	Reserved		
36	20	Reserved		
37	21	**TALARM_IRQn**	TALARM	Temperature Alarm Interrupt
38	22	Reserved		
39	23	Reserved		
40	24	Reserved		
41	25	**ACMP_IRQn**	ACMP	Analog Comparator-0 or Comaprator-1 interrupt
42	26	**PDMA_IRQn**	PDMA	PDMA interrupt
43	27	**I2S_IRQn**	I2S	I2S 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 5-17 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_E000				
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-Enable Control Register	0x0000_0000
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 Priority Control Register	0x0000_0000
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 Priority Control Register	0x0000_0000
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11 Priority Control Register	0x0000_0000
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15 Priority Control Register	0x0000_0000
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19 Priority Control Register	0x0000_0000
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 Priority Control Register	0x0000_0000
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ27 Priority Control Register	0x0000_0000
NVIC_IPR7	SCS_BA+0x41C	R/W	IRQ28 ~ IRQ31 Priority Control Register	0x0000_0000

IRQ0 ~ IRQ31 Set-Enable Control Register （NVIC_ISER）

Register	Offset	R/W	Description	Reset Value
NVIC_ISER	SCS_BA+0x100	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.

Table 5-18 Interrupt Set-Enable Control Register (ISER, address 0xE000_E100) Bit Description

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+0x180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000

Table 5-19 Interrupt Clear-Enable Control Register (ICER, address 0xE000_E180) Bit Description

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+0x200	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000

Table 5-20 Interrupt Set-Pending Control Register (ISPR, address 0xE000_E200)

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+0x280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000

Table 5-21 Interrupt Clear-Pending Control Register (ICPR, address 0xE000_E280)

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+0x400	R/W	IRQ0 ~ IRQ3 Priority Control Register	0x0000_0000

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

Table 5-22 Interrupt Priority Register (IPR0, address 0xE000_E400)

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

IRQ4 ~ IRQ7 Interrupt Priority Register（NVIC_IPR1）

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

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

Table 5-23 Interrupt Priority Register (IPR1, address 0xE000_E404)

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

IRQ8 ~ IRQ11 Interrupt Priority Register （NVIC_IPR2）

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

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

Table 5-24 Interrupt Priority Register (IPR2, address 0xE000_E408)

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

IRQ12 ~ IRQ15 Interrupt Priority Register （NVIC_IPR3）

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

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

Table 5-25 Interrupt Priority Register (IPR3, address 0xE000_E40C)

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

IRQ16 ~ IRQ19 Interrupt Priority Register（NVIC_IPR4）

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

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

Table 5-26 Interrupt Priority Register (IPR4, address 0xE000_E410)

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

IRQ20 ~ IRQ23 Interrupt Priority Register（NVIC_IPR5）

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

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

Table 5-27 Interrupt Priority Register (IPR5, address 0xE000_E414)

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+0x418	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					

Table 5-28 Interrupt Priority Register (IPR6, address 0xE000_E418)

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+0x41C	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					

Table 5-29 Interrupt Priority Register (IPR7, address 0xE000_E41C)

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 ISD9100 series 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 (PWMA) 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 (Reserved) 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 (Reserved) 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 (Reserved) 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 (TALARM) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (Reserved) 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 (ACMP) 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 (I2S) 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

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[2:0]**		

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[2:0]**		

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[2:0]		

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[2:0]		

Bits	Description	
[2:0]	INT_SRC	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: INT0_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[2:0]**		

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[2:0]		

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

IRQ6(PWMA) Interrupt Source Identify Register (IRQ6_SRC)

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

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

Bits	Description	
[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[2:0]**		

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[2:0]**		

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

IRQ12(UART0) Interrupt Source Identify Register (IRQ8_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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: UART0_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[2:0]		

Bits	Description	
[2:0]	INT_SRC	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: SPI0_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[2:0]		

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

IRQ21(TALARM) Interrupt Source Identify Register (IRQ21_SRC)

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

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

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

IRQ25(TALARM) Interrupt Source Identify Register (IRQ25_SRC)

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

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

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

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

IRQ27(I2S) Interrupt Source Identify Register (IRQ27_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ27_SRC	INT_BA+0x6C	R	IRQ27 (I2S) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[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[2:0]		

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[2:0]**		

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[2:0]**		

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[4:0]**				

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 (see Protected Register Lock Key Register (SYS_REGLCTL))
[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
MCU_IRQ[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MCU_IRQ[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MCU_IRQ[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MCU_IRQ[7:0]							

Bits	Description	
[31:0]	**MCU_IRQ**	**MCU IRQ Source Test Mode** In Normal mode (NMI_SEL register bit [7] aaa 0) The device collects interrupts from each peripheral and synchronizes them to interrupt the Cortex-M0. In Test mode (NMI_SEL register bit [7] aaa 1), the interrupts from peripherals are blocked, and the interrupts are replaces by MCU_IRQ[31:0]. When MCU_IRQ[n] is “0” : Writing MCU_IRQ[n] “1” will generate an interrupt to Cortex_M0 NVIC[n]. When MCU_IRQ[n] is “1” (meaning an interrupt is asserted) writing MCU_bit[n] ‘1’ will clear the interrupt Writing MCU_IRQ[n] “0” : has no effect.

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_E000				
SYSCTL_CPUID	SYSINFO_BA+0xD00	R	CPUID Base Register	0x410C_C200
SYSCTL_ICSR	SYSINFO_BA+0xD04	R/W	Interrupt Control State Register	0x0000_0000
SYSCTL_AIRCTL	SYSINFO_BA+0xD0C	R/W	Application Interrupt and Reset Control Register	0x0000_0000
SYSCTL_SCR	SYSINFO_BA+0xD10	R/W	System Control Register	0x0000_0000
SYSCTL_SHPR2	SYSINFO_BA+0xD1C	R/W	System Handler Priority Register 2	0x0000_0000
SYSCTL_SHPR3	SYSINFO_BA+0xD20	R/W	System Handler Priority Register 3	0x0000_0000

CPUID Base Register (SYSCTL_CPUID)

Register	Offset	R/W	Description	Reset Value
SYSCTL_CPUID	SYSINFO_BA+0xD00	R	CPUID Base Register	0x410C_C200

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

Bits	Description	
[31:24]	IMPCODE	**Implementer Code Assigned By ARM** ARM aaa 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 (SYSCTL_ICSR)

Register	Offset	R/W	Description	Reset Value
SYSCTL_ICSR	SYSINFO_BA+0xD04	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	VTPEND[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).
[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 SysTick** Reads back with current state (1 if Pending, 0 if not).
[25]	PSTKICLR	**Clear A pending SysTick** Write 1 to clear a pending SysTick.
[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.
[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.
[8:0]	VTACT	**Vector Active** 0: Thread mode Value > 1: the exception number for the current executing exception.

Application Interrupt and Reset Control Register (SYSCTL_AIRCTL)

Register	Offset	R/W	Description	Reset Value
SYSCTL_AIRCTL	SYSINFO_BA+0xD0C	R/W	**Application Interrupt and Reset Control Register**	0x0000_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	**Endianess** Read Only. Reads 0 indicating little endian machine.
[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.

System Control Register (SYSCTL_SCR)

Register	Offset	R/W	Description	Reset Value
SYSCTL_SCR	SYSINFO_BA+0xD10	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	
[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	**Sleep Deep Control** 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.

System Handler Priority Register 2 (SYSCTL_SHPR2)

Register	Offset	R/W	Description	Reset Value
SYSCTL_SHPR2	SYSINFO_BA+0xD1C	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

System Handler Priority Register 3 (SYSCTL_SHPR3)

Register	Offset	R/W	Description	Reset Value
SYSCTL_SHPR3	SYSINFO_BA+0xD20	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 – SysTick** "0" denotes the highest priority and "3" denotes lowest priority
[23:22]	PRI14	**Priority Of System Handler 14 – PendSV** "0" denotes the highest priority and "3" denotes lowest priority

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 3 sources listed below:

- An internal programmable high frequency oscillator factory trimmed to provide frequencies of 49.152MHz and 32.768MHz to 1% accuracy.
- An external 32kHz crystal
- An internal low power 16 kHz oscillator.

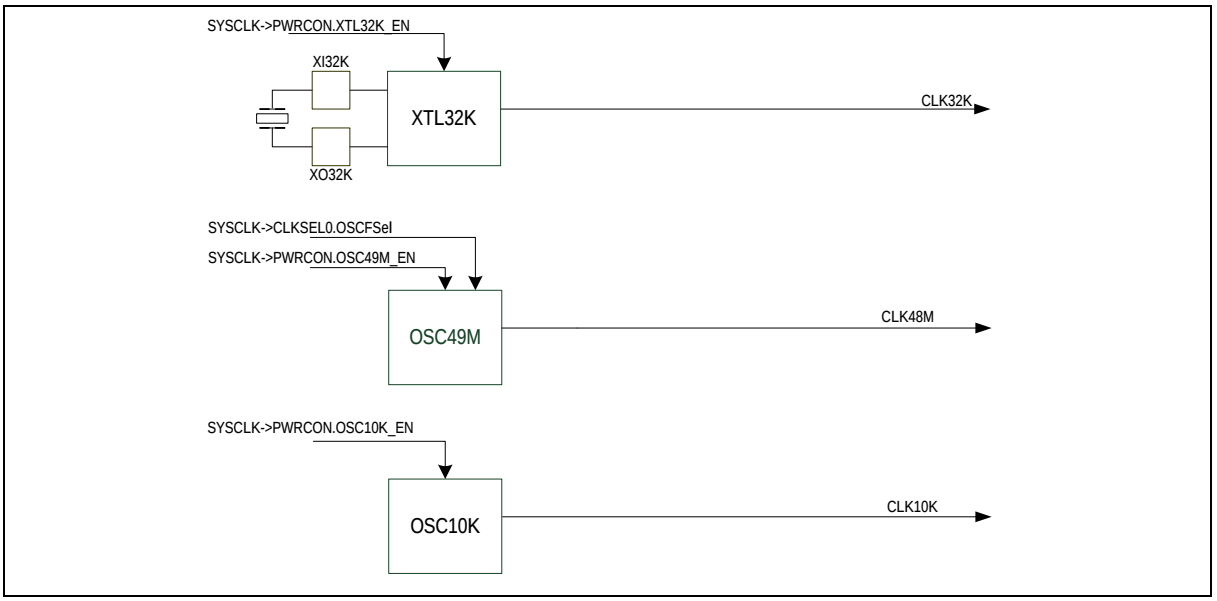


Figure 5-3 Clock generator block diagram

5.3.2 System Clock & SysTick Clock

The system clock has 3 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.

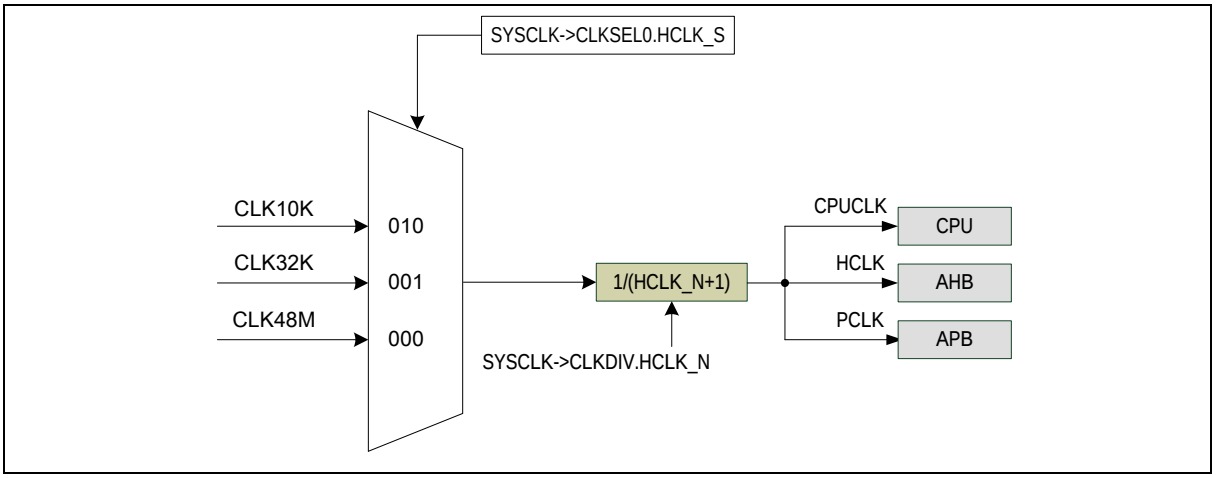


Figure 5-4 System Clock Block Diagram

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

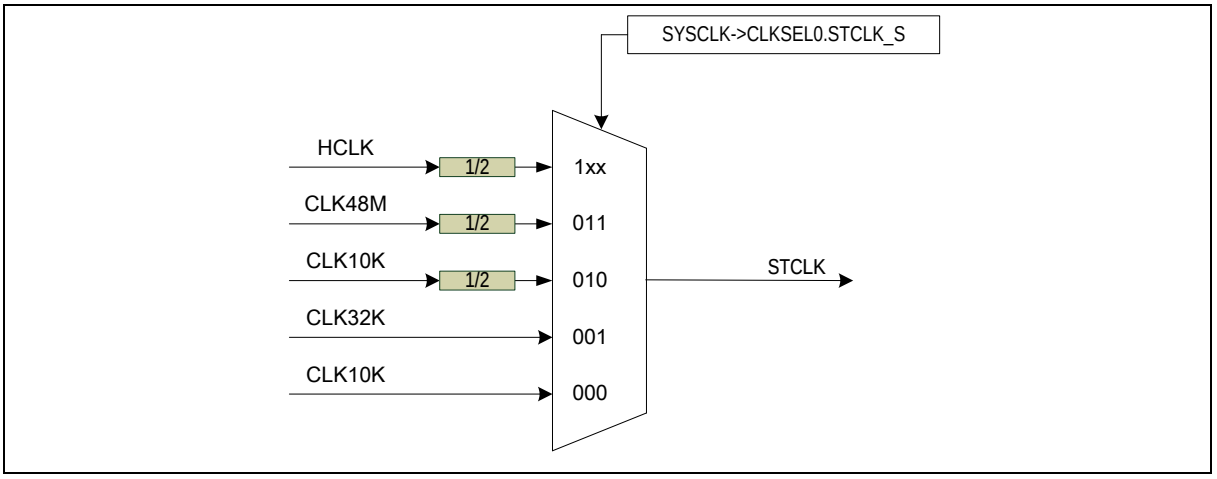


Figure 5-5 SysTick Clock Control Block Diagram

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_SLEEP 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 ISD9100 series is equipped with a Power Management Unit (PMU) that implements a variety of power saving modes. There are four levels of power control with increasing functionality (and power consumption):

- Level0 : Deep Power Down (DPD)
- Level1 : Standby Power Down (SPD)
- Level2 : Deep Sleep
- Level3 : Sleep
- Level4 : 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 16KHz oscillator is configured active and reaches a certain count.
3. A power cycle of main chip supply triggering a POR event.

To assist software in determining previous state of device before a DPD, a one-byte register is available PD_STATE[7:0] that can be loaded with a value to be preserved before issuing a DPD request.

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

- 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 16KHz oscillator will power down in DPD. No timed wakeup is possible.
- CLK_PWRCTL.SELWKTMR: Each bit in this register will trigger a wakeup event after a certain number of OSC16K clock 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 ICECLKST and ICEDATST 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 (exclude GPB0 & GPB1), 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.PD 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: 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 CLK48M oscillator is shutdown and a low speed oscillator is selected, if CLK32K is active this source is selected, if not then CLK16K 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_SLEEP register, note however that HCLK is operating at a low frequency and CLK48M is not available. Deep Sleep mode is entered by setting System Control register bit 2: SCB->SCR |= (1UL << 2) and executing a WFI/WFE instruction. Software can determine whether the device woke up from Deep Sleep by interrogating the register bit CLK_PWRSTSF.DSF.

5.3.4.4 *Level3: 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_SLEEP 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 Clock Control Register Map

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

Register	Offset	R/W	Description	Reset Value
CLK Base Address: CLK_BA = 0x5000_0200				
CLK_PWRCTL	CLK_BA + 0x00	R/W	System Power Control Register	0xXX00_0006
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_0000
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	0x3300_771F
CLK_CLKDIV0	CLK_BA + 0x18	R/W	Clock Divider Number Register	0x0000_0000
CLK_CLKSEL2	CLK_BA + 0x1C	R/W	Clock Source Select Control Register 2	0xFFFF_FFFX
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

5.3.6 Clock Control Register Description

System Power Control Register （CLK_PWRCTL）

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

Register	Offset	R/W	Description	Reset Value
CLK_PWRCTL	CLK_BA + 0x00	R/W	System Power Control Register	0xXX00_0006

Table 5-30 System Power Control Register （CLK_PWRCTL, address 0x5000_0200）

31	30	29	28	27	26	25	24
WKTMRSTS				Reserved	PORWKF	TMRWKF	WKPINWKF
23	22	21	20	19	18	17	16
SELWKTMR				Reserved		LIRCDPDEN	WKPINEN
15	14	13	12	11	10	9	8
				DPDEN	SPDEN	STOP	Reserved
7	6	5	4	3	2	1	0
Reserved				LIRCEN	HIRCEN	LXTEN	Reserved

Table 5-31 System Power Control Register (CLK_PWRCTL, address 0x5000_0200) Bit Description.

Bits	Description	
[31:28]	**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.
[27]	**Reserved**	Reserved
[26]	**PORWKF**	**POI 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 16Khz 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]	**SELWKTMR**	**Select Wakeup Timer** SELWKTMR[0] aaa 1: WAKEUP after 128 OSC16K clocks (12.8 ms) SELWKTMR[1] aaa 1: WAKEUP after 256 OSC16K clocks (25.6 ms) SELWKTMR[2] aaa 1: WAKEUP after 512 OSC16K clocks (51.2 ms) SELWKTMR[3] aaa 1: WAKEUP after 1024 OSC16K clocks (102.4ms)

[19:18]	**Reserved**	Reserved
[17]	**LIRCDPDEN**	**OSC16K Enabled Control** Determines whether OSC16K is enabled in DPD mode. If OSC16K 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
[15:12]	**Reserved**	Reserved
[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]	**STOP**	**Stop** Reserved – do not set to '1'
[8:4]	**Reserved**	Reserved
[3]	**LIRCEN**	**OSC16K Oscillator Enable Bit** 0 = disable 1 = enable (default)
[2]	**HIRCEN**	**OSC49M 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 (see Protected Register Lock Key Register (SYS_REGLCTL))

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
					ISPCKEN	PDMACKEN	HCLKEN

Table 5-32 AHB Device Clock Enable Register (CLK_AHBCLK, address 0x5000_0204) Bit Description.

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_0000

31	30	29	28	27	26	25	24
Reserved	ANACKEN	I2S0CKEN	ADCCKEN	Reserved	SBRAMCKEN	Reserved	Reserved
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	ACMPCKEN	Reserved	PWM0CH01CKEN	CRCCKEN	BFALCKEN	Reserved	UARTCKEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	Reserved	DPWMCKEN	SPI0CKEN	Reserved	Reserved	Reserved	I2C0CKEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TMR1CKEN	TMR0CKEN	RTCCKEN	WDTCKEN	Reserved	Reserved	Reserved	Reserved

Table 5-33 APB Device Clock Enable Control Register (CLK_APBCLK0, address 0x5000_0208) Bit Description.

Bits	Description	
[30]	**ANACKEN**	**Analog Block Clock Enable Control** 0=Disable 1=Enable
[29]	**I2S0CKEN**	**I2S Clock Enable Control** 0=Disable 1=Enable
[28]	**ADCCKEN**	**Audio Analog-Digital-Converter (ADC) Clock Enable Control** 0=Disable 1=Enable
[26]	**SBRAMCKEN**	**Standby RAM Clock Enable Control** 0=Disable 1=Enable
[22]	**ACMPCKEN**	**Analog Comparator Clock Enable Control** 0=Disable 1=Enable

[20]	**PWM0CH01CKEN**	**PWM Block Clock Enable Control** 0=Disable 1=Enable
[19]	**CRCCKEN**	**Cyclic Redundancy Check Block Clock Enable Control** 0=Disable 1=Enable
[18]	**BFALCKEN**	**Biquad Filter And Automatic Level Control Block Clock Enable Control** 0=Disable 1=Enable
[16]	**UARTCKEN**	**UART0 Clock Enable Control** 0=Disable 1=Enable
[13]	**DPWMCKEN**	**Differential PWM Speaker Driver Clock Enable Control** 0=Disable 1=Enable
[12]	**SPI0CKEN**	**SPI0 Clock Enable Control** 0=Disable 1=Enable
[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]	**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							

Table 5-34 DPD State Register (CLK_DPDSTATE, address 0x5000_020C) Bit Description.

Bits	Description	
[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** To set the CLK_DPDSTATE register, write value to this register. Data is latched on next 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

7	6	5	4	3	2	1	0
Reserved	HIRCFSEL	STCLKSEL			HCLKSEL		

Table 5-35 Clock Source Select Register 0 (CLK_CLKSEL0, address 0x5000_0210) Bit Description.

Bits	Description	
[6]	HIRCFSEL	**OSC48M Frequency Select** Determines which trim setting to use for OSC48M internal oscillator. Oscillator is factory trimmed within 1% to: 0= 49.152MHz (Default) 1= 32.768MHz
[5:3]	STCLKSEL	**MCU Cortex_M0 SysTick Clock Source Select** These bits are protected, to write to bits first perform the unlock sequence (see Protected Register Lock Key Register (SYS_REGLCTL)) 000 aaa clock source from 16 kHz internal clock 001 aaa clock source from external 32kHz crystal clock 010 aaa clock source from 16 kHz internal oscillator divided by 2 011 aaa clock source from OSC49M internal oscillator divided by 2 1xx aaa 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 (see Protected Register Lock Key Register (SYS_REGLCTL)) 000 aaa clock source from internal OSC48M oscillator. 001 aaa clock source from external 32kHz crystal clock 010 aaa clock source from internal 16 kHz oscillator clock Others aaa 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	0x3300_771F

31	30	29	28	27	26	25	24
Reserved		PWM0CH01CKSEL		Reserved			
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			DPWMCKSEL			WDTSEL	

Table 5-36 Clock Source Select Register 1 (CLK_CLKSEL1, address 0x5000_0214) Bit Description.

Bits	Description	
[29:28]	PWM0CH01CKSEL	**PWM0 And PWM1 Clock Source Select** PWM0 and PWM1 uses the same clock source, and prescaler 00 = clock source from internal 16 kHz oscillator 01 = clock source from external 32kHz crystal clock 10 = clock source from HCLK 11 = clock source from internal OSC48M oscillator clock
[14:12]	TMR1SEL	**TIMER1 Clock Source Select** 000 aaa clock source from internal 16 kHz oscillator 001 aaa clock source from external 32kHz crystal clock 010 aaa clock source from HCLK 011 aaa clock source from external pin (GPIOA[15]) 1xx aaa clock source from internal OSC48M oscillator clock
[10:8]	TMR0SEL	**TIMER0 Clock Source Select** 000 aaa clock source from internal 16 kHz oscillator 001 aaa clock source from external 32kHz crystal clock 010 aaa clock source from HCLK 011 aaa clock source from external pin (GPIOA[14]) 1xx aaa clock source from internal OSC48M oscillator clock

[4]	**DPWMCKSEL**	**Differential Speaker Driver PWM Clock Source Select** 0 = OSC48M clock 1 = 2x OSC48M clock
[1:0]	**WDTSEL**	**WDT CLK Clock Source Select** 00 = clock source from internal OSC48M oscillator clock 01 = clock source from external 32kHz crystal clock 10 = clock source from HCLK/2048 clock 11 = clock source from internal 16 kHz oscillator clock

Clock Divider Register (CLK_CLKDIV0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV0	CLK_BA + 0x18	R/W	Clock Divider Number Register	0x0000_0000

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

Table 5-37 Clock Divider Register (CLK_CLKDIV0, address 0x5000_0218) Bit Description.

Bits	Description	
[23:16]	**ADCDIV**	**ADC Clock Divide Number From ADC Clock Source** The ADC clock frequency aaa (ADC clock source frequency) / (ADCDIV + 1)
[11:8]	**UARTDIV**	**UART Clock Divide Number From UART Clock Source** The UART clock frequency aaa (UART clock source frequency) / (UARTDIV + 1)
[3:0]	**HCLKDIV**	**HCLK Clock Divide Number From HCLK Clock Source** The HCLK clock frequency aaa (HCLK 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	0xFFFF_FFFX

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						**I2S0SEL**	

Table 5-38 Clock Source Select Control Register 2 (CLK_CLKSEL2, address 0x5000_021C) Bit Description.

Bits	Description	
[31:2]	Reserved	Reserved
[1:0]	**I2S0SEL**	**I2S Clock Source Select** 00 = clock source from internal 16 kHz oscillator 01 = clock source from external 32kHz crystal clock 10 = clock source from HCLK 11 = clock source from internal OSC48M oscillator clock

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

Table 5-39 Sleep Clock Enable Control Register (CLK_SLEEPCTL, address 0x5000_0220). Bit Description.

31	30	29	28	27	26	25	24
Reserved	ANACKEN	I2S0CKEN	ADCCKEN	Reserved	SBRAMCKEN	Reserved	Reserved
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	ACMPCKEN	Reserved	PWM0CH01CKEN	CRCCKEN	BFALCKEN	Reserved	UARTCKEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	Reserved	DPWMCKEN	SPI0CKEN	Reserved	Reserved	Reserved	I2C0CKEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TMR1CKEN	TMR0CKEN	RTCCKEN	WDTCKEN	Reserved	ISPCKEN	PDMACKEN	HCLKEN

Bits	Description	
[30]	**ANACKEN**	**Analog Block Sleep Clock Enable Control** 0=Disable 1=Enable
[29]	**I2S0CKEN**	**I2S Sleep Clock Enable Control** 0=Disable 1=Enable
[28]	**ADCCKEN**	**Audio Analog-Digital-Converter (ADC) Sleep Clock Enable Control** 0=Disable 1=Enable
[26]	**SBRAMCKEN**	**Standby RAM Sleep Clock Enable Control** 0=Disable 1=Enable
[22]	**ACMPCKEN**	**Analog Comparator Sleep Clock Enable Control** 0=Disable 1=Enable

[20]	**PWM0CH01CKEN**	**PWM Block Sleep Clock Enable Control** 0=Disable 1=Enable
[19]	**CRCCKEN**	**Cyclic Redundancy Check Sleep Block Clock Enable Control** 0=Disable 1=Enable
[18]	**BFALCKEN**	**Biquad filter/ALC block Sleep Clock Enable Control** 0=Disable 1=Enable
[16]	**UARTCKEN**	**UART0 Sleep Clock Enable Control** 0=Disable 1=Enable
[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
[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]	**HCLKEN**	**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

7	6	5	4	3	2	1	0
Reserved					**SPDF**	**STOPF**	**DSF**

Table 5-40 Power State Flag Register (CLK_PWRSTSF, address 0x5000_0224) Bit Description.

Bits	Description	
[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

Table 5-41 Debug Power Down Register (CLK_DBGPD, address 0x5000_0228) Bit Description.

7	6	5	4	3	2	1	0
ICEDATST	**ICECLKST**	**Reserved**					**DISPDREQ**

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

5.4 General Purpose I/O

5.4.1 Overview and Features

Up to 24 General Purpose I/O pins are available on the ISD9100 series. These are shared peripheral special function pins under control of the alternate configuration registers. These 24 pins are arranged in 2 ports named with GPIOA, and GPIOB. GPIOA has sixteen pins and GPIOB has eight. Each one of the 24 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 5-15). 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_MODE. *(x=A or B).* Each pin has two bits of control giving four possible states:

5.4.2.1 *Input Mode*

For Px_MODE.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.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 Px_DOUT is driven to the pin.

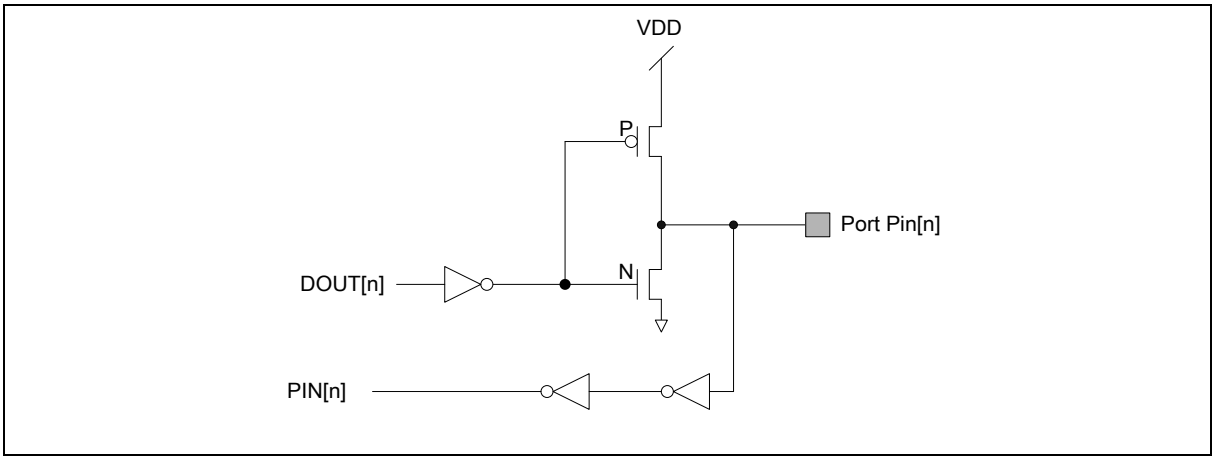


Figure 5-6 Output Mode: Push-Pull Output

5.4.2.3 *Open-Drain Mode*

For Px_MODE.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.

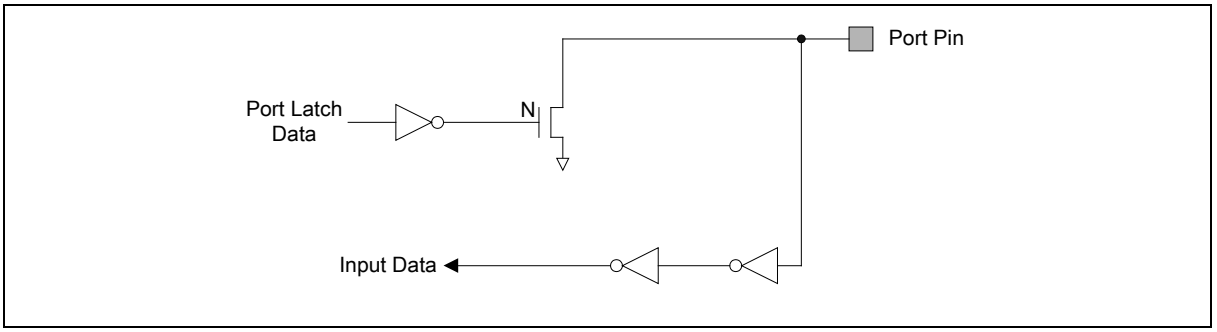


Figure 5-7 Open-Drain Output

5.4.2.4 *Quasi-bidirectional Mode Explanation*

For Px_MODE.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 for 2 clock cycles. After this the pin has an internal pull-up resistor connected. Note that the source current capability in quasi-bidirectional mode is approximately 200uA to 30uA for VDD form 5.0V to 2.4V.

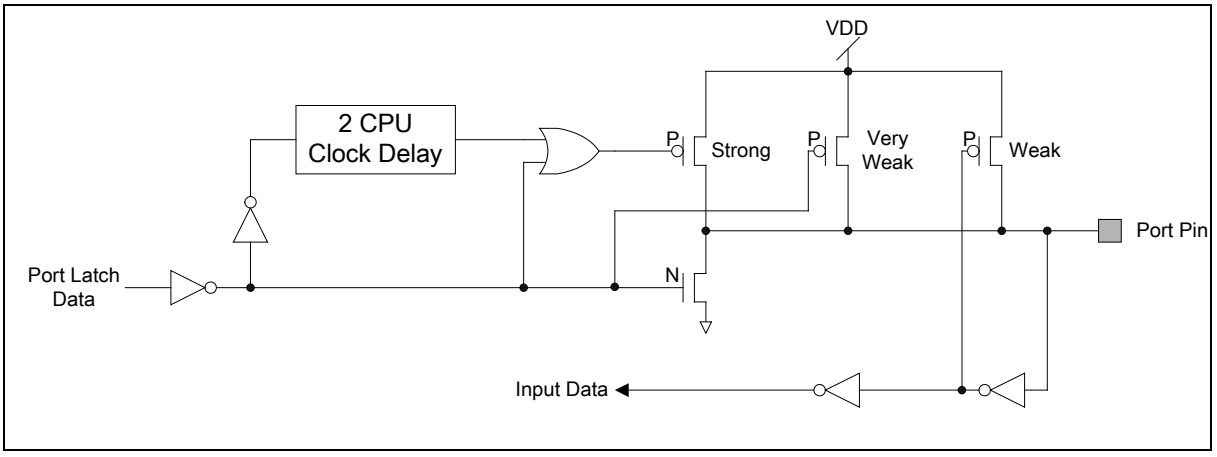


Figure 5-8 Quasi-bidirectional GPIO Mode

5.4.3 GPIO Control Register Map

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

Register	Offset	R/W	Description	Reset Value
GPIO Base Address: **GPIO_BA = 0x5000_4000**				
PA_MODE	GPIO_BA+0x000	R/W	GPIO Port A Pin I/O Mode Control	0xFFFF_FFFF
PA_DINOFF	GPIO_BA+0x004	R/W	GPIO Port A Pin Digital 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 Mode Control	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 Trigger Source Indicator	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 Digital Input Disable	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	GPIO Port B Data Output Value	0x0000_XXFF
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 Mode Control	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 Trigger Source Indicator	0x0000_0000
GPIO_DBCTL	GPIO_BA+0x180	R/W	Interrupt De-bounce Control	0x0000_0020

5.4.4 GPIO Control 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

Table 5-42 GPIO Mode Control Register

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**	**MODE*n***	**GPIO*x* I/O Pin[*n*] Mode Control** Determine each I/O type of GPIO*x* 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 Digital Input Disable	0x0000_0000
PB_DINOFF	GPIO_BA+0x044	R/W	GPIO Port B Pin Digital Input Disable	0x0000_0000

Table 5-43 GPIO Input Disable Register

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

Bits	Description	
[n] **n=16,17..31**	**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_XXFF

Table 5-44 GPIO Data Output Register (Px_DOUT)

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

Bits	Description	
[*n*] **n=0,1..15**	**DOUT**	**GPIOx 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. 0 = GPIO port [A/B] Pin[*n*] will drive Low if the corresponding output mode bit is set. 1 = GPIO port [A/B] Pin[*n*] will drive High 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

Table 5-45 GPIO Data Output Write Mask Register (Px_DATMSK)

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[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATMSK[7:0]							

Bits	Description	
[*n*] **n=0,1..15**	**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 DOUT*n* bit is write-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

Table 5-46 GPIO PIN Value Register (Px_PIN)

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

Bits	Description	
[n] **n=0,1..15**	**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

Table 5-47 GPIO Debounce Enable Register (Px_DBEN)

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[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBEN[7:0]							

Bits	Description	
[*n*] **n=0,1..15**	**DBEN**	**Port [A/B] Input Signal De-bounce Enable** 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 Mode Control	0xXXXX_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	GPIO Port B Interrupt Mode Control	0xXXXX_0000

Table 5-48 GPIO Interrupt Mode Control (Px_INTTYPE)

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

Bits	Description	
[*n*] **n=0,1..15**	**TYPE**	**Port [A/B] Edge Or Level Detection Interrupt Control** 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

Table 5-49 GPIO Interrupt Enable Control Register (Px_INTEN)

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

Bits	Description	
[n+16] **n=0,1..15**	**RHIEN**	**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
[n] **n=0,1..15**	**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 Trigger Source (Px_INTSRC)

Register	Offset	R/W	Description	Reset Value
PA_INTSRC	GPIO_BA+0x020	R/W	GPIO Port A Interrupt Trigger Source Indicator	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	GPIO Port B Interrupt Trigger Source Indicator	0x0000_0000

Table 5-50 GPIO Interrupt Trigger Source Register (Px_INTSRC)

15	14	13	12	11	10	9	8
INTSRC[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTSRC[7:0]							

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

Interrupt De-bounce Control (GPIO_DBCTL)

Register	Offset	R/W	Description	Reset Value
GPIO_DBCTL	GPIO_BA+0x180	R/W	Interrupt De-bounce Control	0x0000_0020

Table 5-51 GPIO Interrupt De-bounce Control Register (GPIO_DBCTL)

7	6	5	4	3	2	1	0
Reserved		**ICLKON**	**DBCLKSRC**	**DBCLKSEL**			

Bits	Description	
[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 aaa 6, then interrupt is sampled every 2^6 aaa 64 de-bounce clocks. If DBCLKSRC is 16KHz oscillator this would be a 64ms de-bounce.

5.5 Brownout Detection and Temperature Alarm

The ISD9100 series is equipped with a Brown-Out voltage detector and Over Temperature Alarm. 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.

TALARM and BOD operation require that the OSC16K low power oscillator is enabled (CLK_PWRCTL.LIRCDPDEN = 0).

The over temperature alarm is designed to protect the chip from dangerously high internal temperatures, generally associated with excessive load (or short circuit) on the speaker driver. The temperature alarm can generate an interrupt to which the CPU can respond and shut down the speaker driver. It is recommended that users implement this function due to the drive strength of the speaker driver has the capability of damaging the chip.

5.5.1 Brownout and Temperature Alarm Register Map

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

Register	Offset	R/W	Description	Reset Value
BOD Base Address: **BODTALM_BA = 0x4008_4000**				
BODTALM_BODSEL	BODTALM_BA+0x00	R/W	Brown Out Detector Select Register	0x0000_0000
BODTALM_BODCTL	BODTALM_BA+0x04	R/W	Brown Out Detector Enable Register	0x0000_00XX
BODTALM_TALMSEL	BODTALM_BA+0x08	R/W	Temperature Alarm Select Register	0x0000_0000
BODTALM_TALMCTL	BODTALM_BA+0x0C	R/W	Temperature Alarm Enable Register	0x0000_00XX
BODTALM_BODDTMR	BODTALM_BA+0x10	R/W	Brown Out Detector Timer Register	0x0003_03E3

Brown-Out Detector Select Register (BODTALM_BODSEL)

Register	Offset	R/W	Description	Reset Value
BODTALM_BODSEL	BODTALM_BA+0x00	R/W	Brown Out Detector Select Register	0x0000_0000

Table 5-52 Brownout Detector Select Register (BODTALM_BODSEL, address 0x4008_4000)

7	6	5	4	3	2	1	0
Reserved				**BODHYS**	**BODVL**		

Bits	Description	
[31:4]	Reserved	Reserved
[3]	**BODHYS**	**BOD Hysteresis** 0= Hysteresis Disabled. 1= Enable Hysteresis of BOD detection.
[2:0]	**BODVL**	**BOD Voltage Level** 111b aaa 4.6V 110b aaa 3.0V 101b aaa 2.8V 100b aaa 2.625V 011b aaa 2.5V 010b aaa 2.4V 001b aaa 2.2V 000b aaa 2.1V

Brown-Out Detector Enable Register (BODTALM_BODCTL)

This register is initialized by user flash configuration bit config0[23] (see Section 6.7). If config0[23]=1, then reset value of BODEN is 0x7. The effect of this is to generate a NMI interrupt (default NMI interrupt is BOD interrupt) if BOD circuit detects a voltage below 2.1V. The NMI ISR can be defined by the user to respond to this low voltage level.

Register	Offset	R/W	Description	Reset Value
BODTALM_BODCTL	BODTALM_BA+0x04	R/W	Brown Out Detector Enable Register	0x0000_00XX

Table 5-53 Detector Enable Register (BODTALM_BODCTL, address 0x4008_4004)

7	6	5	4	3	2	1	0
Reserved			**BODOUT**	**BODIF**	**BODINTEN**	**BODEN**	

Bits	Description	
[31:5]	Reserved	Reserved
[4]	**BODOUT**	**Output of BOD Detection Block** This signal can be monitored to determine the current state of the BOD comparator. Read ‘1’ implies that VCC is less than BODVL.
[3]	**BODIF**	**Current Status Of Interrupt** Latched whenever a BOD event occurs and BODINTEN aaa 1. Write ‘1’ to clear.
[2]	**BODINTEN**	**BOD Interrupt Enable** 0= Disable BOD Interrupt. 1= Enable BOD Interrupt.
[1:0]	**BODEN**	**BOD Enable** 1xb aaa Enable continuous BOD detection. 01b aaa Enable time multiplexed BOD detection. See BODTALM_BODDTMR register. 00b aaa Disable BOD Detection.

Detection Time Multiplex Register (BODTALM_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 OSC16K 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
BODTALM_BODDTMR	BODTALM_BA+0x10	R/W	Brown Out Detector Timer Register	0x0003_03E3

Table 5-54 Detection Time Multiplex Register (BODTALM_BODDTMR, address 0x4008_4010)

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

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)

Temperature Alarm Select Register (BODTALM_TALMSEL)

Register	Offset	R/W	Description	Reset Value
BODTALM_TALMSEL	BODTALM_BA+0x08	R/W	Temperature Alarm Select Register	0x0000_0000

Table 5-55 Temperature Alarm Select Register (BODTALM_TALMSEL, address 0x4008_4008)

7	6	5	4	3	2	1	0
Reserved				**TALMVL**			

Bits	Description	
[31:4]	Reserved	Reserved
[3:0]	**TALMVL**	**Temperature Alarm Sense Level** 0000:105C 0001:115C 0010:125C 0100:135C 1000:145C

Temperature Alarm Enable Register (BODTALM_TALMCTL)

Register	Offset	R/W	Description	Reset Value
BODTALM_TALMCTL	BODTALM_BA+0x0C	R/W	Temperature Alarm Enable Register	0x0000_00XX

Table 5-56 Temperature Alarm Enable Register (BODTALM_TALMCTL, address 0x4008_400C)

7	6	5	4	3	2	1	0
Reserved				**TALMIF**	**TALMIEN**	**TALMOUT**	**TALMEN**

Bits	Description	
[31:4]	Reserved	Reserved
[3]	**TALMIF**	**Current status of interrupt** Latched whenever a Temperature Sense event occurs and IE aaa 1. Write '1' to clear.
[2]	**TALMIEN**	**TALARM Interrupt Enable** 0 = Disable TALARM Interrupt 1 = Enable TALARM Interrupt
[1]	**TALMOUT**	**Output of TALARM Block** Can be polled to determine whether TALARM active (be 1).
[0]	**TALMEN**	**TALARM Enable** 0 = Disable TALARM Detection 1 = Enable TALARM Detection

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). Please refer to Figure 5-9 for more detail I2C BUS Timing.

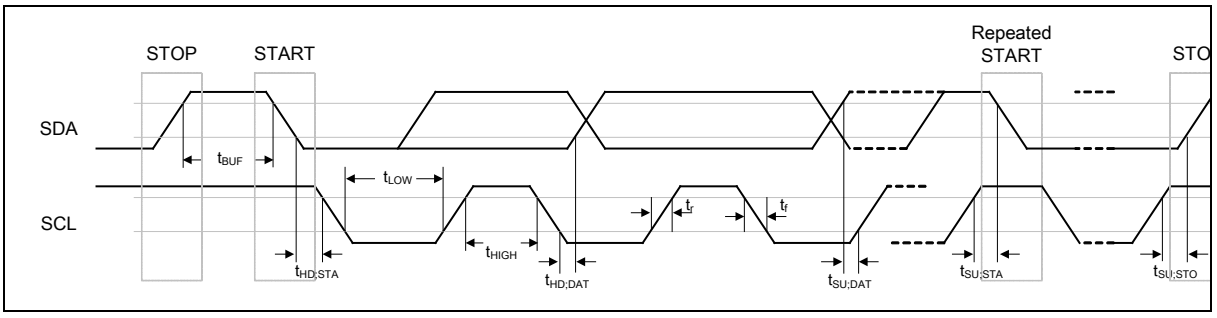


Figure 5-9 I2C Bus Timing

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 ENS1 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. 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

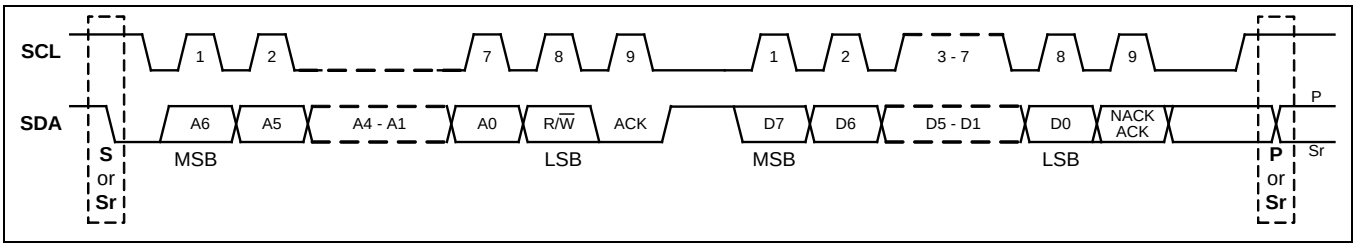


Figure 5-10 I2C Protocol

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, see Figure 5-11.

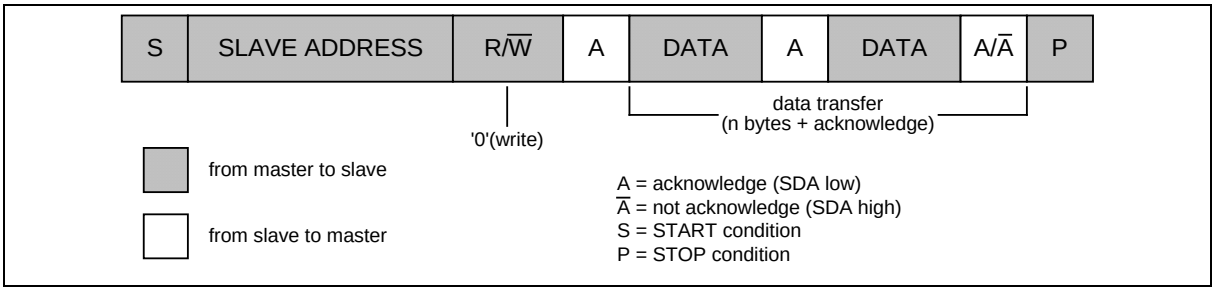


Figure 5-11 Master Transmits Data to Slave

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.

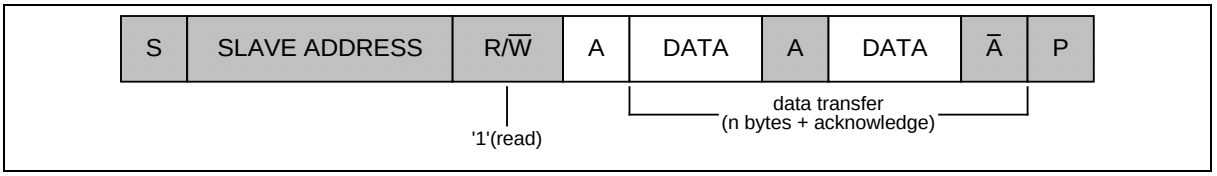


Figure 5-12 Master Reads Data from Slave

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.

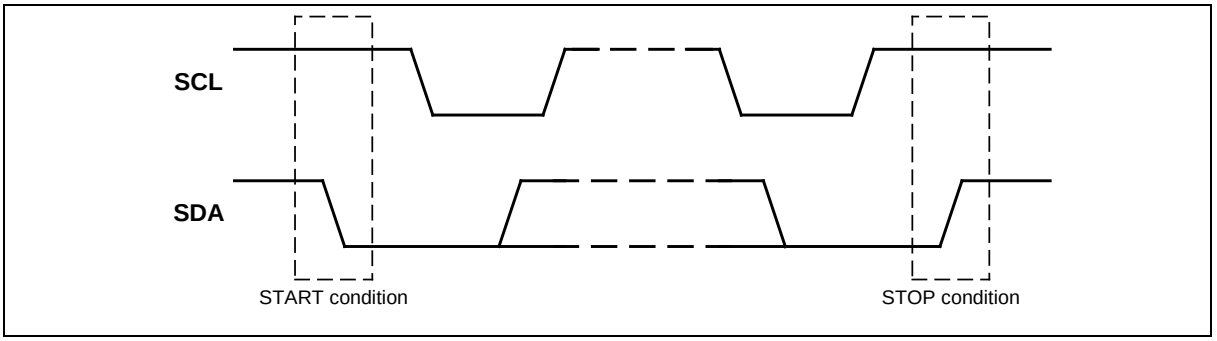


Figure 5-13 START and STOP condition

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.

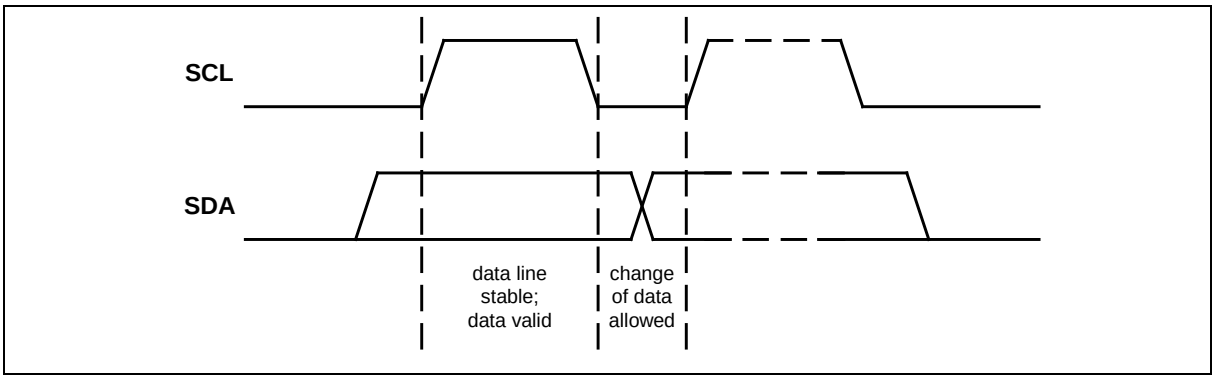


Figure 5-14 Bit Transfer on the I2C bus

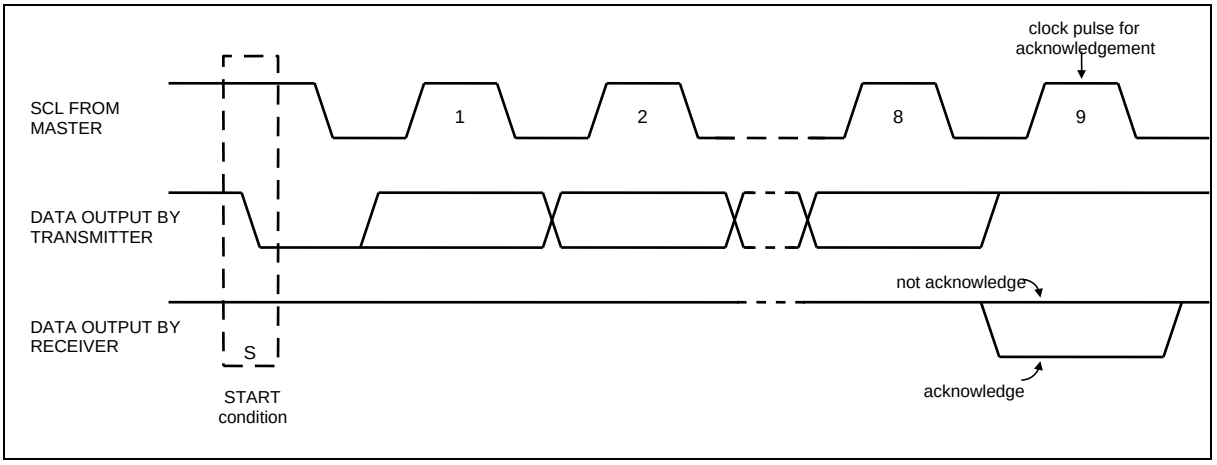


Figure 5-15 Acknowledge on the I2C bus

5.6.2 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), ADDRn (address registers, n=0~3), ADRMn (address mask registers, n=0~3), I2C_CLKDIV (clock rate register) and I2C_TOCTL (Time-out counter register). Bits 31~ bit 8 of these I2C special function registers are reserved. These bits do not have any functions and are all zero if read back.

When I2C port is enabled by setting I2CEN (I2C_CTL[6]) to high, the internal states will be controlled by I2C_CTL and I2C logic hardware. Once a new status code is generated and stored in I2C_STATUS, the I2C Interrupt Flag bit SI (I2C_CTL[3]) will be set automatically. If the Enable Interrupt bit EI (I2C_CTL[7]) is set high at this time, the I2C interrupt will be generated. The bit field I2C_STATUS[7:3] stores the internal state code, the lowest 3 bits of I2C_STATUS are always zero and the contents are stable until SI is cleared by software. The base address of the I2C peripheral on the ISD9100 series is 0x4002_0000.

5.6.2.1 *Address Registers (ADDR)*

I2C port is equipped with four slave address registers ADDRn (n=0~3). The contents of the register are irrelevant when I2C is in master mode. In the slave mode, the bit field ADDRn[7:1] must be loaded with the MCU's own slave address. The I2C hardware will react if the contents of ADDR are matched with the received slave address.

The I2C ports support the "General Call" function. If the GC bit (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.

5.6.2.2 *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.

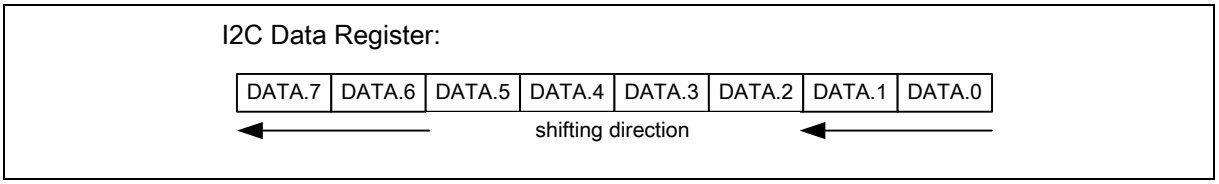


Figure 5-16 I2C Data Shift Direction

5.6.2.3 *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 ENS1 = "0".

INTEN Enable Interrupt.

I2CEN Set to enable I2C serial function block. When ENS=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 2C 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.

5.6.2.4 *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.

5.6.2.5 *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 400 kHz 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.

5.6.2.6 *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 (TIF=1) and generates I2C interrupt to CPU or stops counting by clearing ENTI 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 5-17 for the 14-bit time-out counter. User can clear TIF by writing one to this bit.

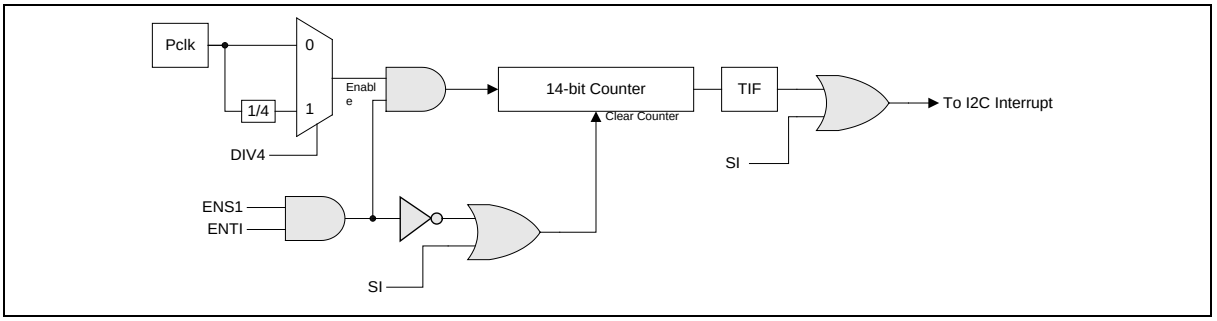


Figure 5-17: I2C Time-out Count Block Diagram

5.6.3 Register Mapping

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_0000
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

5.6.4 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

7	6	5	4	3	2	1	0
INTEN	I2CEN	STA	STO	SI	AA	Reserved	Reserved

Bits	Description	
[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 EI (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 aaa 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.

I2C DATA REGISTER (I2C_DAT)

Register	Offset	R/W	Description	Reset Value
I2C_DAT	I2C_BA+0x08	R/W	I2C DATA Register	0x0000_0000

7	6	5	4	3	2	1	0
DAT [7:0]							

Bits	Description	
[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_0000

7	6	5	4	3	2	1	0
STATUS[7:0]							

Bits	Description	
[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 STATUS contains F8H, no serial interrupt is requested. All other STATUS values correspond to defined I2C states. When each of these states is entered, a status interrupt is requested (SI aaa 1). A valid status code is present in 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 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

7	6	5	4	3	2	1	0
DIVIDER[7:0]							

Bits	Description	
[7:0]	**DIVIDER**	**I2C Clock Divided Register** The I2C clock rate bits: Data Baud Rate of I2C aaa PCLK /(4x(CLK+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

7	6	5	4	3	2	1	0
Reserved					**TOCEN**	**TOCDIV4**	**TOIF**

Bits	Description	
[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 (I2C_ADDRx)

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

7	6	5	4	3	2	1	0
ADDR[7:1]							**GC**

Bits	Description	
[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 (I2C_ADDRMSKx)

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

7	6	5	4	3	2	1	0
ADDRMSKx[7:1]							**Reserved**

Bits	Description	
[n] **n=1,2..7**	**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 I2C_ADDRx registers masking bits from the address comparison.

5.6.5 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.

5.6.5.1 *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.

5.6.5.2 *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.

5.6.5.3 *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.

5.6.5.4 *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.

5.6.6 Data Transfer Flow in Five Operating Modes

The five operating modes are: Master/Transmitter, Master/Receiver, Slave/Transmitter, Slave/Receiver and GC Call. Bits STA, STO and AA in I2C_CTL register will determine the next state of the SIO hardware after SI flag is cleared. Upon completion of the new action, a new status code will be updated and the SI flag will be set. If the I2C interrupt control bit INTEN (I2C_CTL[7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

Data transfers in each mode are shown in the following figures.

*** Legend for the following five figures:

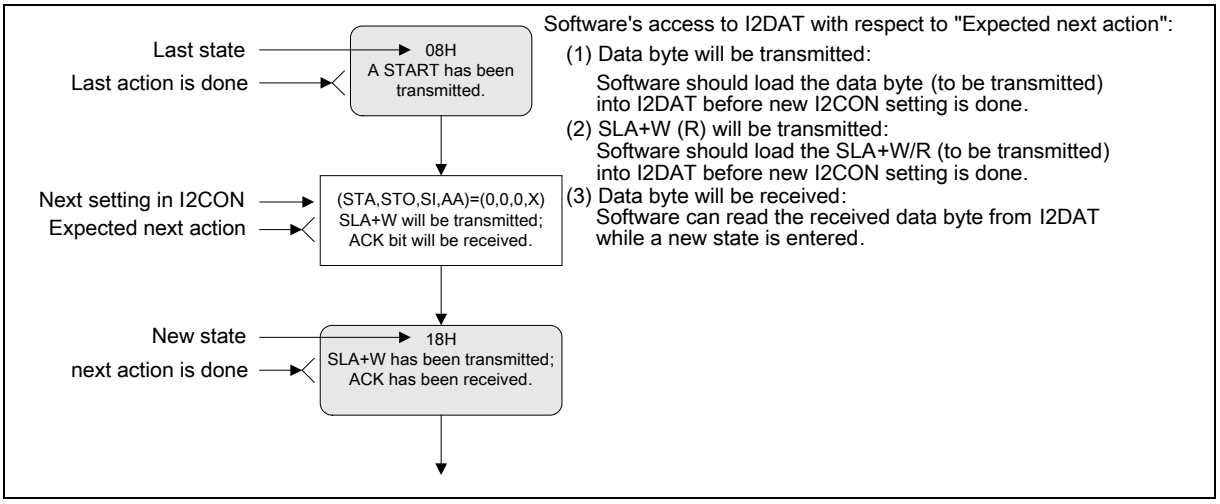


Figure 5-18 Legend for the following four figures

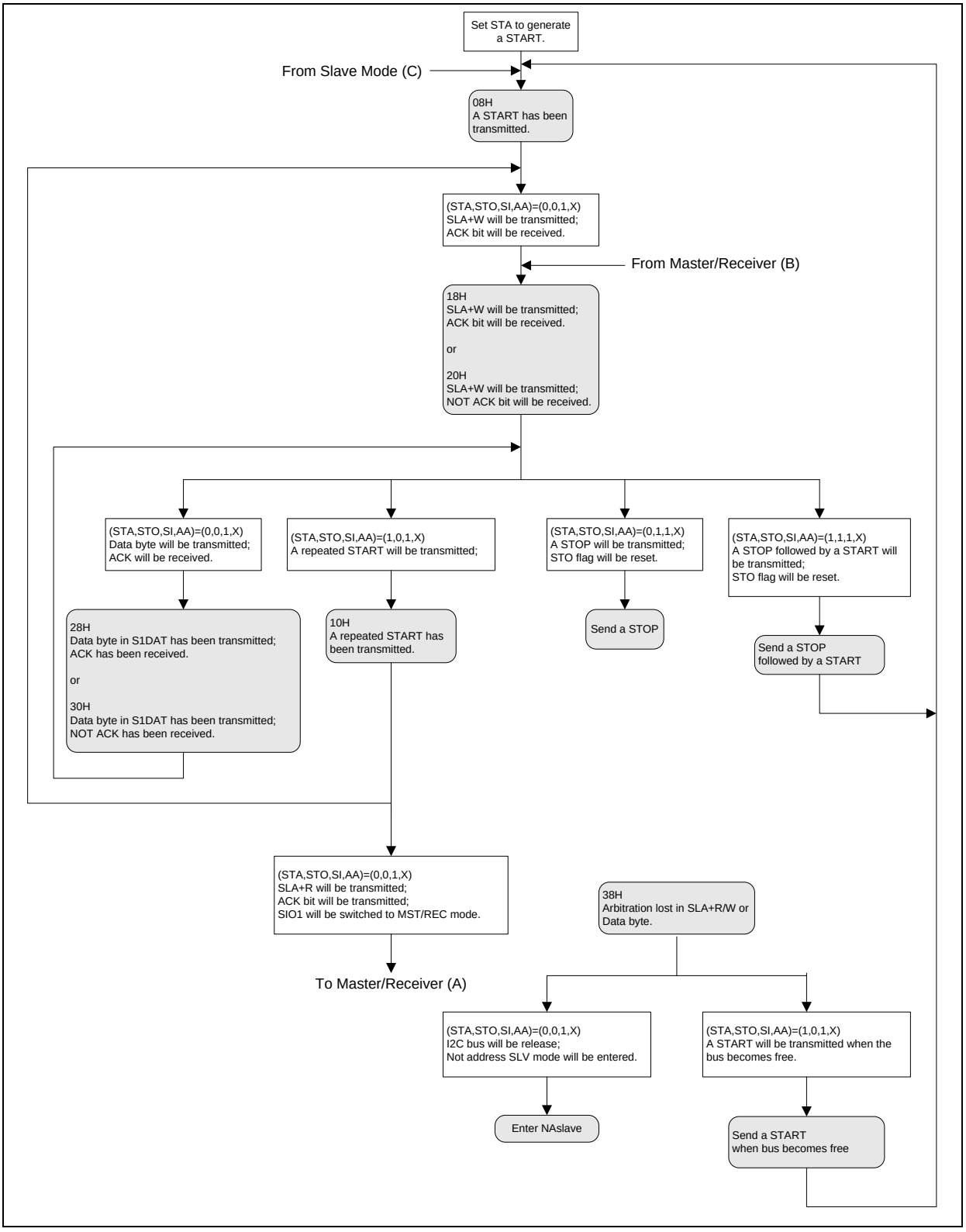


Figure 5-19 Master Transmitter Mode

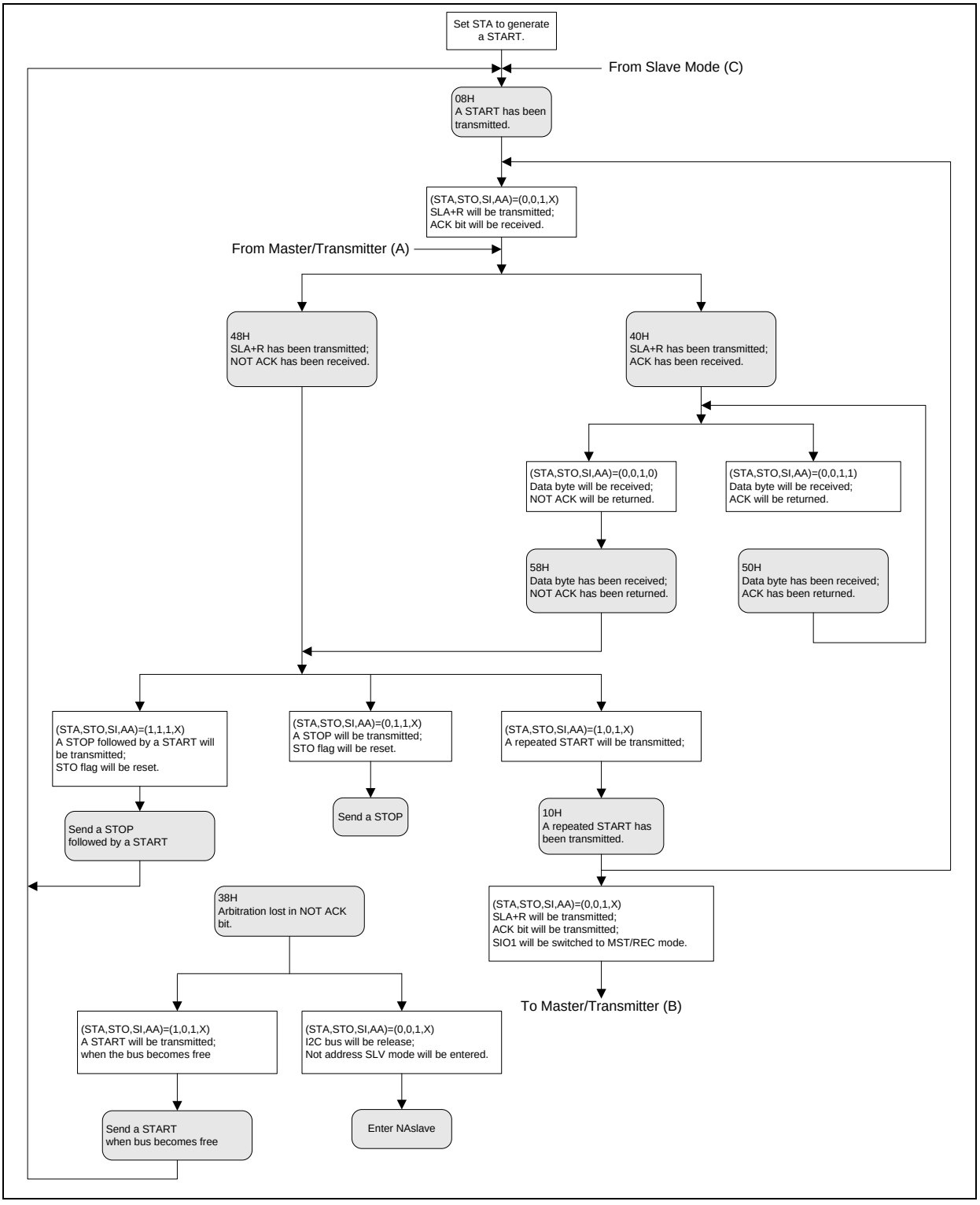


Figure 5-20 Master Receiver Mode

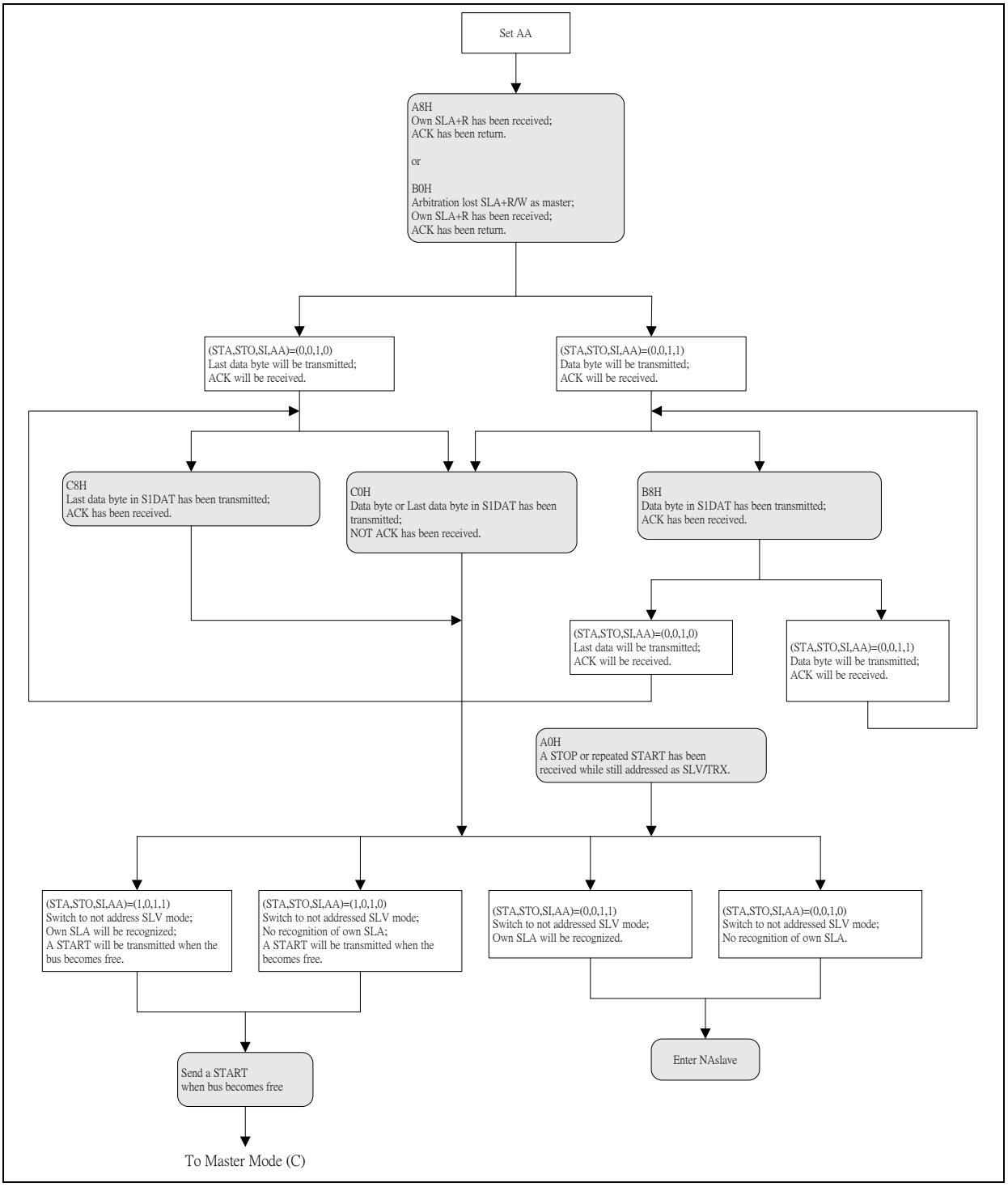


Figure 5-21 Slave Transmitter Mode

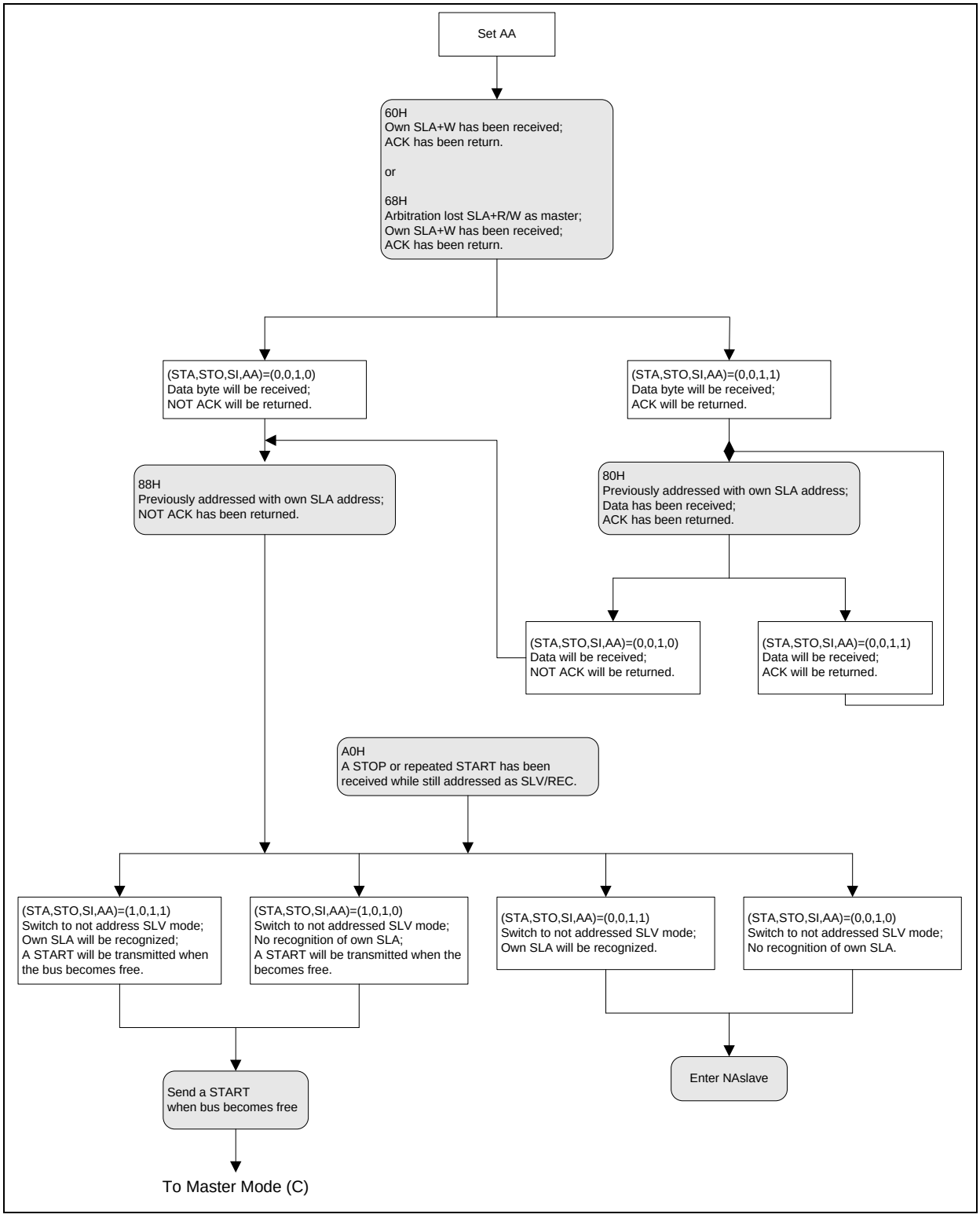


Figure 5-22 Slave Receiver Mode

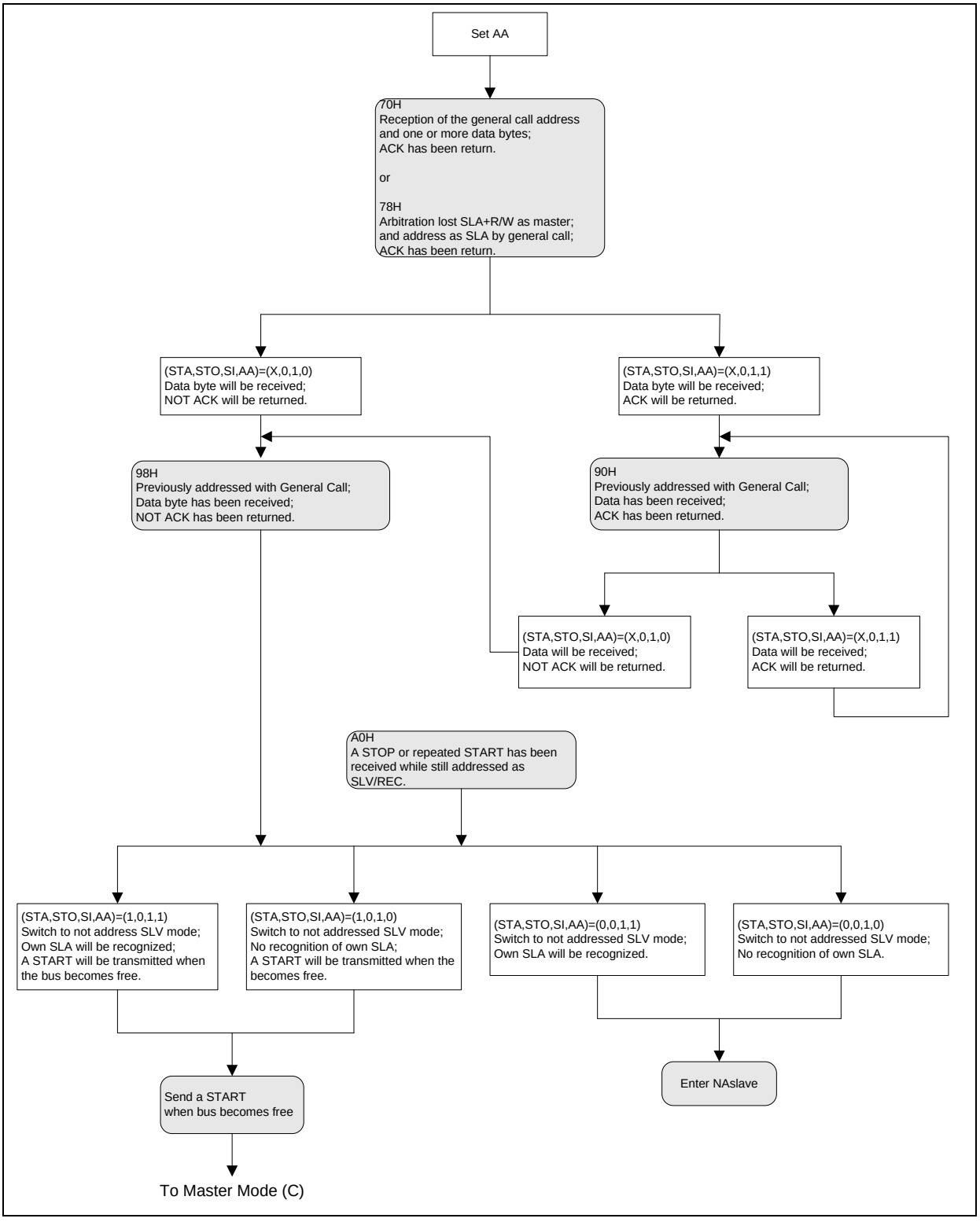


Figure 5-23 GC Mode

5.7 PWM Generator and Capture Timer

5.7.1 Introduction

The ISD9100 series has a PWM Generator which can be configured as 2 independent PWM outputs, PWM0~PWM1, or as a complementary PWM pair, (PWM0, PWM1) with a programmable dead-zone generator. The PWM Generator has an 8-bit prescaler, 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, PWM0 and PWM1 perform complementary paired PWM function; the paired PWM timing, period, duty and dead-time are determined by PWM0 timer and Dead-zone generator 0. Refer to Figure 5-25 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 (PWM_PERIODx) 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 PWM0 share one timer which is included in PWM0; and the Capture1 and PWM1 share PWM1 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 (PWM_RCAPDATx) 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 (Rising latch Interrupt enable) and PWM_CAPCTL01.CFLIEN0 (Falling latch Interrupt enable) to determine the condition of interrupt occurrence. Capture channel 1 has the same feature by setting PWM_CAPCTL01.CRLIEN1and PWM_CAPCTL01.CFLIEN1. 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 2 PWM channels or a 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 (FCAPDAT) and Capture interrupt flag (CAPIFx)

5.7.3 PWM Generator Architecture

The following figures illustrate the architecture of the PWM.

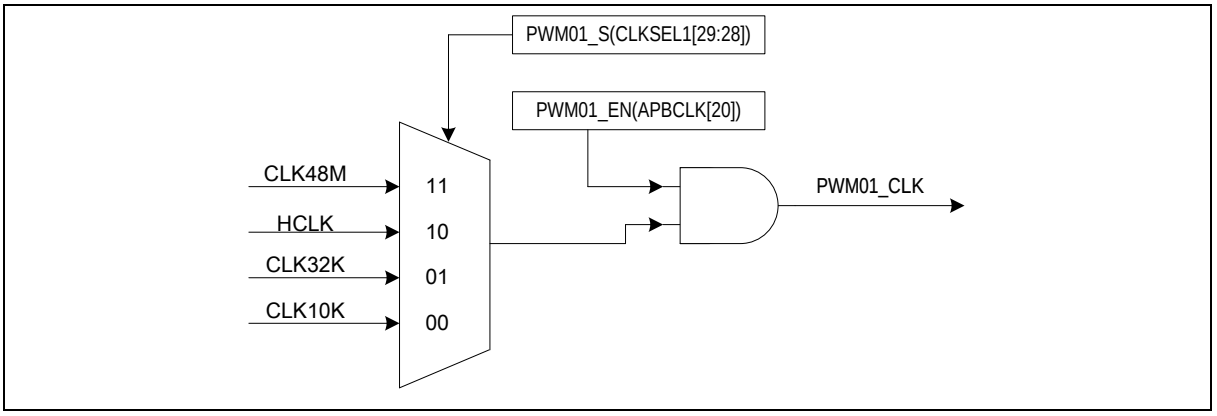


Figure 5-24 PWM Generator Clock Source Control

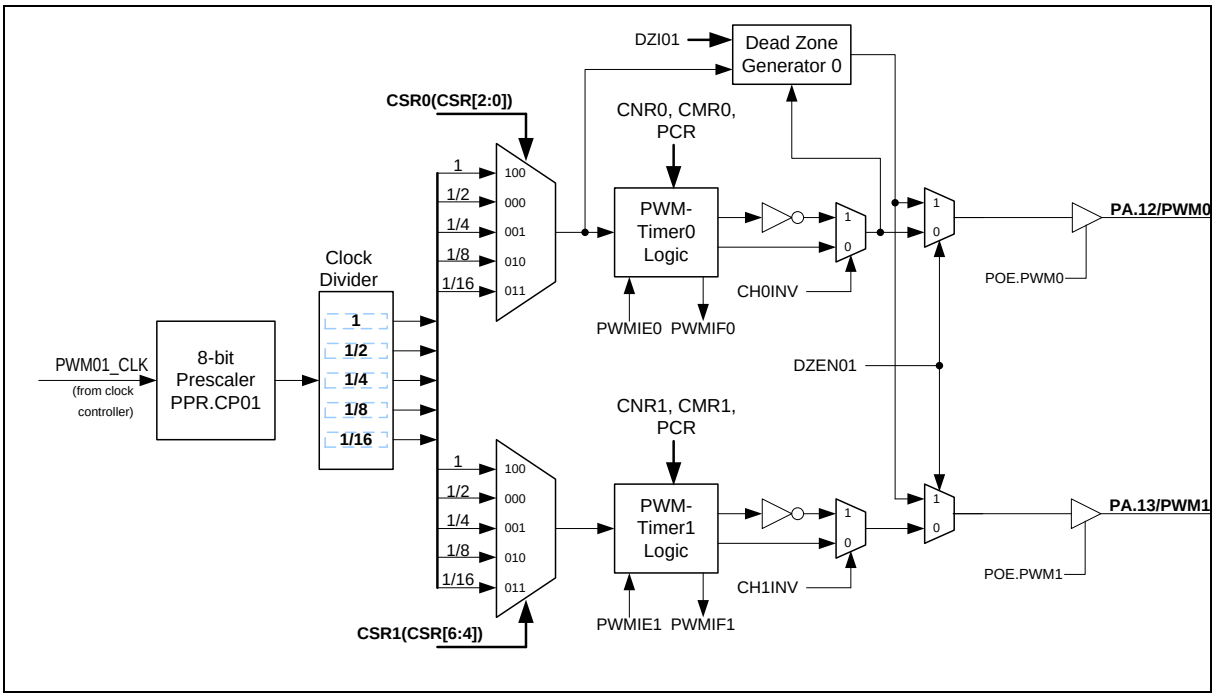


Figure 5-25 PWM Generator Architecture Diagram

5.7.4 PWM-Timer Operation

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

- PWM frequency = PWM01_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.

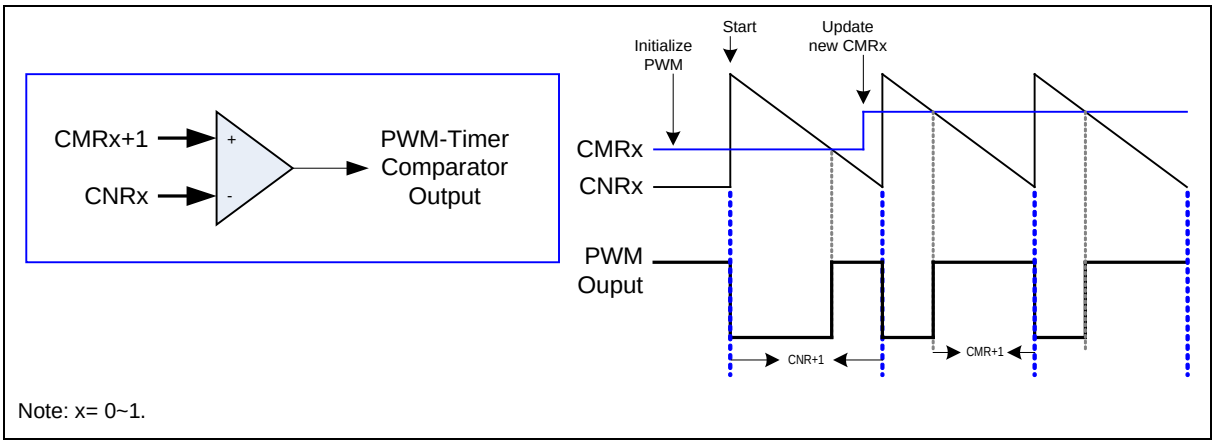


Figure 5-26 PWM Generation Timing

The procedure to operate the PWM generator is shown in Figure 5-27. 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.

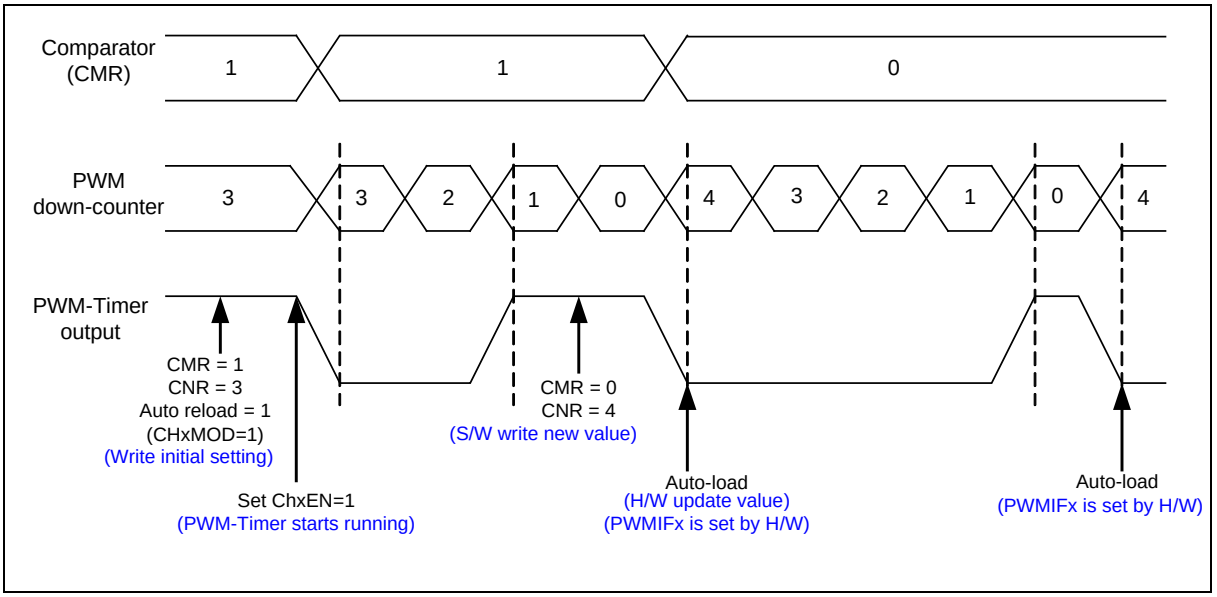


Figure 5-27 PWM-Timer Operation Timing

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

The ISD9100 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_PERIODx and current PWM counter value can be read from PWM_CNTx.

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.

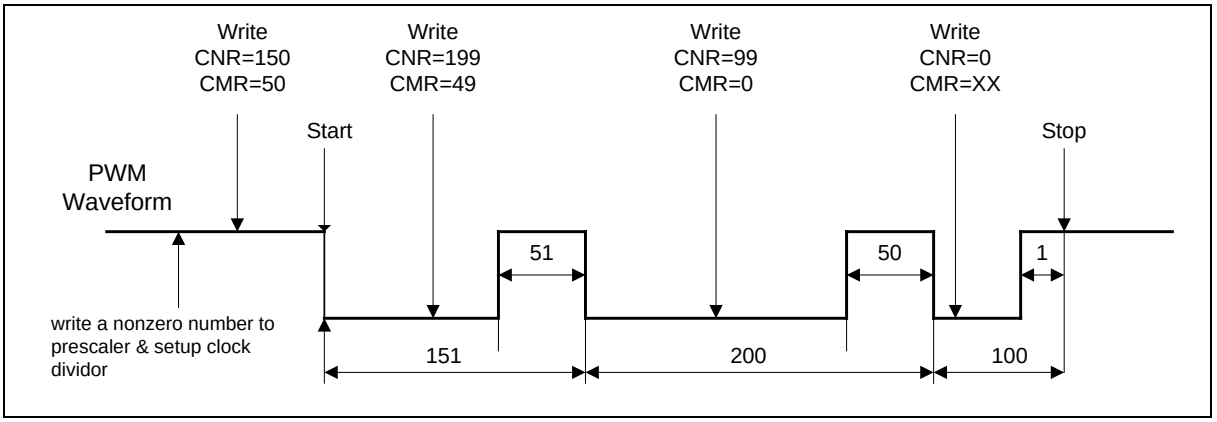


Figure 5-28 PWM Double Buffering.

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.

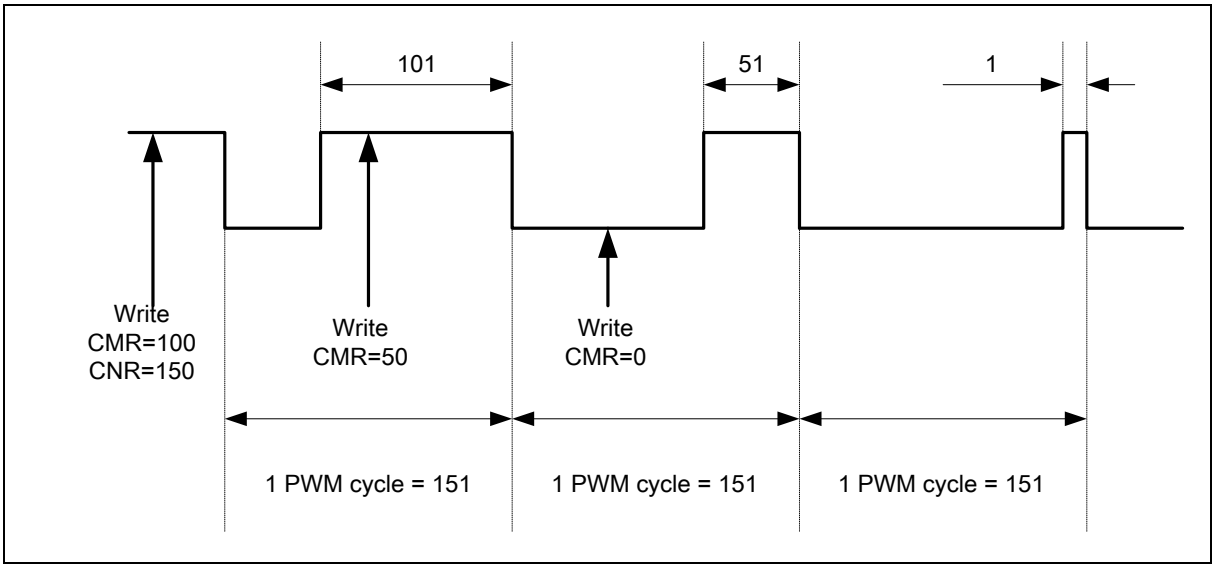


Figure 5-29 PWM Controller Duty Cycle Modulation (PERIOD = 150).

5.7.7 Dead-Zone Generator

The ISD9100 series 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 PWM_CLKPSC.DTCNT01 to determine the Dead Zone interval. The Dead Zone generator behavior is demonstrated in Figure 5-30.

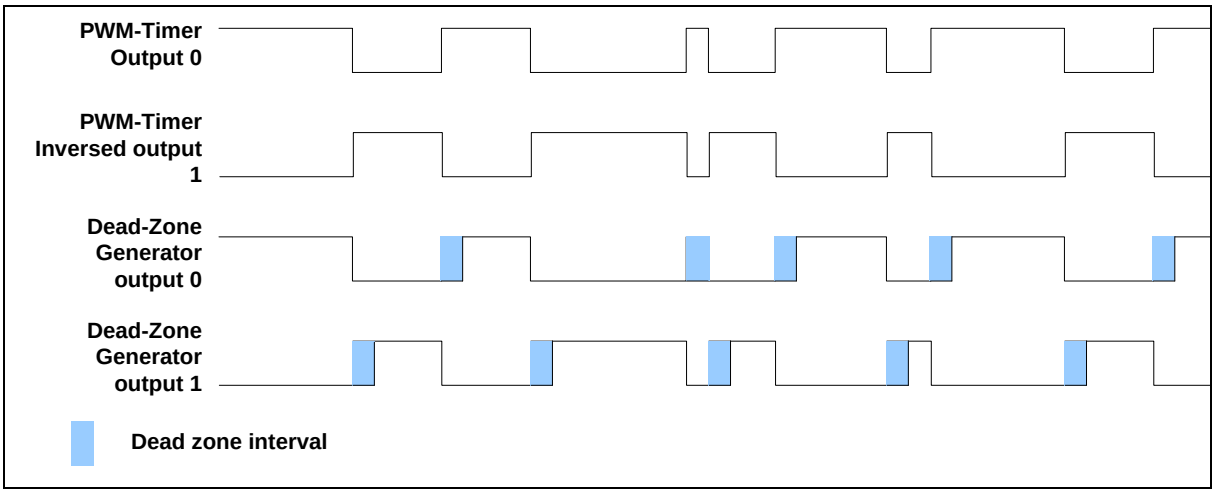


Figure 5-30 Paired-PWM Output with Dead Zone Generation Operation

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 PWM0 share one timer and Capture channel 1 and PWM1 share another timer. The capture timer latches PWM-counter to RCAPDAT when input channel has a rising transition and latches PWM-counter to FCAPDAT 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.

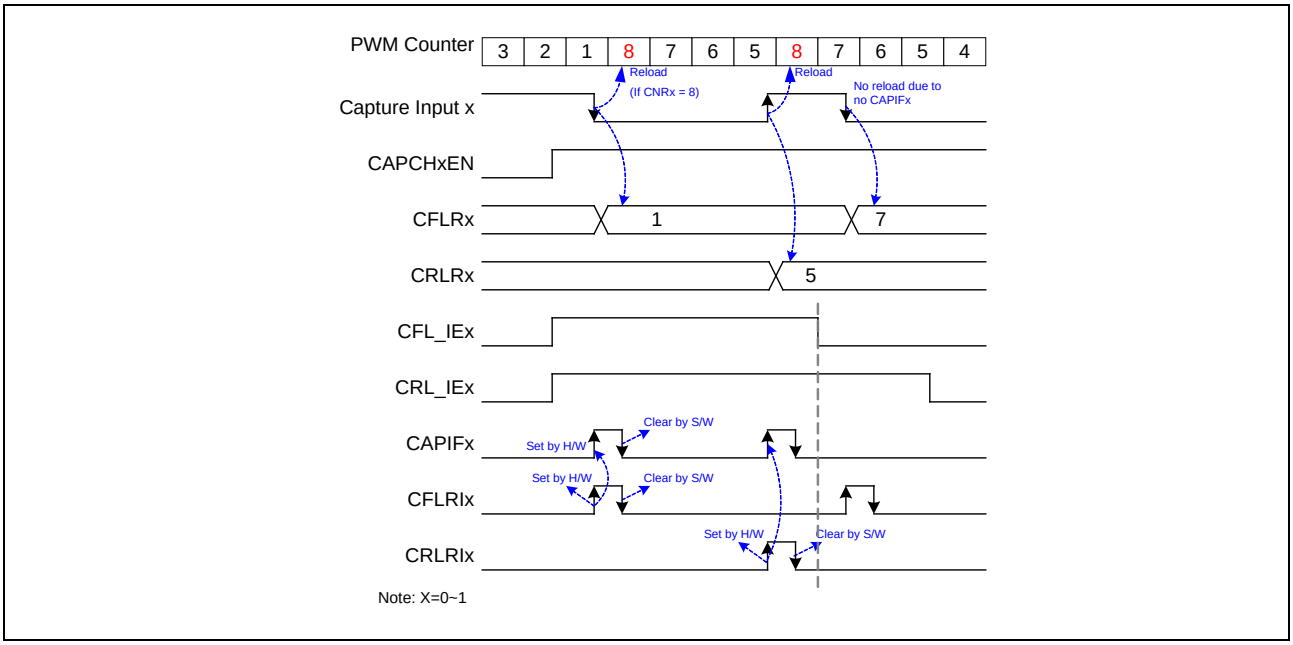


Figure 5-31 Capture Operation Timing

Figure 5-31 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 PWM_IRQ. PWM 0 and Capture 0 share one interrupt, PWM1 and Capture 1 share the same interrupt. Figure 5-32 demonstrates the architecture of PWM-Timer interrupts.

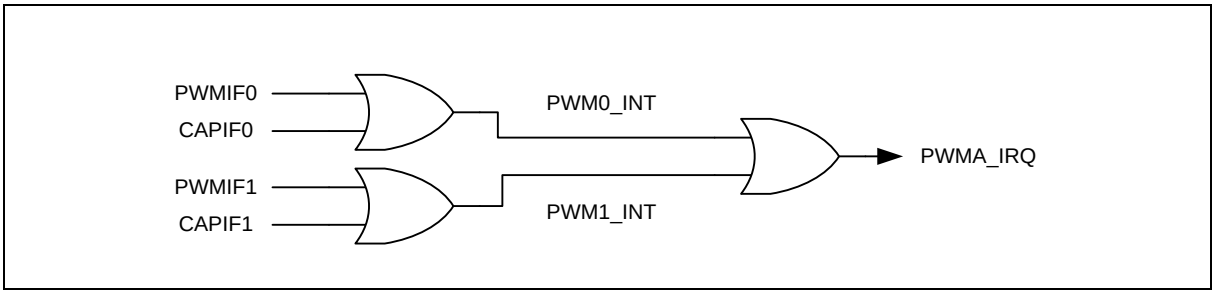


Figure 5-32 PWM-Timer Interrupt Architecture Diagram

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_CMPDATx) to set PWM duty cycle.
5. Setup PWM down-counter register (PWM_PERIODx) 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 (PERIOD) as 0, and monitor CNT (current value of 16-bit down-counter). When CNT reaches to 0, disable PWM-Timer (CNTENx in PWM_CTL). ***(Recommended)***

Method 2:

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

Method 3:

Disable PWM-Timer directly (CNTENx in PWM_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)
4. Setup PWM down-counter (PWM_PERIODx)
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, **C**: Only value 0 can be written

Register	Offset	R/W	Description	Reset Value
PWMA 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_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 0	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_CAPINEN	PWM_BA+0x078	R/W	Capture Input Enable Register	0x0000_0000
PWM_POEN	PWM_BA+0x07C	R/W	PWM Output Enable Register for PWM0~PWM1	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
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DTCNT01							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC01							

Table 5-57 PWM Pre-Scaler Register (PWM_CLKPSC, address 0x4004_0000).

Bits	Description	
[31:24]	**Reserved**	Reserved
[23:16]	**DTCNT01**	**Dead Zone Interval Register For Pair Of PWM0 And PWM1** These 8 bits determine dead zone length. The unit time of dead zone length is that from clock selector 0.
[15:8]	**Reserved**	Reserved
[7:0]	**CLKPSC01**	**Clock Pre-scaler** Clock input is divided by (CLKPSC01 + 1). If CLKPSC01 aaa 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

7	6	5	4	3	2	1	0
Reserved	**CLKDIV1**			**Reserved**	**CLKDIV0**		

Table 5-58 PWM Clock Select Register (PWM_CLKDIV, address 0x4004_0004).

Bits	Description	
[31:7]	**Reserved**	Reserved
[6:4]	**CLKDIV1**	**Timer 1 Clock Source Selection** Value : Input clock divided by 0 : 2 1 : 4 2 : 8 3 : 16 4 : 1
[2:0]	**CLKDIV0**	**Timer 0 Clock Source Selection** (Table is as CLKDIV1)

PWM Control Register (PWM_CTL)

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

15	14	13	12	11	10	9	8
Reserved				CNTMODE1	PINV1	Reserved	CNTEN1
7	6	5	4	3	2	1	0
Reserved			DTEN01	CNTMODE0	PINV0	Reserved	CNTEN0

Table 5-59 PWM Control Register (PWM_CTL, address 0x4004_008).

Bits	Description	
[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
[8]	CNTEN1	**PWM-Timer 1 Enable/Disable Start Run** 0 = Stop PWM-Timer 1 1 = Enable PWM-Timer 1 Start/Run
[4]	DTEN01	**Dead-Zone 0 Generator Enable/Disable** 0 = Disable 1 = Enable Note: When Dead-Zone Generator is enabled, the pair of PWM0 and PWM1 become a complementary pair.
[3]	CH0MOD	**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]	CH0INV	**PWM-Timer 0 Output Inverter ON/OFF** 0 = Inverter OFF 1 = Inverter ON

[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_PERIODx)

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

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

Table 5-60 PWM Counter Register (PWM_PERIODx, address 0x4004_00C+C*x).

Bits	Description	
[15:0]	**PERIOD**	**PWM Counter/Timer Reload Value** PERIOD determines the PWM period. PWM frequency aaa PWM01_CLK/(prescale+1)*(clock divider)/(PERIOD+1); Duty ratio aaa (CMP+1)/(PERIOD+1). CMP > aaa PERIOD: PWM output is always high. CMP < PERIOD: PWM low width aaa (PERIOD-CMP) unit; PWM high width aaa (CMP+1) unit. CMP aaa 0: PWM low width aaa (PERIOD) unit; PWM high width aaa 1 unit (Unit aaa one PWM clock cycle) **Note:** Any write to PERIOD will take effect in next PWM cycle.

PWM Comparator Register (PWM_CMPDATx)

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

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

Table 5-61 PWM Comparator Register (PWM_CMPDATx, address 0x4004_0010 + C*x).

Bits	Description	
[15:0]	**CMP**	**PWM Comparator Register** CMP determines the PWM duty cycle. PWM frequency aaa PWM01_CLK/(prescale+1)*(clock divider)/(PERIOD+1); Duty Cycle aaa (CMP+1)/(PERIOD+1). CMP > aaa PERIOD: PWM output is always high. CMP < PERIOD: PWM low width aaa (PERIOD-CMP) unit; PWM high width aaa (CMP+1) unit. CMP aaa 0: PWM low width aaa (PERIOD) unit; PWM high width aaa 1 unit (Unit aaa one PWM clock cycle) Note: Any write to CMP will take effect in next PWM cycle.

PWM Data Register 1-0 (PWM_CNTx)

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

15	14	13	12	11	10	9	8
CNT[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[7:0]							

Table 5-62 PWM Data Register (PWM_CNTx, address 0x4004_0014 + C*x).

Bits	Description	
[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

7	6	5	4	3	2	1	0
Reserved						**PIEN1**	**PIEN0**

Table 5-63 PWM Interrupt Enable Register (PWM_INTEN, address 0x4004_0040).

Bits	Description	
[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

7	6	5	4	3	2	1	0
Reserved						**PIF1**	**PIF0**

Table 5-64 PWM Interrupt Flag Register (PWM_INTSTS, address 0x4004_0044).

Bits	Description	
[1]	**PIF1**	**PWM Timer 1 Interrupt Flag** Flag is set by hardware when PWM1 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 PWM0 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 0	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

Table 5-65 Capture Control Register (PWM_CAPCTL01, address 0x4004_0050).

Bits	Description	
[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 aaa 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 aaa 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
[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 aaa 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 aaa 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 Rising Latch Register1-0 (PWM_RCAPDATx)

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

15	14	13	12	11	10	9	8
RCAPDAT[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RCAPDAT[7:0]							

Table 5-66 Capture Rising Latch Register (PWM_RCAPDATx, address 0x4004_0058 +C*x).

Bits	Description	
[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 Register1-0 (PWM_FCAPDATx)

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

15	14	13	12	11	10	9	8
FCAPDAT[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FCAPDAT[7:0]							

Table 5-67 Capture Falling Latch Register (PWM_FCAPDATx, address 0x4004_005C + C*x).

Bits	Description	
[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

7	6	5	4	3	2	1	0
Reserved						**CAPINEN[1:0]**	

Table 5-68 Capture Input Enable Register (PWM_CAPINEN, address 0x4004_0078).

Bits	Description	
[1:0]	**CAPINEN**	**Capture Input Enable Register** 0 : OFF (PA[13:12] pin input disconnected from Capture block) 1 : ON (PA[13:12] pin, if in PWM alternative function, will be configured as an input and fed to capture function) CAPINEN[1:0] Bit 10 Bit x1 : Capture channel 0 is from PA [12] Bit 1x : Capture channel 1 is from PA [13]

PWM Output Enable Register (PWM_POEN)

Register	Offset	R/W	Description	Reset Value
PWM_POEN	PWM_BA+0x07C	R/W	PWM Output Enable Register for PWM0~PWM1	0x0000_0000

7	6	5	4	3	2	1	0
Reserved						**POEN1**	**POEN0**

Table 5-69 PWM Output Enable (PWM_POEN, address 0x4004_007C).

Bits	Description	
[1]	**POEN1**	**PWM1 Output Enable Register** 0 = Disable PWM1 output to pin. 1 = Enable PWM1 output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function.
[0]	**POEN0**	**PWM0 Output Enable Register** 0 = Disable PWM0 output to pin. 1 = Enable PWM0 output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function.

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.TICKSEL. 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.AIF) 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).

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

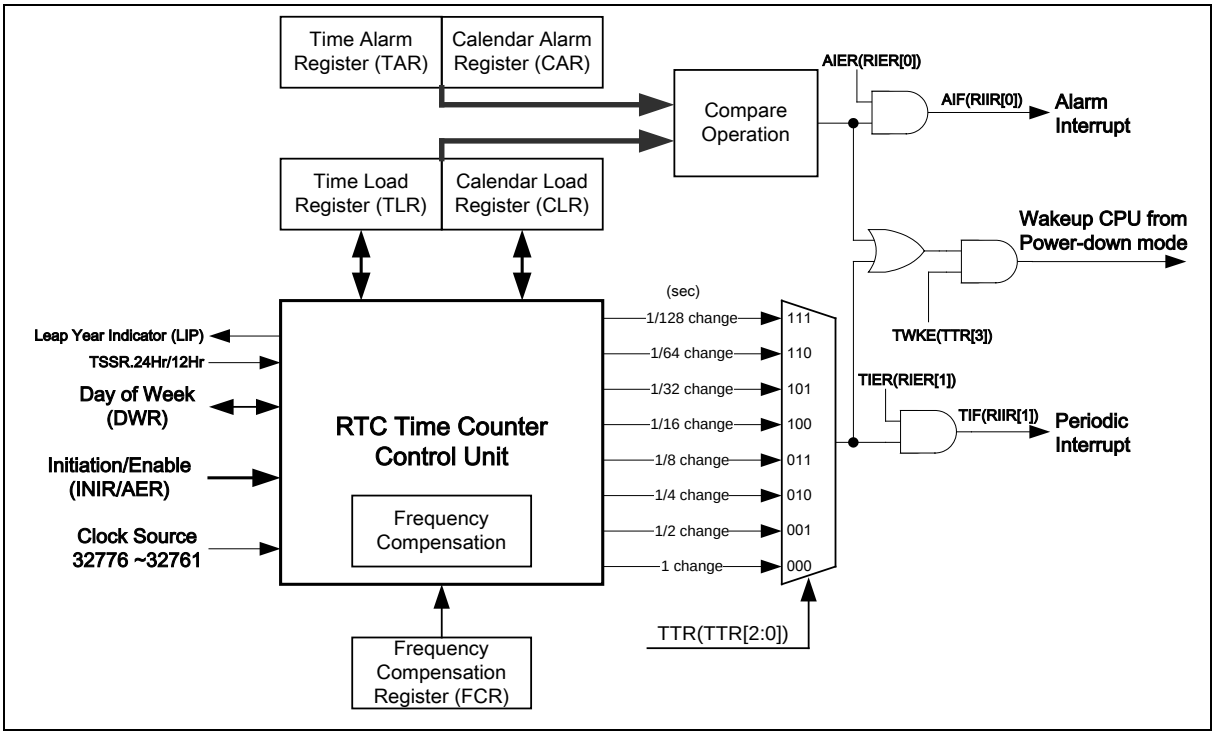


Figure 5-33 RTC Block Diagram

5.8.4 RTC Function 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 to 0xA5EB1357, 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.RWENF 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.

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 Table 5-76 for details. The 12/24 hour time scale selection depends on RTC_CLKFMT bit 0.

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. When periodic time tick interrupt is enabled by setting RTC_INTEN.TICKIEN 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.AIF) 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

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	0x0005_0101
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_0006
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**

Table 5-70 RTC Initialization Register (RTC_INIT, address 0x4000_8000).

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							

Table 5-71 RTC Access Enable Register (RTC_RWEN, address 0x4000_8004).

Bits	Description	
[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] ! aaa 0xA965. The effect of RTC_RWEN.RWENF on access to each register is given Table 5-72. Table 5-72 RTC_RWEN.RWENF Register Access Effect. Register : RWENF aaa 1 : RWENF aaa 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 aaa Enable RTC access Others aaa Disable RTC access

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**					

Table 5-73 RTC Frequency Compensation Register (RTC_FREQADJ, address 0x4000_8008).

Bits	Description	
[11:8]	**INTEGER**	**Integer Part** Register should contain the value (INT(F_{actual}) – 32761) Ex: Integer part of detected value aaa 32772, RTC_FREQADJ.INTEGER aaa 32772-32761 aaa 11 (1011b) The range between 32761 and 32776
[5:0]	**FRACTION**	**Fractional Part** Formula aaa (fraction part of detected value) x 60 Refer to 5.8.4.4 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

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**			

Table 5-74 RTC Time Load Register (RTC_TIME, address 0x4000_800C).

Bits	Description	
[21:20]	**TENHR**	10 Hour Time Digit (0~3)
[19:16]	**HR**	1 Hour Time Digit (0~9)
[14:12]	**TENMIN**	10 Min Time Digit (0~5)
[11:8]	**MIN**	1 Min Time Digit (0~9)
[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	0x0005_0101

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**			

Table 5-75 RTC Calendar Load Register (RTC_CAL, address 0x4000_80010).

Bits	Description	
[23:20]	**TENYEAR**	10-Year Calendar Digit (0~9)
[19:16]	**YEAR**	1-Year Calendar Digit (0~9)
[12]	**TENMON**	10-Month Calendar Digit (0~1)
[11:8]	**MON**	1-Month Calendar Digit (0~9)
[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

Table 5-76 RTC Time Scale Selection Register (RTC_CLKFMT, address 0x4000_8014).

7	6	5	4	3	2	1	0
Reserved							**24HEN**

Bits	Description	
[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_0006

7	6	5	4	3	2	1	0
Reserved					**WEEKDAY**		

Table 5-77 RTC Day of Week Register (RTC_WEEKDAY, address 0x4000_8018).

Bits	Description	
[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			

Table 5-78 RTC Time Alarm Register (RTC_TALM, address 0x4000_801C).

Bits	Description	
[21:20]	TENHR	10 Hour Time Digit of Alarm Setting (0~3)
[19:16]	HR	1 Hour Time Digit of Alarm Setting (0~9)
[14:12]	TENMIN	10 Min Time Digit of Alarm Setting (0~5)
[11:8]	MIN	1 Min Time Digit of Alarm Setting (0~9)
[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			

Table 5-79 RTC Calendar Alarm Register (RTC_CALM, address 0x4000_8020).

Bits	Description	
[23:20]	TENYEAR	10-Year Calendar Digit of Alarm Setting (0~9)
[19:16]	YEAR	1-Year Calendar Digit of Alarm Setting (0~9)
[12]	TENMON	10-Month Calendar Digit of Alarm Setting (0~1)
[11:8]	MON	1-Month Calendar Digit of Alarm Setting (0~9)
[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

7	6	5	4	3	2	1	0
Reserved							**LEAPYEAR**

Table 5-80 RTC Leap Year Indicator Register (RTC_LEAPYEAR, address 0x4000_8024).

Bits	Description	
[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

7	6	5	4	3	2	1	0
Reserved						**TICKIEN**	**ALMIEN**

Table 5-81 RTC Interrupt Enable Register (RTC_INTEN, address 0x4000_8028).

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

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

7	6	5	4	3	2	1	0
Reserved						TICKIF	ALMIF

Table 5-82 RTC Interrupt Indication Register (RTC_INTSTS, address 0x4000_802C).

Bits	Description	
[1]	TICKIF	**RTC Time-Tick Interrupt Flag** When RTC Time-Tick Interrupt is enabled (RTC_INTEN.TICKIEN=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.ALMIEN=1), RTC unit will set AIF 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

7	6	5	4	3	2	1	0
Reserved				TWKEN	TICKSEL		

Table 5-83 RTC Time-Tick Register (RTC_TICK, address 0x4000_8030).

Bits	Description	
[3]	TWKEN	**RTC Timer Wakeup CPU Function Enable Bit** If TWKEN 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 Register** The RTC time tick period for Periodic Time-Tick Interrupt request. Time Tick (second) : 1 / (2^TICKSEL) Note: This register can be read back after the RTC is active.

5.9 Serial Peripheral Interface (SPI) 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 ISD9100 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 SPI controller can be set as a master with up to 2 slave select (SSB) address lines to access two slave devices; it also can be set as a slave controlled by an off-chip master device.

5.9.2 Features

- Supports master or slave mode operation.
- Supports one or two channels of serial data.
- 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.

5.9.3 SPI Block Diagram

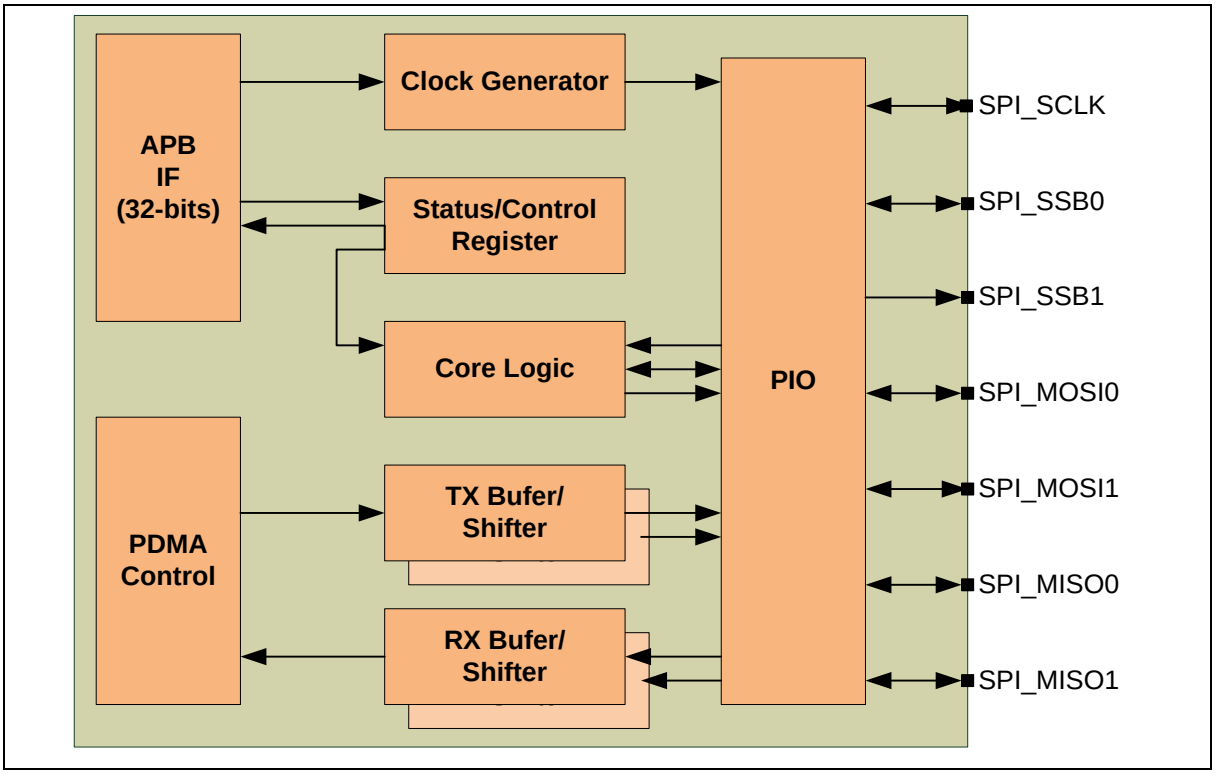


Figure 5-34 SPI Block Diagram

5.9.4 SPI Function Descriptions

Master/Slave Mode

This SPI controller can be configured as in master or slave mode by setting the SLAVE bit (SPI_CTL.SLAVE). In master mode the ISD9100 series generates SCLK and SSB signals to access one or more slave devices. In slave mode the ISD9100 series 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 of Figure 5-35 and Figure 5-36.

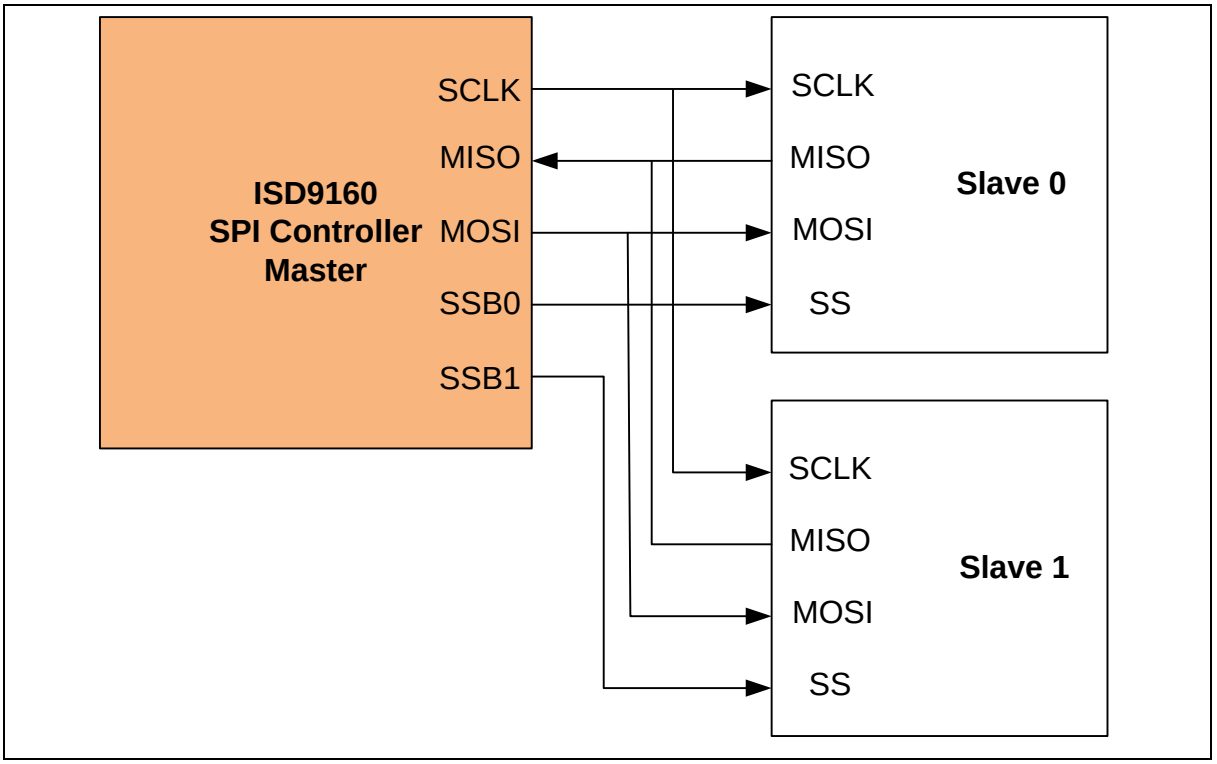


Figure 5-35 SPI Master Mode Application Block Diagram

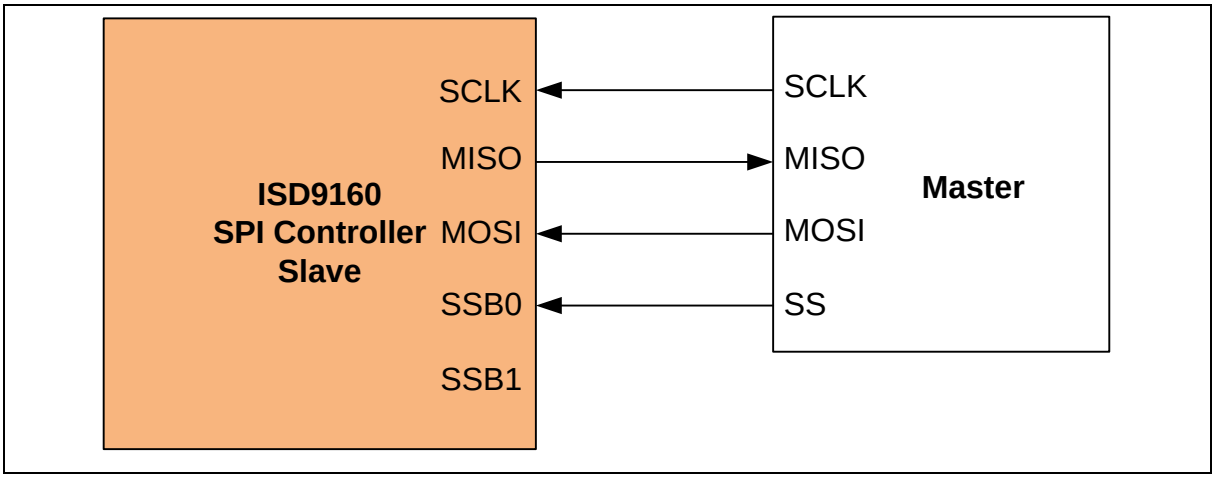


Figure 5-36 SPI Slave Mode Application Block Diagram

Slave Select

In master mode, the SPI controller can address up to two off-chip slave devices through the slave select output pins SPI_SSB0 and SPI_SSB1. Only one slave can be addressed at any one time. If more slave address lines are required, GPIO pins can be manually configured to provide additional SSB lines. In slave mode, the off-chip master device drives the slave select signal SPI_SSB0 to address the SPI controller. The slave select signal can be programmed to be active low or active high via the SPI_SSCTL.SSACTPOL bit. In addition the SPI_SSCTL.LVTRGEN 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.

Automatic Slave Select

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

Serial Clock

In master mode, writing a divisor into the SPI_CLKDIV.DIVIDER0 register will program the output frequency of serial clock to the SPI_SCLK output port. In slave mode, the off-chip master device drives the serial clock through the SPI_SCLK.

Clock Polarity

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

Transmit/Receive Bit Length

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

Burst Mode

The SPI controller has a burst mode controlled by the SPI_CTL.TXCNT field. If set to 0x01, SPI controller will burst two transactions from the SPI_TX0 and SPI_TX1 registers as shown in the waveform below:

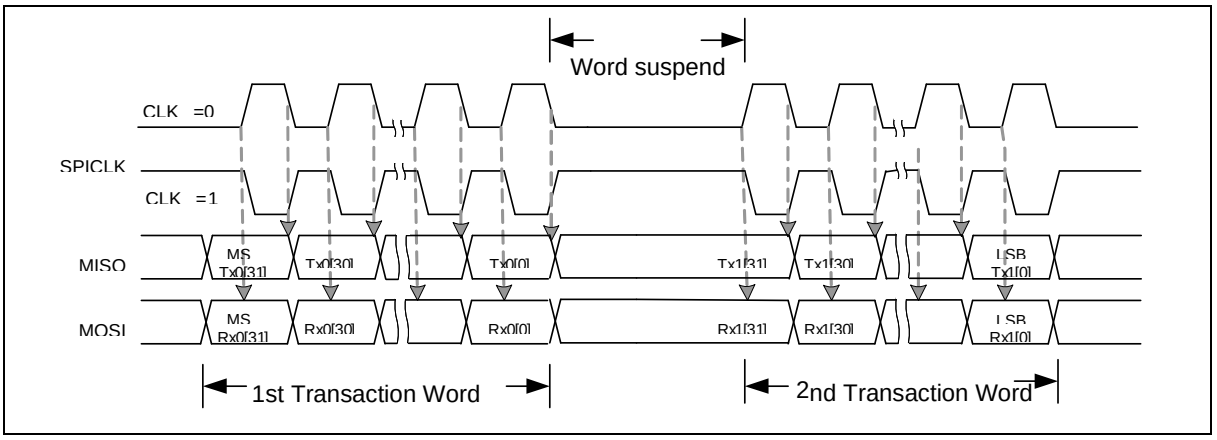


Figure 5-37 Two Transactions in One Transfer (Burst Mode)

LSB First

The SPI_CTL.LSB bit defines the **bit** order of data transmission. If LSB=0 then MSB of transfer word is sent first in time. If LSB=1 then LSB of transfer word is sent first in time.

Transmit Edge

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

Receive Edge

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

Word Sleep Suspend

The bit field SPI_CTL.SUSPITV provides a configurable suspend interval of SUSPITV+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 CLKPOL = 0. If CLKPOL = 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 SUSPITV is 0x0 (2 serial clock cycles). Word Sleep only occurs when TXCNT=1. For TXCNT=0, this parameter will determine a Byte Sleep condition if the BYTEITV bit is set.

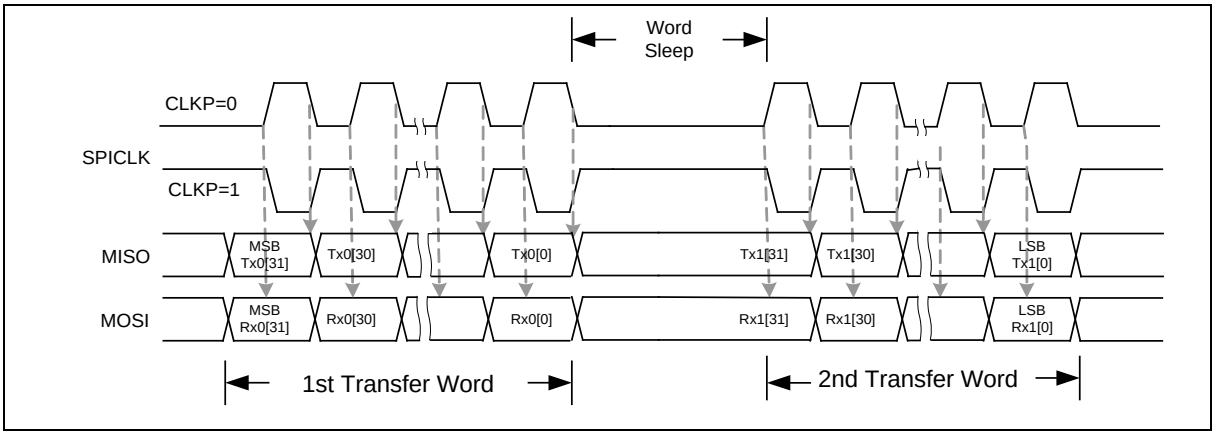


Figure 5-38 Word Sleep Suspend Mode

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 (SPI_CTL.LSB = 0) and the SPI_CTL.REORDER bit is set, the data stored in the TX buffer and RX buffer will be rearranged such that the least significant **physical byte** is processed first. For DWIDTH =0 (32 bits transfer), the sequence of transmitted/received data will be BYTE0, BYTE1, BYTE2, and then BYTE3. If DWIDTH is set to 24-bits, the sequence will be BYTE0, BYTE1, and BYTE2. The rule of 16-bits mode is the same as above, see Figure 5-39.

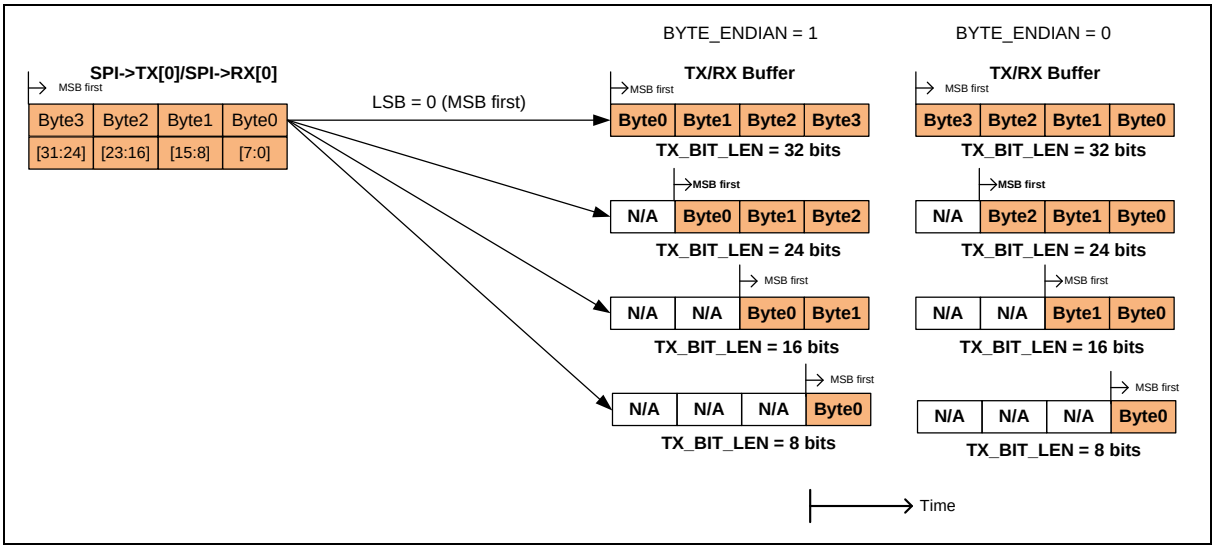


Figure 5-39 Byte Re-Ordering Transfer

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 5-40 Byte Order in Memory

It can be seen from Figure 5-40 that byte order for an array of bytes is different than that of an array of

words. Now consider if this data were to be sent to the SPI port; the user could:

1. Set DWIDTH=8 and send data byte-by-byte SPI_TX0 = ucSPI_DATA[i++]
2. Set DWIDTH=32 and send word-by-word SPI_TX0 = 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 Figure 5-41 where a int pointer points to the byte data array.

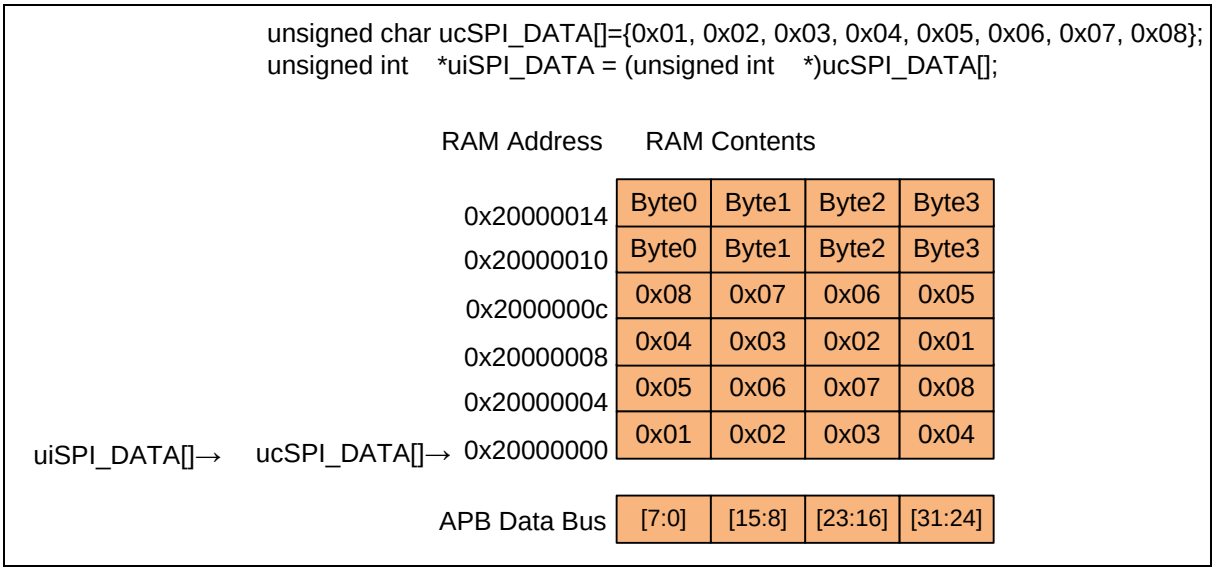


Figure 5-41 Byte Order in Memory

Now if we set DWIDTH=32 and sent word-by-word SPI_TX0 = 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}.

Byte Sleep Suspend

In master mode, if SPI_CTL.BYTEITV is set to 1, the hardware will insert a suspend interval of SPI_CTL.SUSPITV+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 DWIDTH=0x00.

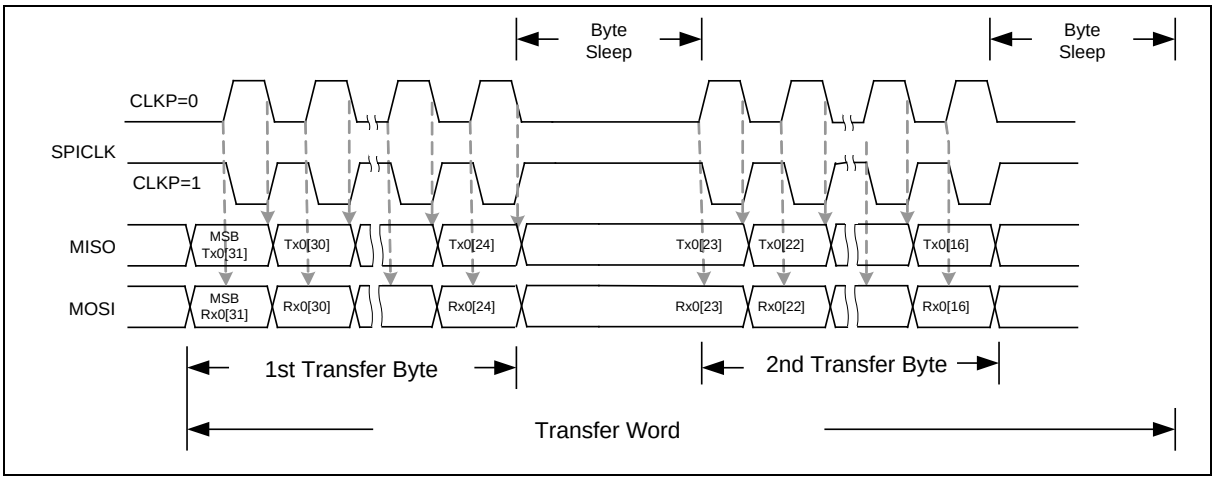


Figure 5-42 Byte Suspend Mode

Interrupt

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

FIFO Mode

The SPI controller supports a dual buffer mode when SPI_CTL.FIFOEN 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 BUSY 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 SPI_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 BUSY 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 SUSPITV parameter.

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 SPI_RX1 and SPI_TX1 buffers. Mode is enabled by setting the SPI_CTL.TWOBIT bit. This mode is only available when TXCNT=0.

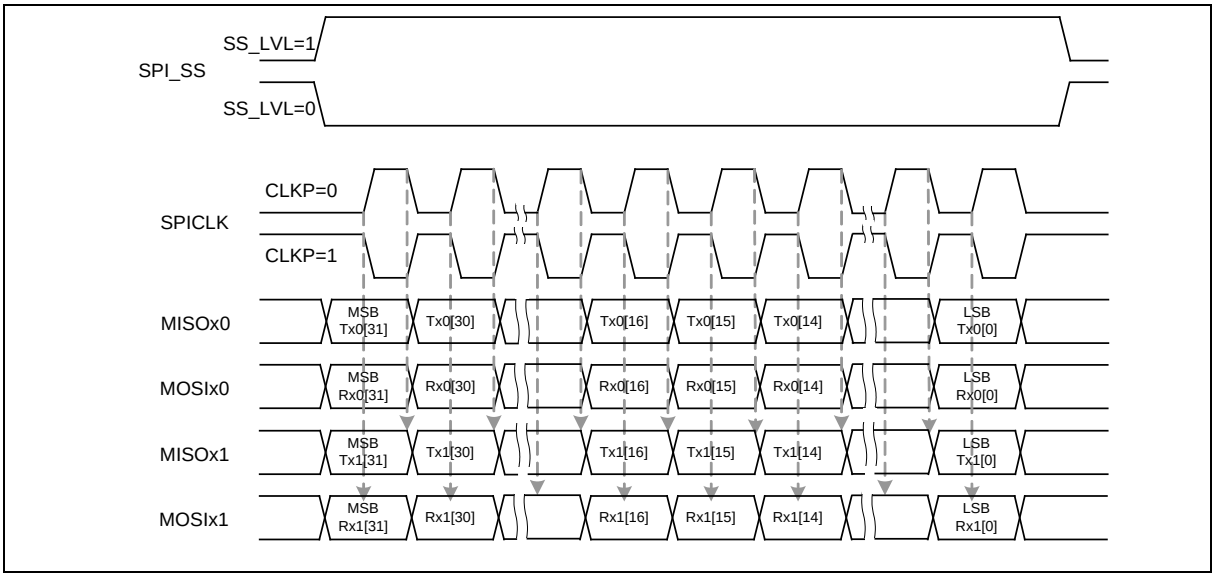


Figure 5-43 Two Bits Transfer Mode

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 SPI_CTL.VARCLKEN is enabled. The frequency pattern format is defined in SPI_VARCLK register. If the bit content of VARCLK is '0' the output period for that bit is determined by setting of SPI_CLKDIV.DIVIDER0, if the bit content of VARCLK is '1', the output period for that bit is determined by the SPI_CLKDIV.DIVIDER1 register. The following figure shows the timing relationships of serial clock (SCLK), to the VARCLK, the DIVIDER0 and the DIVIDER1 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 SPI_VARCLK and must be set 1 cycle before the next clock option. For example, if there are 5 CLK1 cycle in SPICLK, the SPI_VARCLK shall set 9 '0' in the MSB of SPI_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 DWIDTH must be programmed as 0x10 (16 bits mode only).

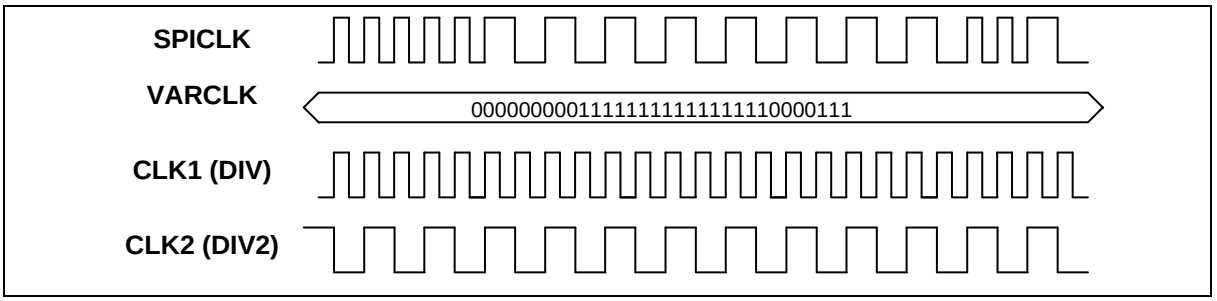
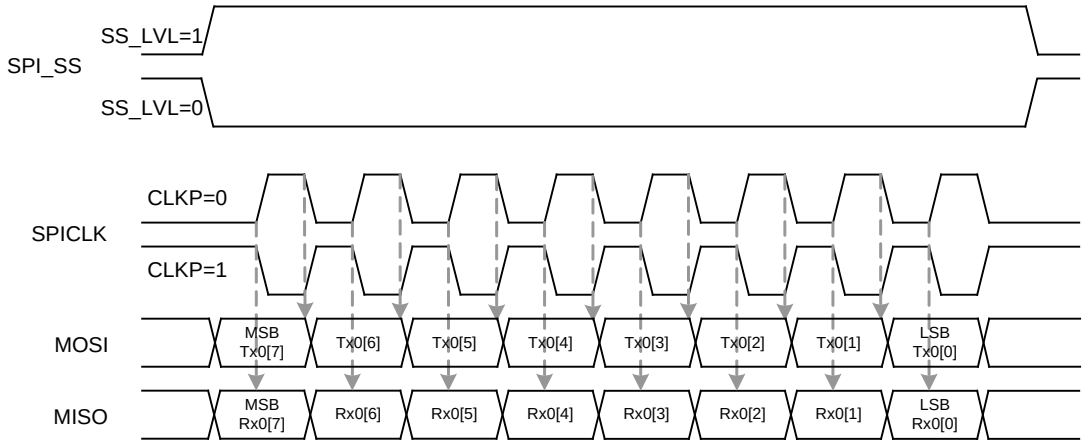


Figure 5-44 Variable Serial Clock Frequency

5.9.5 SPI Timing Diagram

In master/slave mode, the device address/slave select (SPI_SSB0/1) signal can be configured as active low or active high by the SPI_SSCTL.SSACTPOL bit. In slave mode, the SPI_SSCTL.LVTRGEN will determine whether the slave select signal is treated as a level triggered or edge triggered signal.

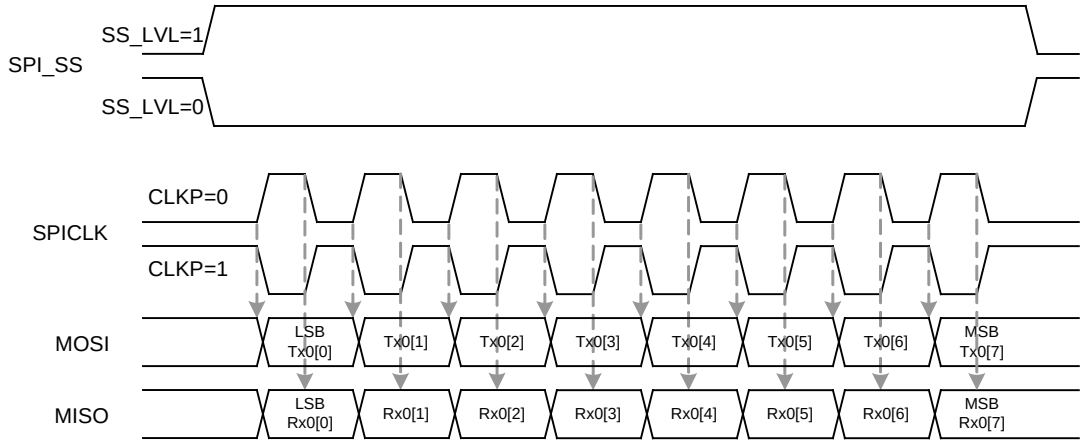
The serial clock phase and polarity is controlled by CLKPOL, RXNET and TXNEG bits. The bit length of a transfer word is configured by the DWIDTH parameter. Whether data transmission is MSB first or LSB first is controlled by the SPI_CTL.LSB bit. Four examples of SPI timing diagrams for master/slave operations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



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

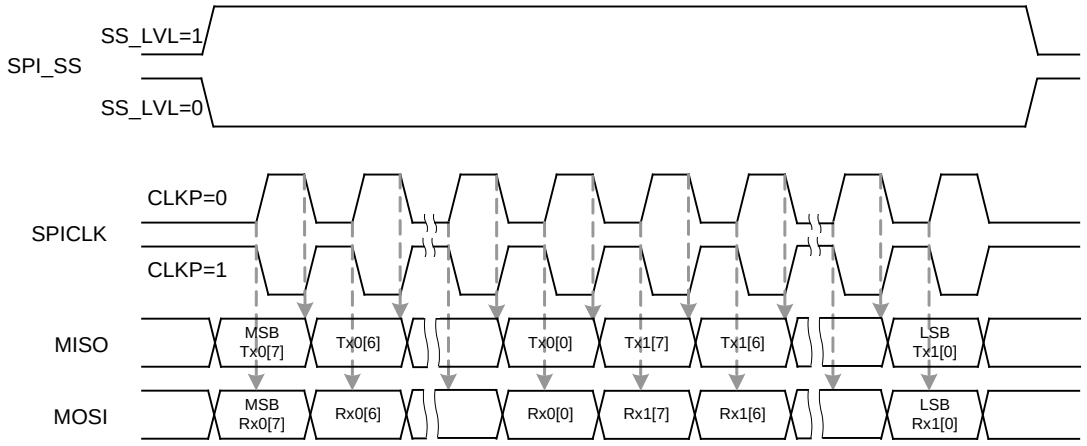
Figure 5-45 SPI Timing in Master Mode



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

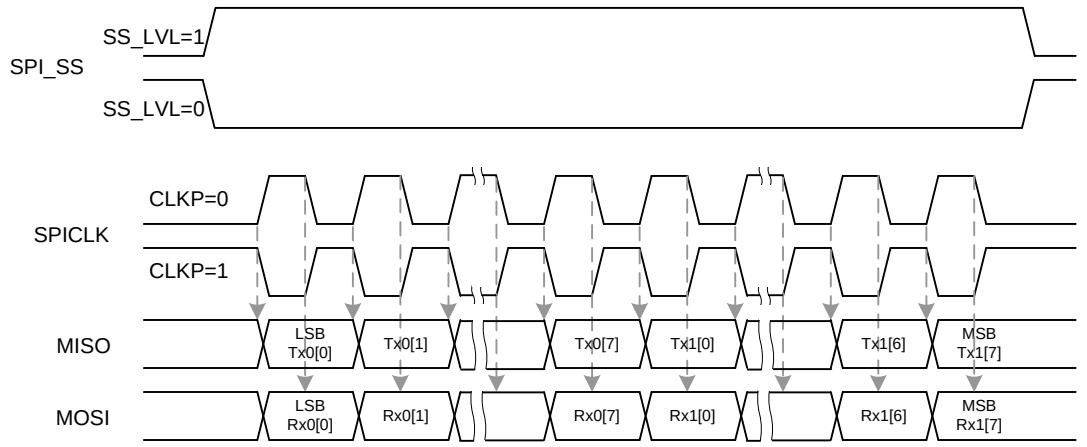
Figure 5-46 SPI Timing in Master Mode (Alternate Phase of SPICLK)



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 5-47 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 5-48 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 SPI interface to the above specifications perform the following steps:

1) Write a divisor into the SPI_CLKDIV register to determine the output frequency of serial clock. Driver function DrvSPI_SetClock(0,10000000,0) can be used to achieve this.
2) Configure the SPI_SSCTL register to address device. For example to manually address, set SPI-SPI_SSCTL.AUTOSS=0, SPI_SSCTL.SS_LVL=0 for active low SS. When software wishes to address device it will set SPI_SSCTL.SS=1 to output an active SS on SPI_SSB0 pin.
3) Configure the SPI_CTL register. Set SPI_CTL.SLAVE=0 for master mode, set SPI_CTL.CLKPOL=0 for SCLK polarity normally low, set SPI_CTL.TXNEG=1 so that data changes on falling edge of SCLK, set SPI_CTL.RXNET=0 so that data is latched into device on positive edge of SCLK, set SPI_CTL.DWIDTH=8 and SPI_CTL.TXCNT=0 for a single byte transfer and finally set SPI_CTL.LSB=0 for MSB first transfer.
4) If manually selecting slave device set SPI_SSCTL.SS=1.
5) To transmit one byte of data, write data to SPI_TX0 register. If only doing a receive, write a dummy byte to SPI_TX0 register.
6) Enable the SPI_CTL.BUSY 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.UNITIEN bit is set) or by polling the BUSY bit which will be cleared to 0 by hardware automatically at end of transmission. --

7) Read out the received one byte data from SPI_RX0
8) Go to 5) to continue another data transfer or set SPI_SSCTL.SS=0 to deactivate the off-chip slave devices.

- Example 2, SPI controller is set as a slave device that controlled by an off-chip master device with the following characteristics:
  - Data bit latched on positive edge of serial clock
  - Data bit driven on negative edge of serial clock
  - Data be transferred from LSB first
  - SCLK high in idle state
  - Only one byte data be transmitted/received in a transfer
  - Slave select signal is active high level trigger

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

1) Configure the SPI_SSCTL register. SPI_SSCTL.SSACTPOL=1 for active high slave select, SPI_SSCTL.LVTRGEN=1 for level sensitive trigger.

2) Configure the SPI_CTL register. Set SPI_CTL.SLAVE=1 for slave mode, set SPI_CTL.CLKPOL=1 for SCLK polarity idle high, set SPI_CTL.TXNEG=1 so that data changes on falling edge of SCLK, set SPI_CTL.RXNET=0 so that data is latched into device on positive edge of SCLK, set SPI_CTL.DWIDTH=8 and SPI_CTL.TXCNT=0 for a single byte transfer and finally set SPI_CTL.LSB=1 for LSB first transfer.

3) If SPI slave is to transmit one byte of data to the off-chip master device, write first byte to SPI_TX0 register. If no data to be transmitted write a dummy byte.

4) Enable the BUSY 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 SPI_CTL.UNITIEN bit is set) or by polling the BUSY bit which will be cleared to 0 by hardware automatically at end of transmission. --

5) Read out the received data from SPI_RX0 register.

6) Go to 3) to continue another data transfer or disable the BUSY bit to stop data transfer.

5.9.7 SPI Serial Interface Control Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI0 Base Address: **SPI0_BA = 0x4003_0000**				
SPI_CTL	SPI0_BA + 0x00	R/W	Control and Status Register	0x0000_0004
SPI_CLKDIV	SPI0_BA + 0x04	R/W	Clock Divider Register (Master Only)	0x0000_0000
SPI_SSCTL	SPI0_BA + 0x08	R/W	Slave Select Register	0x0000_0000
SPI_RX0	SPI0_BA + 0x10	R	Data Receive Register 0	0x0000_0000
SPI_RX1	SPI0_BA + 0x14	R	Data Receive Register 1	0x0000_0000
SPI_TX0	SPI0_BA + 0x20	W	Data Transmit Register 0	0x0000_0000
SPI_TX1	SPI0_BA + 0x24	W	Data Transmit Register 1	0x0000_0000
SPI_VARCLK	SPI0_BA + 0x34	R/W	Variable Clock Pattern Register	0x007F_FF87
SPI_PDMACTL	SPI0_BA + 0x38	R/W	SPI DMA Control Register	0x0000_0000

NOTE 1: When software programs SPI_CTL, the BUSY bit should be written last.

5.9.8 SPI Control Register Description

SPI Control and Status Register (SPI_CTL)

Register	Offset	R/W	Description	Reset Value
SPI_CTL	SPI0_BA + 0x00	R/W	Control and Status Register	0x0000_0004

31	30	29	28	27	26	25	24
Reserved			PDMASSEN	TXFULL	TXEMPTY	RXFULL	RXEMPTY
23	22	21	20	19	18	17	16
VARCLKEN	TWOBIT	FIFOEN	REORDER	BYTEITV	SLAVE	UNITIEN	UNITIF
15	14	13	12	11	10	9	8
SUSPITV				CLKPOL	LSB	TXNUM	
7	6	5	4	3	2	1	0
DWIDTH					TXNEG	RXNET	GOBUSY

Table 5-84 SPI Control and Status Register (SPI_CTL, address 0x4003_0000)

Bits	Description	
[31:28]	Reserved	Reserved
[28]	PDMASSEN	**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** 0 = The transmit data FIFO is not full. 1 = The transmit data FIFO is full.
[26]	TXEMPTY	**Transmit FIFO Empty Status** 0 = The transmit data FIFO is not empty. 1 = The transmit data FIFO is empty.
[25]	RXFULL	**Receive FIFO Full Status** 0 = The receive data FIFO is not full. 1 = The receive data FIFO is full.
[24]	RXEMPTY	**Receive FIFO Empty Status** 0 = The receive data FIFO is not empty. 1 = The receive data FIFO is empty.
[23]	VARCLKEN	**Variable Clock Enable (Master Only)** 0 = The serial clock output frequency is fixed and determined only by the value of DIVIDER0. 1 = SCLK output frequency is variable. The output frequency is determined by the value of SPI_VARCLK, DIVIDER0, and DIVIDER1. Note that when enabled, the setting of DWIDTH must be programmed as 0x10 (16 bits mode)

[22]	**TWOBIT**	**Two Bits Transfer Mode** 0 = Disable two-bit transfer mode. 1 = Enable two-bit transfer mode. Note that when enabled in master mode, MOSI0 data comes from SPI_TX0 and MOSI1 data from SPI_TX1. Likewise SPI_RX0 receives bit stream from MISO0 and SPI_RX1 from MISO1. Note that when enabled, the setting of TXCNT must be programmed as 0x00
[21]	**FIFOEN**	**FIFO Mode** 0 = No FIFO present on transmit and receive buffer. 1 = Enable FIFO on transmit and receive buffer.
[20]	**REORDER**	**Byte Endian Reorder Function** This function changes the order of bytes sent/received to be least significant physical byte first.
[19]	**BYTEITV**	**Insert Sleep Interval Between Bytes** This function is only valid for 32bit transfers (DWIDTH aaa 0). If set then a pause of (SUSPITV+2) SCLK cycles is inserted between each byte transmitted.
[18]	**SLAVE**	**Master Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**UNITIEN**	**Interrupt Enable** 0 = Disable SPI Interrupt. 1 = Enable SPI Interrupt to CPU.
[16]	**UNITIF**	**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]	**SUSPITV**	**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 CLKPOL aaa 0. If CLKPOL aaa 1, the interval is from the rising clock edge to the falling clock edge. The default value is 0x0. When TXCNT aaa 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: (SUSPITV[3:0] + 2) * period of SCLK
[11]	**CLKPOL**	**Clock Polarity** 0 = SCLK idle low. 1 = SCLK idle high.

[10]	**LSB**	**LSB First** 0 = The MSB is transmitted/received first (which bit in SPI_TX0/1 and SPI_RX0/1 register that is depends on the DWIDTH field). 1 = The LSB is sent first on the line (bit 0 of SPI_TX0/1), and the first bit received from the line will be put in the LSB position in the Rx register (bit 0 of SPI_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]	**DWIDTH**	**Transmit Bit Length** This field specifies how many bits are transmitted in one transmit/receive. Up to 32 bits can be transmitted. DWIDTH aaa 0x01 --- 1 bit DWIDTH aaa 0x02 --- 2 bits ---- DWIDTH aaa 0x1f --- 31 bits DWIDTH aaa 0x00 --- 32 bits
[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]	**RXNET**	**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]	**GOBUSY**	**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 BUSY bit. When a transfer is in progress, writing to any register of the SPI master/slave core has no effect.

SPI Divider Register (SPI_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SPI_CLKDIV	SPI0_BA + 0x04	R/W	Clock Divider Register (Master Only)	0x0000_0000

31	30	29	28	27	26	25	24
DIVIDER1[15:8]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DIVIDER1[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DIVIDER0[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER0[7:0]							

Table 5-85 SPI Clock Divider Register (SPI_CLKDIV, address 0x4003_0004)

Bits	Description	
[31:16]	**DIVIDER1**	**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: Fsclk aaa Fpclk / ((DIVIDER1+1) * 2)
[15:0]	**DIVIDER0**	**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: Fsclk aaa Fpclk / ((DIVIDER0+1) * 2) In slave mode, the period of SPI clock driven by a master shall satisfy Fsclk < aaa (Fpclk / 5) In other words, the maximum frequency of SCLK clock is one fifth of the SPI peripheral clock.

SPI Slave Select Register (SPI_SSCTL)

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

7	6	5	4	3	2	1	0
Reserved		LVTRGSTS	LVTRGEN	AUTOSS	SSACTPOL	SS[1:0]	

Table 5-86 SPI Slave Select Register (SPI_SSCTL, address 0x4003_0008)

Bits	Description	
[31:6]	Reserved	Reserved
[5]	LVTRGSTS	**Level Trigger Flag** When the LVTRGEN bit is set in slave mode, this bit can be read to indicate the received bit number is met the requirement or not. 0=One of the received number and the received bit length doesn't meet the requirement in one transfer. 1=The received number and received bits met the requirement which defines in TXCNT and DWIDTH among one transfer. Note: This bit is READ only
[4]	LVTRGEN	**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 SSACTPOL to decide the signal is active low or active high.
[3]	AUTOSS	**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 SPI_SSCTL[1:0] register. 1 = If this bit is set, SPISSx0/1 signals are generated automatically. It means that device/slave select signal, which is set in SPI_SSCTL[1:0] register is asserted by the SPI controller when transmit/receive is started by setting BUSY, and is de-asserted after each transmit/receive is finished.
[2]	SSACTPOL	**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.
[1:0]	SS	**Slave Select Register (Master only)** If AUTOSS 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 AUTOSS bit is set, writing 1 to any bit location of this field will select appropriate SPISSx0/1 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 level of SPISSx0/1 is specified in SSACTPOL). **Note:** SPISSx0 is always defined as device/slave select input signal in slave mode.

SPI Data Receive Register (RX)

Register	Offset	R/W	Description	Reset Value
SPI_RX0	SPI0_BA + 0x10	R	Data Receive Register 0	0x0000_0000
SPI_RX1	SPI0_BA + 0x14	R	Data Receive Register 1	0x0000_0000

Table 5-87 SPI Data Receive Register (SPI_RX0/SPI_RX1, address 0x4003_0010/0x4003_0014)

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 SPI_CTL register. For example, if DWIDTH is set to 0x08 and TXCNT is set to 0x0, bit Rx0[7:0] holds the received data. **NOTE:** The Data Receive Registers are read only registers.

SPI Data Transmit Register (TX)

Register	Offset	R/W	Description	Reset Value
SPI_TX0	SPI0_BA + 0x20	W	Data Transmit Register 0	0x0000_0000
SPI_TX1	SPI0_BA + 0x24	W	Data Transmit Register 1	0x0000_0000

Table 5-88 SPI Data Transmit Register (SPI_TX0/SPI_TX1, address 0x4003_0020/0x4003_0024)

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 SPI_CTL register. For example, if DWIDTH is set to 0x08 and the TXCNT is set to 0x0, the bit TX0[7:0] will be transmitted in next transfer. If DWIDTH is set to 0x00 and TXCNT is set to 0x1, the core will perform two 32-bit transmit/receive successive using the same setting (the order is TX0[31:0], TX1[31:0]).

SPI Variable Clock Pattern Flag Register (SPI_VARCLK)

Register	Offset	R/W	Description	Reset Value
SPI_VARCLK	SPI0_BA + 0x34	R/W	Variable Clock Pattern Register	0x007F_FF87

Table 5-89 SPI Variable Clock Pattern Register (SPI_VARCLK, address 0x4003_0034)

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 DIVIDER0. If the bit field of VARCLK is '1', the output frequency of SCLK is given by the value of DIVIDER1. Refer to register DIVIDER0. Refer to Variable Serial Clock Frequency paragraph for detailed description. Note: Used for CLKPOL = 0 only, 16 bit transmission.

DMA Control Register (SPI_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPI_PDMACTL	SPI0_BA + 0x38	R/W	SPI DMA Control Register	0x0000_0000

7	6	5	4	3	2	1	0
Reserved						RXPDMAEN	TXPDMAEN

Table 5-90 SPI DMA Control Register (SPI_PDMACTL, address 0x4003_0038)

Bits	Description	
[1]	RXPDMAEN	**Receive DMA Start** Set this bit to 1 will start the receive DMA process. SPI module will issue request to DMA module automatically.
[0]	TXPDMAEN	**Transmit DMA Start** 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 BUSY bit of SPI_CTL register. The DMA controller inside SPI module will set it automatically whenever necessary.

5.10 Timer Controller

5.10.1 General Timer Controller

The ISD9100 series 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.

5.10.2 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).

5.10.3 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 Figure 5-49 for the timer controller block diagram. There are five options of clock source for each channel, Figure 5-50 illustrate the clock source control function.

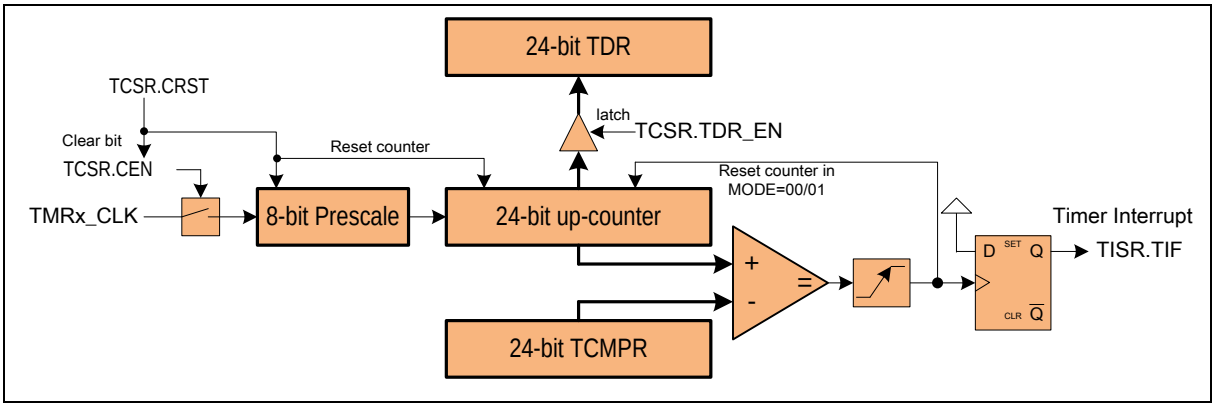


Figure 5-49 Timer Controller Block Diagram

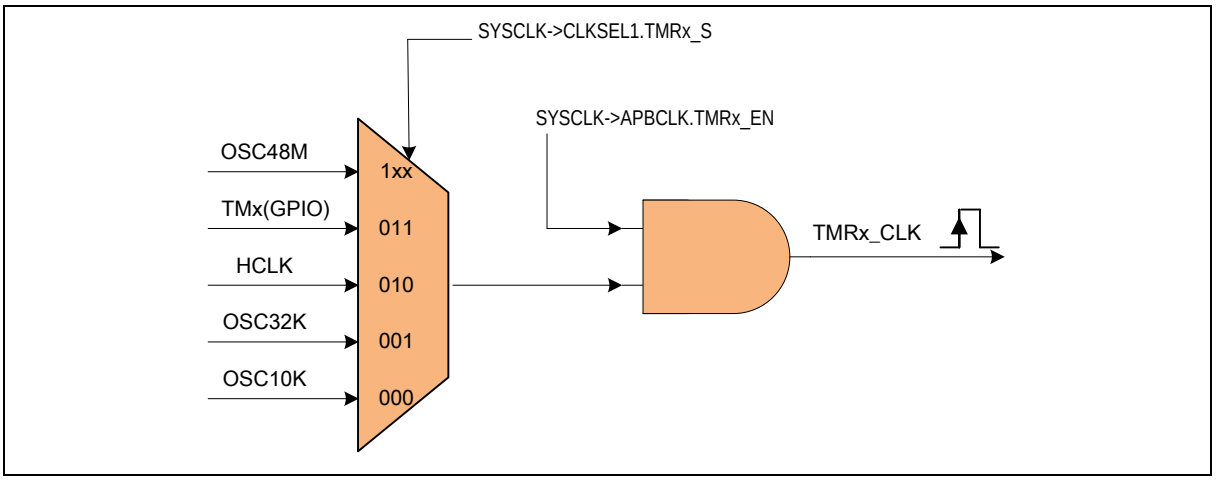


Figure 5-50 Clock Source of Timer Controller

5.10.4 Timer Controller Register Map

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

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

Timer Control Register (TIMERx_CTL)

Register	Offset	R/W	Description	Reset Value
TIMERx_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[1:0]		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[7:0]							

Table 5-91 Timer Control and Status Register (TIMERx_CTL, address 0x4001_0000 + *x* *0x20).

Bits	Description	
[31]	Reserved	Reserved
[30]	CNTEN	**Counter Enable Bit** 0 = Stops/Suspends counting 1 = Starts counting Note1: Setting CNTEN aaa 1 enables 24-bit counter. It continues count from last value. Note2: This bit is auto-cleared by hardware in one-shot mode (OPMODE aaa 00b) when the timer interrupt is generated (INTEN aaa 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 = CMPDAT (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, prescale and also force CNTEN to 0. 0 = No effect. 1 = Reset Timer's prescale 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 aaa 0, then there is no scaling.

Timer Compare Register (TIMERx_CMP)

Register	Offset	R/W	Description	Reset Value
TIMERx_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[23:16]							
15	14	13	12	11	10	9	8
CMPDAT [15:8]							
7	6	5	4	3	2	1	0
CMPDAT[7:0]							

Table 5-92 Timer Compare Register (TIMERx_CMP, address 0x4001_0004 + *x* * 0x20)

Bits	Description	
[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 aaa 1. The CMPDAT value defines the timer cycle time. Time out period aaa (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 CNTEN 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 (TIMERx_INTSTS)

Register	Offset	R/W	Description	Reset Value
TIMERx_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**

Table 5-93 Timer Interrupt Status Register (TIMERx_INTSTS, address 0x4001_0008 + x * 0x20)

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 (TIMERx_CNT)

Register	Offset	R/W	Description	Reset Value
TIMERx_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[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[7:0]							

Table 5-94 Timer Data Register (TIMERx_CNT, address 0x4001_000C + x *0x20).

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.

5.11 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 5-95 Watchdog Timeout Interval Selection

TOUTSEL	Interrupt Timeout	Watchdog Reset Timeout	RSTCNT Timeout Interval (WDT_CLK=49.152 MHz)	RSTCNT Timeout Interval (WDT_CLK=32kHz)
000	2^4 WDT_CLK	(2^4 + 1024) WDT_CLK	21.2 us	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

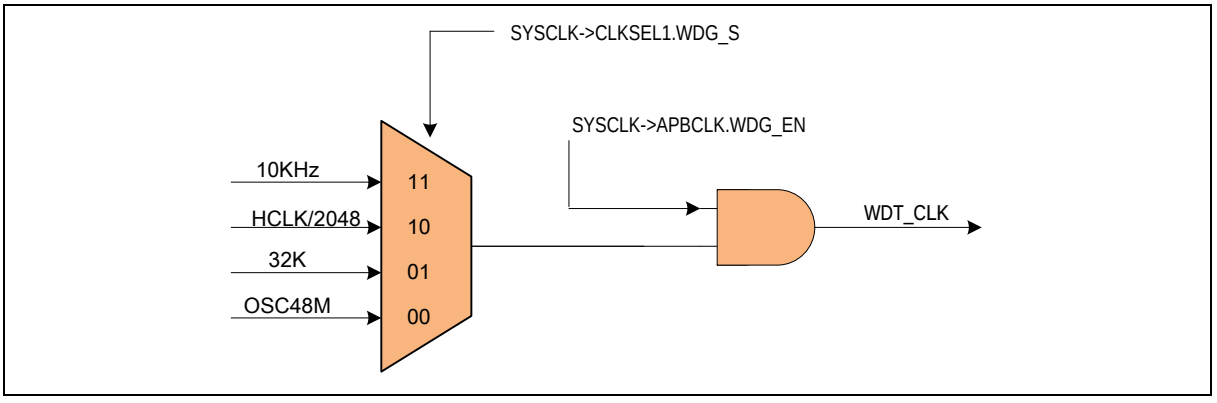


Figure 5-51 Watchdog Timer Clock Control

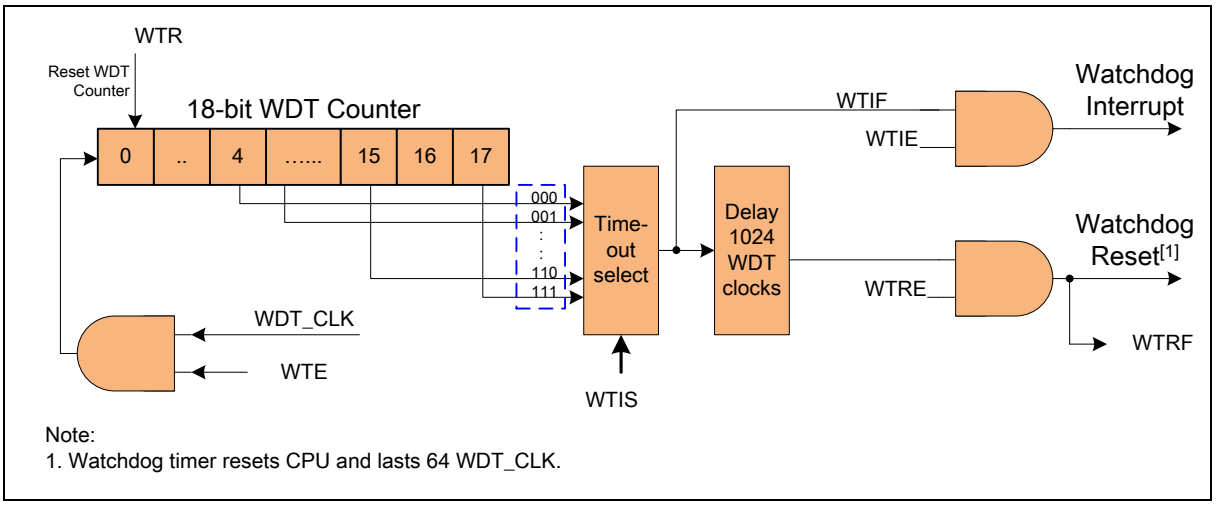


Figure 5-52 Watchdog Timer Block Diagram

5.11.1 Watchdog Timer Control Registers 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

Watchdog Timer Control Register (WDT_CTL)

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

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

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

Bits	Description	
[31:11]	Reserved	Reserved
[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 aaa 2^(2xWTIS+4) x WDT_CLK Reset Timeout aaa (2^(2xWTIS+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
[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

5.12 UART Interface Controller

The ISD9100 series 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).

5.12.1 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 (BAUD). Table 5-96 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 5-96 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 5-97 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

5.12.2 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.

5.12.3 Block Diagram

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

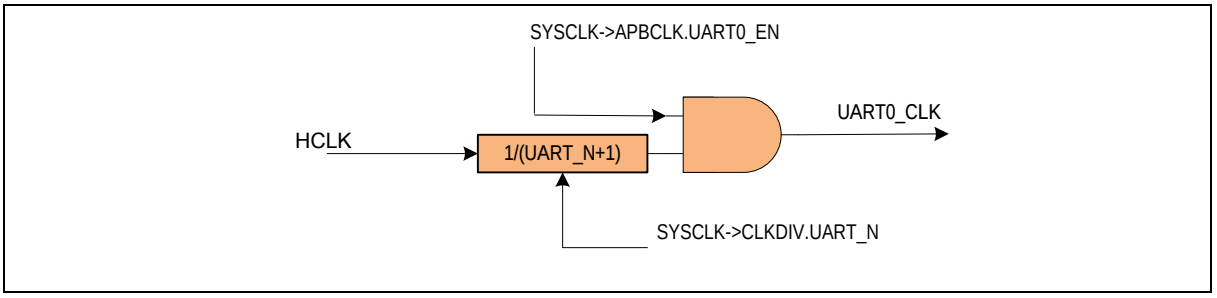


Figure 5-53 UART Clock Control Diagram

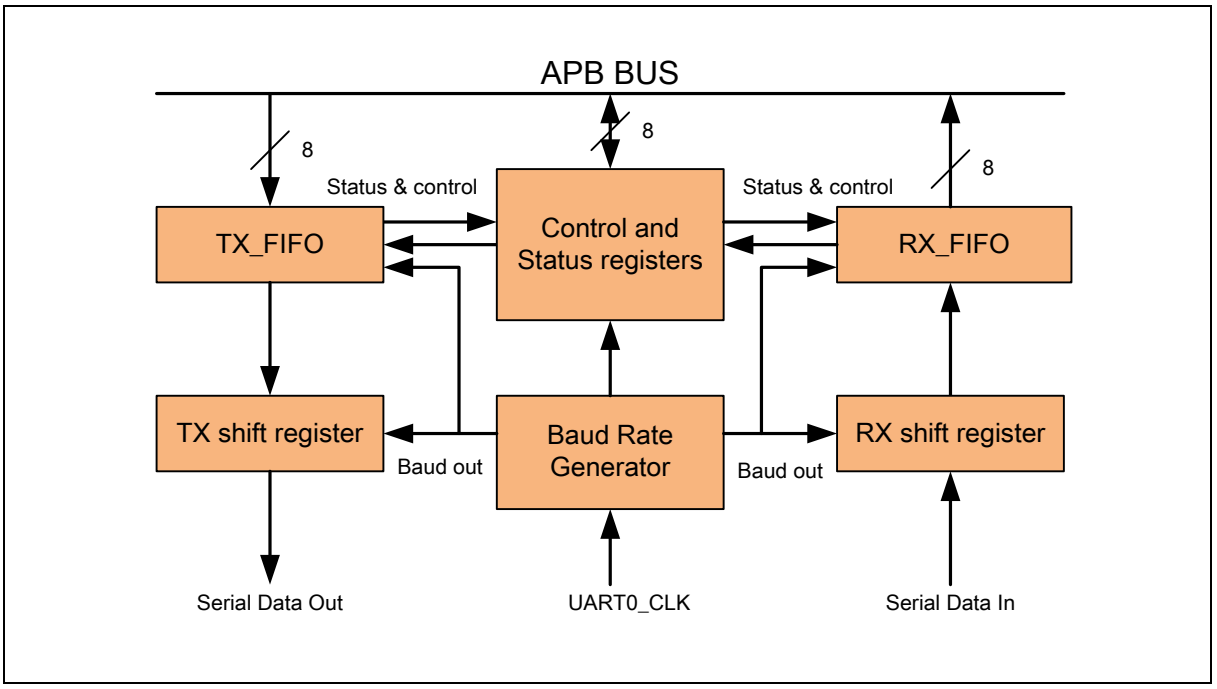


Figure 5-54 UART Block Diagram

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 UART0_CLK clock by the divisor to get the desired baud rate clock. Refer to Table 5-96 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 5-55 demonstrates the auto-flow control block diagram.

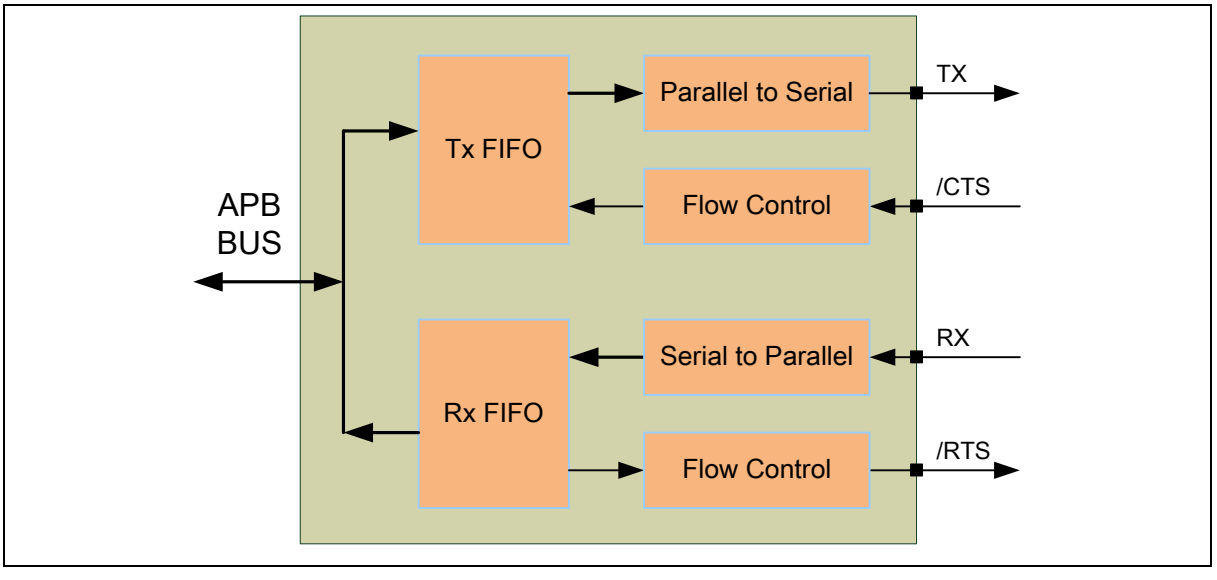


Figure 5-55 Auto Flow Control Block Diagram

5.12.4 IrDA Mode

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

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.

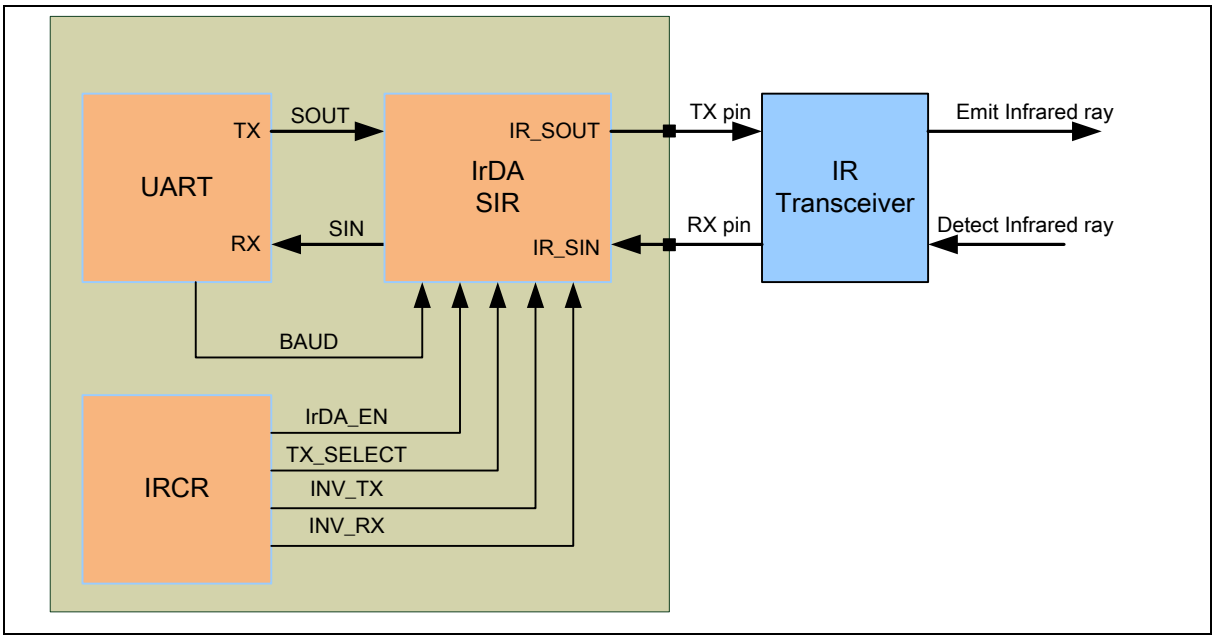


Figure 5-56 IrDA Block Diagram

5.12.4.1 *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 3/16 the period of the baud rate.

5.12.4.2 *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.

5.12.4.3 *IrDA SIR Operation*

The IrDA SIR Encoder/decoder provides functionality which converts between UART data stream and half duplex serial SIR interface. Figure 5-57 shows the IrDA encoder/decoder waveform:

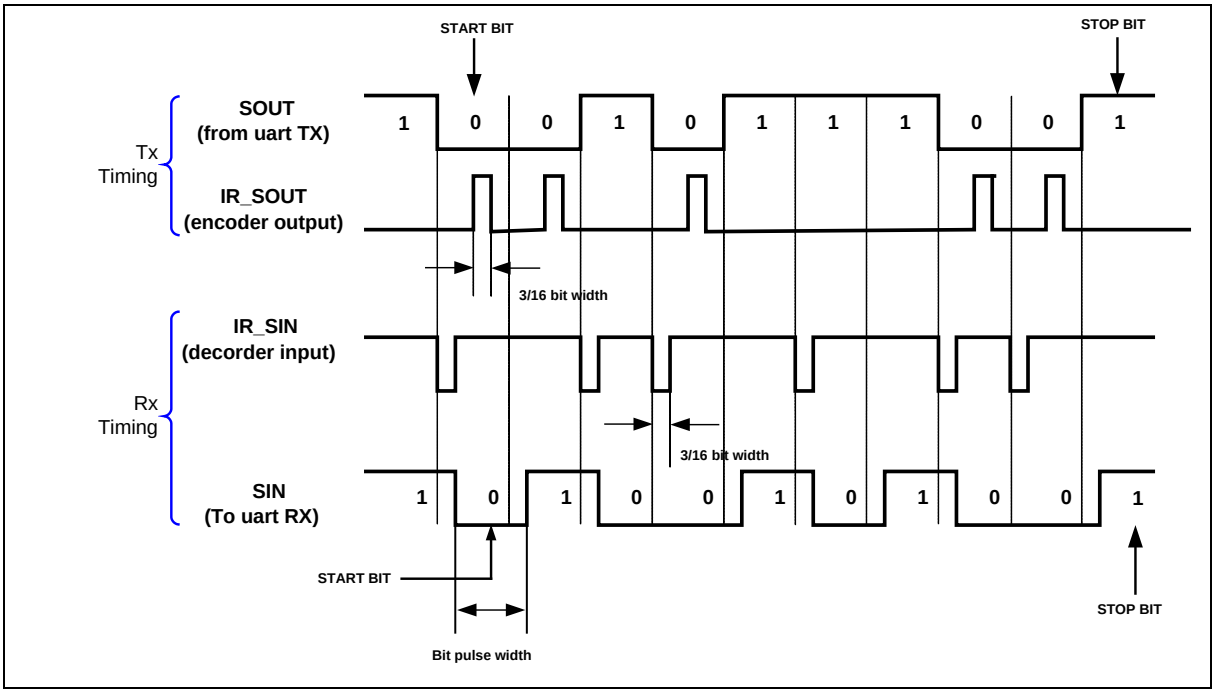


Figure 5-57 IrDA Tx/Rx Timing Diagram

5.12.5 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/).

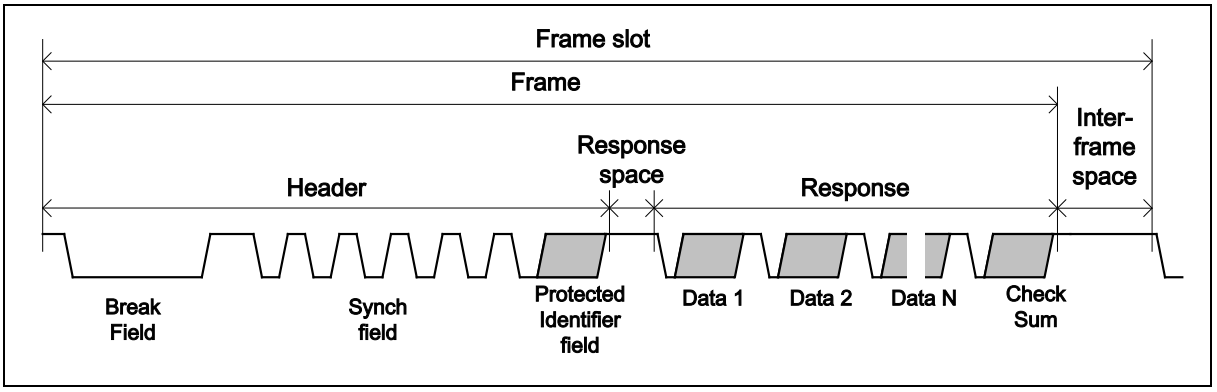


Figure 5-58 Structure of LIN Frame

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.LINTX_EN 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.LINRX_EN 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.

5.12.6 UART Interface Control Register Map

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

Register	Offset	R/W	Description	Reset Value
UART0 Base Address: **UART0_BA = 0x4005_0000**				
UART_DAT	UART0_BA + 0x00	R/W	UART0 Receive/Transfer FIFO Register.	Undefined
UART_INTEN	UART0_BA + 0x04	R/W	UART0 Interrupt Enable Register.	0x0000_0000
UART_FIFO	UART0_BA + 0x08	R/W	UART0 FIFO Control Register.	0x0000_0000
UART_LINE	UART0_BA + 0x0C	R/W	UART0 Line Control Register.	0x0000_0000
UART_MODEM	UART0_BA + 0x10	R/W	UART0 Modem Control Register.	0x0000_0000
UART_MODEMSTS	UART0_BA + 0x14	R/W	UART0 Modem Status Register.	0x0000_0000
UART_FIFOSTS	UART0_BA + 0x18	R/W	UART0 FIFO Status Register.	0x1040_4000
UART_INTSTS	UART0_BA + 0x1C	R/W	UART0 Interrupt Status Register.	0x0000_0002
UART_TOUT	UART0_BA + 0x20	R/W	UART0 Time Out Register	0x0000_0000
UART_BAUD	UART0_BA + 0x24	R/W	UART0 Baud Rate Divisor Register	0x0F00_0000
UART_IRDA	UART0_BA + 0x28	R/W	UART0 IrDA Control Register.	0x0000_0040
UART_ALTCTL	UART0_BA + 0x2C	R/W	UART0 LIN Control Register.	0x0000_0000
UART_FUNCSEL	UART0_BA + 0x30	R/W	UART0 Function Select Register.	0x0000_0000

5.12.7 UART Interface Control Register Description

Receive FIFO Data Register (UART_DAT)

Register	Offset	R/W	Description	Reset Value
UART_DAT	UART0_BA + 0x00	R/W	UART0 Receive/Transfer FIFO Register.	Undefined

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							

Table 5-98 UART Receive FIFO Data Register (UART_DAT, address 0x4005_0000)

Bits	Description	
[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 (UART_INTEN)

Register	Offset	R/W	Description	Reset Value
UART_INTEN	UART0_BA + 0x04	R/W	UART0 Interrupt Enable Register.	0x0000_0000

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

Table 5-99 UART Interrupt Enable Register (UART_INTEN, address 0x4005_0004)

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 (UART_FIFO)

Register	Offset	R/W	Description	Reset Value
UART_FIFO	UART0_BA + 0x08	R/W	UART0 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

Table 5-100 UART FIFO Control Register (UART_FIFO, address 0x4005_0008)

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
[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 2 : 8
[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 transmitting 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 (UART_LINE)

Register	Offset	R/W	Description	Reset Value
UART_LINE	UART0_BA + 0x0C	R/W	UART0 Line Control Register.	0x0000_0000

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

Table 5-101 UART Line Control Register (UART_LINE, address 0x4005_000C)

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 (UART_MODEM)

Register	Offset	R/W	Description	Reset Value
UART_MODEM	UART0_BA + 0x10	R/W	UART0 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

Table 5-102 UART Modem Control Register (UART_MODEM, address 0x4005_0010)

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 aaa 0, this bit controls whether RTS pin is active or not. 0 = Drive RTS inactive (aaa ~RTSACTLV). 1 = Drive RTS active (aaa RTSACTLV).

Modem Status Register (UART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UART_MODEMSTS	UART0_BA + 0x14	R/W	UART0 Modem Status Register.	0x0000_0000

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

Table 5-103 UART Modem Status Register (UART_MODEMSTS, address 0x4005_0014)

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 aaa 1 **NOTE:** This bit is cleared by writing 1 to itself.

FIFO Status Register (UART_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
UART_FIFOSTS	UART0_BA + 0x18	R/W	UART0 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

Table 5-104 UART FIFO Status Register (UART_FIFOSTS, address 0x4005_0018)

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 (UART_INTSTS)

Register	Offset	R/W	Description	Reset Value
UART_INTSTS	UART0_BA + 0x1C	R/W	UART0 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	BUF_ERR _INT	RXTOINT	MODEMINT	RLSINT	THERINT	RDAINT
7	6	5	4	3	2	1	0
LINIF	Reserved	BUFERRIF	RXTOIF	MODENIF	RLSIF	THREIF	RDAIF

Table 5-105 UART Interrupt Status Register (UART_INTSTS, address 0x4005_001C)

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 Table 5-106.

Table 5-106 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	**DMA_BUFERRIF** = (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 5-107 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 BUFERRINT	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 (UART_TOUT)

Register	Offset	R/W	Description	Reset Value
UART_TOUT	UART0_BA + 0x20	R/W	UART0 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**						

Table 5-108 UART Time Out Register (UART_TOUT, address 0x4005_0020)

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 (TOUT_CNT) 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 (UART_BAUD)

Register	Offset	R/W	Description	Reset Value
UART_BAUD	UART0_BA + 0x24	R/W	UART0 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 5-110.

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[15:0]							
7	6	5	4	3	2	1	0
BRD[7:0]							

Table 5-109 UART Baud Rate Divider Register (UART_BAUD, address 0x4005_0024)

Bits	Description	
[31:30]	Reserved	Reserved
[29]	**BAUDM1**	**Divider X Enable** The baud rate equation is: Baud Rate aaa UART_CLK / [M * (BRD + 2)] ; The default value of M is 16. 0 = Disable divider X (M aaa 16) 1 = Enable divider X (M aaa EDIVM1+1, with EDIVM1 ≥ 8). Refer to Table 5-110 for more information. NOTE: When in IrDA mode, this bit must disabled.
[28]	**BAUDM0**	**Divider X equal 1** 0: M aaa EDIVM1+1, with restriction EDIVM1 ≥ 8. 1: M aaa 1, with restriction BRD[15:0] ≥ 3. Refer to Table 5-110 for more information.
[27:24]	**EDIVM1**	**Divider X** The baud rate divider M aaa EDIVM1+1.
[23:16]	Reserved	Reserved
[15:0]	**BRD**	**Baud Rate Divider** Refer to Table 5-110 for more information.

Table 5-110 Baud Rate Equations.

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)] , requires B ≥ 8
2	1	1	Don't care	A	UART_CLK / (A+2), requires A ≥ 3

IrDA Control Register (UART_IRDA)

Register	Offset	R/W	Description	Reset Value
UART_IRDA	UART0_BA + 0x28	R/W	UART0 IrDA Control Register.	0x0000_0040

7	6	5	4	3	2	1	0
Reserved	**RXINV**	**TXINV**	**Reserved**		**LOOPBACK**	**TXEN**	**Reserved**

Table 5-111 UART IrDA Control Register (UART_IRDA, address 0x4005_0028)

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 (UART_ALTCTL)

Register	Offset	R/W	Description	Reset Value
UART_ALTCTL	UART0_BA + 0x2C	R/W	UART0 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**			

Table 5-112 UART LIN Network Control Register (UART_ALTCTL, address 0x4005_002C)

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.
[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 (UART_FUNCSEL)

Register	Offset	R/W	Description	Reset Value
UART_FUNCSEL	UART0_BA + 0x30	R/W	UART0 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**

Table 5-113 UART Function Select Register (UART_FUNCSEL, address 0x4005_0030)

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.

5.13 I2S Audio PCM Controller

5.13.1 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.

5.13.2 Features

- I2S can operate as either master or slave
- Master clock generation for slave device synchronization.
- Capable of handling 8, 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.

5.13.3 I2S Block Diagram

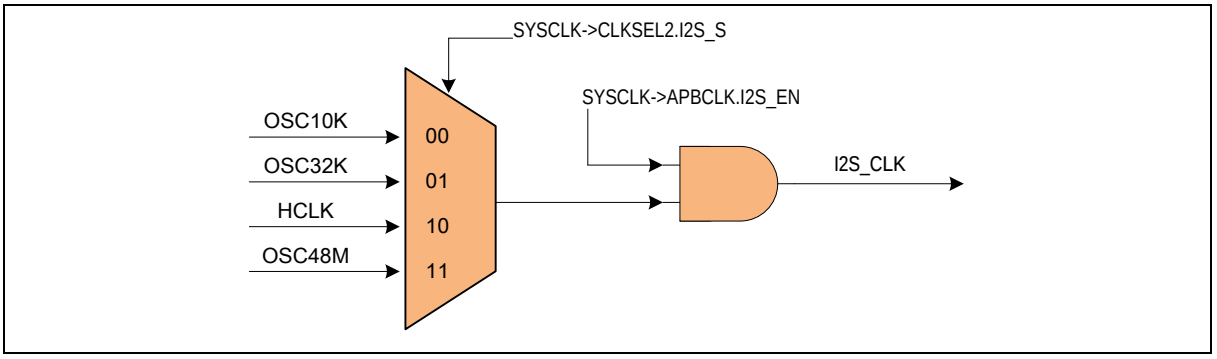


Figure 5-59 I2S Clock Control Diagram

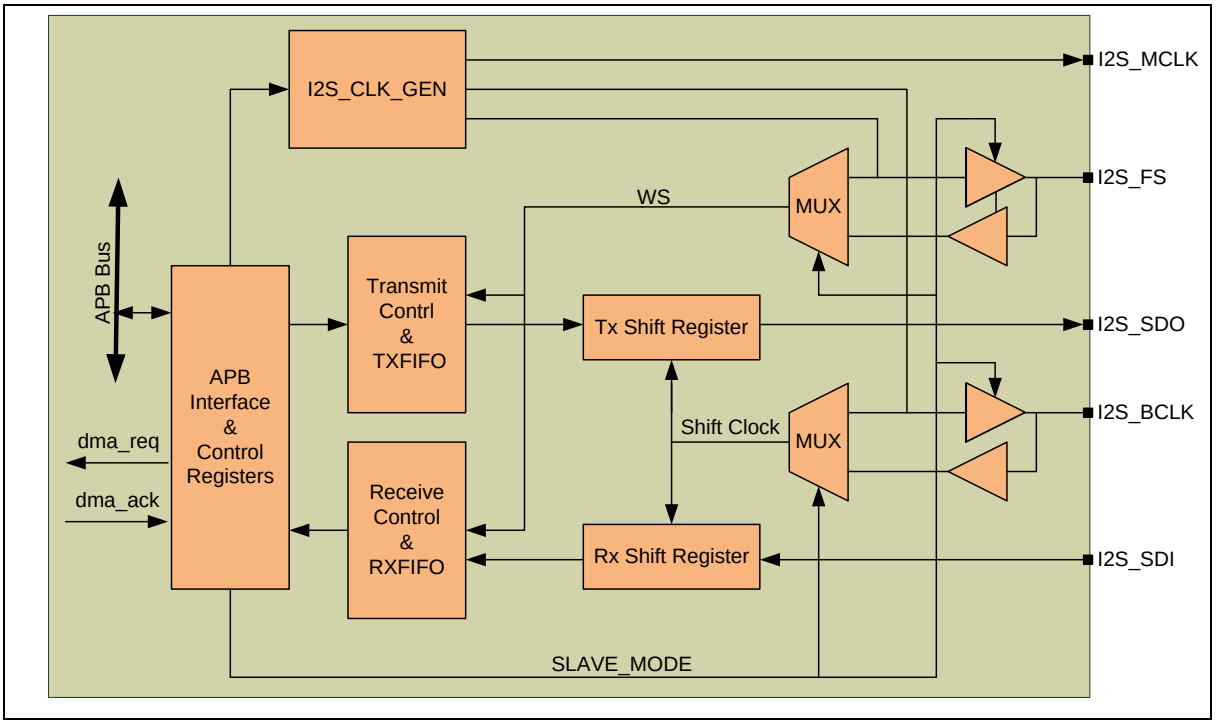


Figure 5-60 I2S Controller Block Diagram

5.13.4 I2S Operation

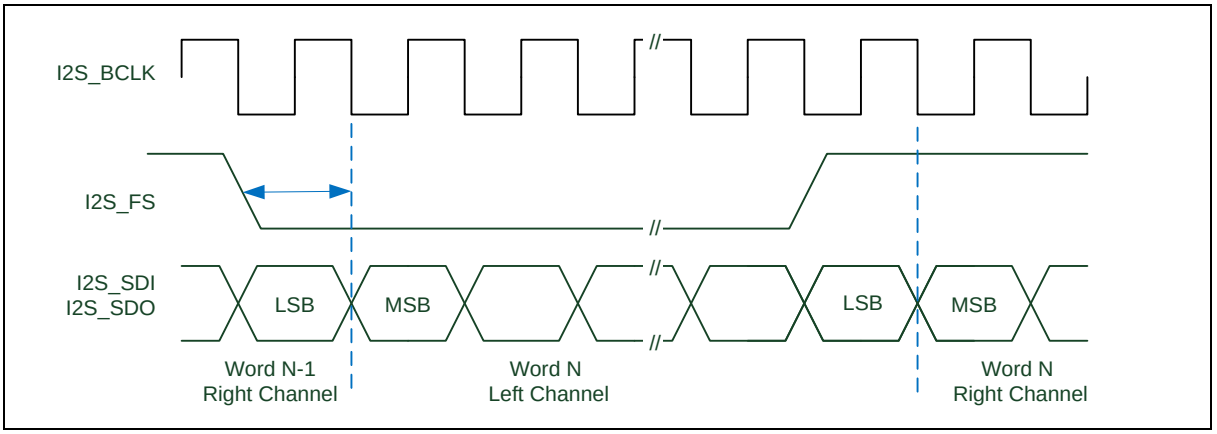


Figure 5-61 I2S Bus Timing Diagram (Format =0)

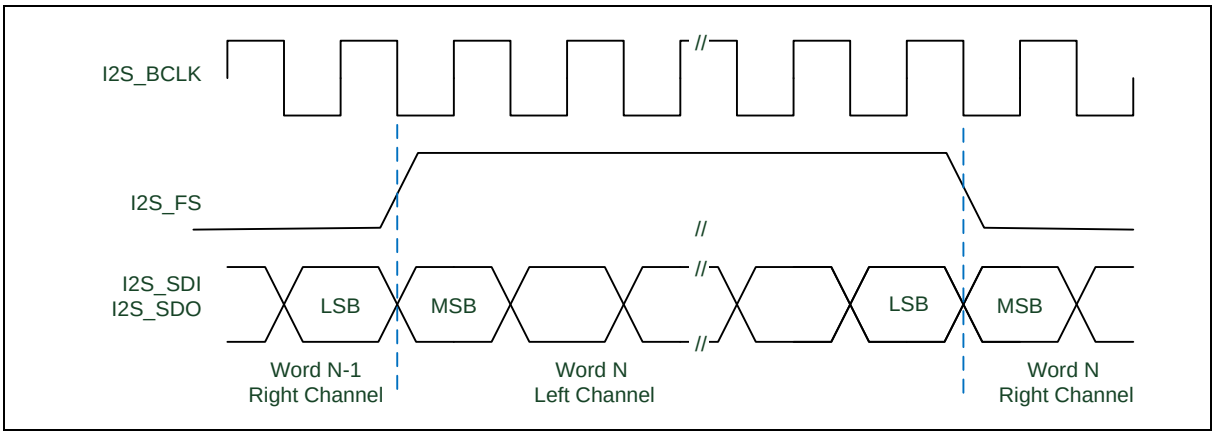


Figure 5-62 MSB Justified Timing Diagram (Format=1)

5.13.5 FIFO operation

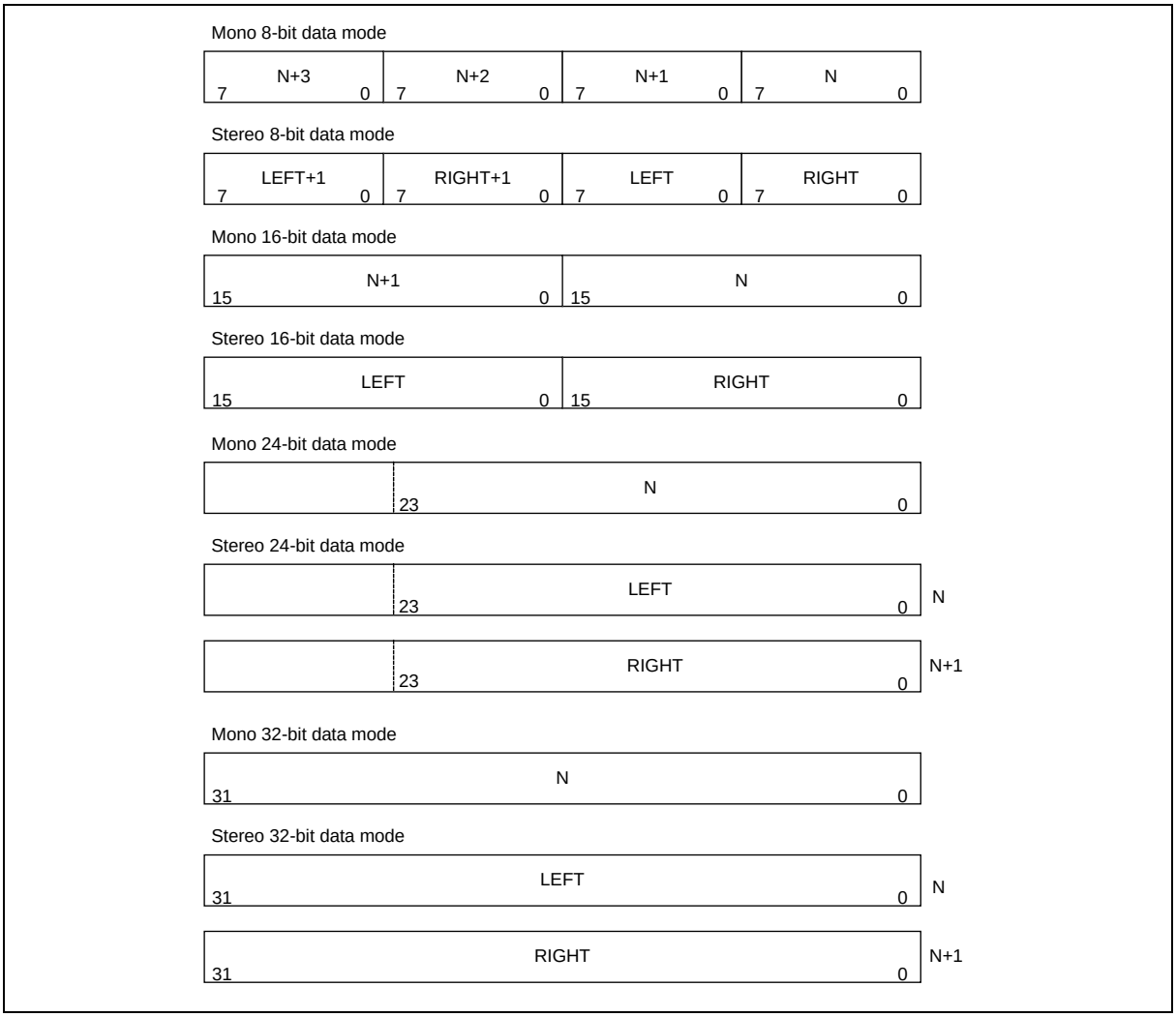


Figure 5-63 FIFO contents for various I2S modes

5.13.6 I2S Control 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_CLKDIV	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_1000
I2S_TX	I2S_BA + 0x10	W	I2S Transmit FIFO Register	0x0000_0000
I2S_RX	I2S_BA + 0x14	R	I2S Receive FIFO Register	0x0000_0000

5.13.7 I2S Control 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**

Table 5-114 I2S Control Register (I2S_CTL, address 0x400A_0000)

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 receiving FIFO, internal pointer is reset to FIFO start point, and RXTH 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 transmitting FIFO, internal pointer is reset to FIFO start point, and TXTH 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 ISD91xx 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 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 ISD91xx. 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 5-61 and Figure 5-62 for timing differences.
[6]	**MONO**	**Monaural data** This parameter sets whether mono or stereo data is processed. See Figure 5-63 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 5-63 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

15	14	13	12	11	10	9	8
BCLKDIV							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**MCLKDIV**		

Table 5-115 I2S Clock Divider Register (I2S_CLKDIV, address 0x400A_0004)

Bits	Description	
[31:16]	Reserved	Reserved
[15:8]	**BCLKDIV**	**Bit Clock Divider** If I2S operates in master mode, bit clock is provided by ISD91xx. 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) aaa Fs x Word_width_in_bytes x 16 For example if Fs aaa 16kHz, and word width is 2-bytes (16bit) then desired bit clock frequency is 512kHz. The bit clock frequency is given by: F(BCLK) aaa F(I2S_CLK) / 2x(BCLKDIV+1) Or, BCLKDIV aaa F(I2S_CLK) / (2 x F(BCLK)) -1 So if F(I2S_CLK) aaa HCLK aaa 49.152MHz$F_{I2S_CLK} = HCLK = 49.152MHz$, desired F(BCLK) aaa 512kHz$F_{BCLK} = 512kHz$ then BCLKDIV aaa 47$BCLK_DIV$=/
[7:3]	Reserved	Reserved
[2:0]	**MCLKDIV**	**Master Clock Divider** ISD9100 series can generate a master clock to synchronously drive an external audio device. If MCLKDIV is set to 0, MCLK is the same as I2S_CLK clock input, otherwise MCLK frequency is given by: F(MCLK) aaa F(I2S_CLK) / (2xMCLKDIV) Or, MCLKDIV aaa F(I2S_CLK) / (2 x F(MCLK)) If the desired MCLK frequency is 254Fs and Fs aaa 16kHz then MCLKDIV aaa 6

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

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

Table 5-116 I2S Interrupt Enable Register (I2S_IEN, address 0x400A_0008)

Bits	Description	
[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
[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_1000

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

Table 5-117 I2S Status Register (I2S_STATUS, address 0x400A_000C)

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	0x0000_0000

Table 5-118 I2S Transmit FIFO Register (I2S_TX, address 0x400A_0010)

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	0x0000_0000

Table 5-119 I2S Receive FIFO Register (I2S_RX, address 0x400A_0014)

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.

5.14 Cyclic Redundancy Check (CRC) Controller

5.14.1 Overview and Features

The ISD9100 series contains a hardware CRC Generator for checking validity of data streams. The CRC function supported is CRC-16-CCITT ($x^{16} + x^{12} + x^5 + 1$). The hardware CRC allows very fast CRC calculation without utilizing any CPU cycles.

The CRC Controller takes input of even sized packets (2, 4, 8 etc.) of up to 512 bytes long and produces a 16bit CRC output. Input to the CRC Controller is via word access of 4 bytes (32 bits) at a time. This word is configurable to either MSB first or LSB first format

5.14.2 Operation

The procedure to use the CRC Generator is as follows:

- Write to CRC_CTL.MODE register to determine data format. A write to this register initializes the packet counter.
- Write to CRC_CTL.PKTLEN register to set the packet length (up to 512 bytes, even sizes only, e.g. 2,4,8). A write to this register resets the CRC value to 0xFFFF.
- A read of CRC_CHECKSUM will return 0xFFFF
- Send data to CRC Generator (CRC_DAT) one (32bit) word at a time. CRC Generator extracts bytes from the word in the order specified by the CRC_CTL.MODE control bit.
- Current CRC result is available from CRC_CHECKSUM register four clock cycles after input word written, including intermediate results. The CRC Generator will stop processing data after CRC_CTL.PKTLEN+1 bytes are sent.

5.14.3 Example

The following is an example of using CRC Generation and Checking with a packet length of 4 bytes.

If the following code was executed:

CRC_CTL.PKTLEN = 3; // Initialize the CRC Generator for a 4 byte packet.

CRC_DAT = 0x2dcf4633; // Note data is sent MSB first in this mode.

Internally the CRC generator would perform the following CRC calculations:

Data In	*CRC*
initial	0xffff
0x2D	0x143f
0xCF	0x4516
0x46	0x2663
0x33	0x2194

The CRC result is 0x2194 after the byte sequence 0x2D, 0xCF, 0x46, 0x33 is processed by the generator.

The 2 byte result can be appended to the original data for checking as shown in the following table.

CRC_CTL.PKTLEN = 5; // Initialize the CRC Generator for a 6 byte packet.

CRC_DAT = 0x2dcf4633; // Note data is sent MSB first in this mode.

CRC_DAT = 0x21940000; // Note data is sent MSB first in this mode.

Data In	*CRC*
initial	0xffff
0x2D	0x143f
0xCF	0x4516
0x46	0x2663
0x33	0x2194
0x21	0x9400
0x94	0x0000

After parsing the 4 bytes data + 2 bytes of CRC result through the CRC Generator, the final result should be 0 which indicates correct data has been transferred.

5.14.4 CRC Controller Register Map

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

Register	Offset	R/W	Description	Reset Value
CRC Base Address: **CRC_BA = 0x4009_0000**				
CRC_CTL	CRC_BA+0x00	R/W	CRC Enable Control Register	0x0000_0000
CRC_DAT	CRC_BA+0x04	R/W	CRC Input Register	0x0000_0000
CRC_CHECKSUM	CRC_BA+0x08	R	CRC Output Register	0x0000_FFFF

5.14.5 CRC Control Register Description

CRC Enable Control

Register	Offset	R/W	Description	Reset Value
CRC_CTL	CRC_BA+0x00	R/W	CRC Enable Control Register	0x0000_0000

Table 5-120 CRC Enable Control Register

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

Bits	Description	
[16]	**MODE**	**CRC LSB mode** Determines whether CRC Generator processes input words (32bit/4Bytes) LSB (least significant byte) first or MSB (most significant byte) first. 0 = CRC input is MSB first (default). 1 = CRC input is LSB first. For example if MODE aaa 1, and 0x01020304 is written to CRC_DAT, bytes will be processed in order 0x04, 0x03, 0x02, 0x01. If MODE aaa 0, then order would be 0x01, 0x02, 0x3, 0x04. Writing any value to this register will flush all previous calculations and restart a new CRC calculation.
[8:0]	**PKTLEN**	**CRC Packet Length** Indicates number of bytes of CRC input to process. CRC calculation will stop once input number of bytes aaa PKTLEN+1. Maximum packet size is 512 bytes, for PKTLEN aaa 511. Writing any value to this register will flush all previous calculations and restart a new CRC calculation.

CRC Input

Register	Offset	R/W	Description	Reset Value
CRC_DAT	CRC_BA+0x04	R/W	CRC Input Register	0x0000_0000

Table 5-121 CRC Input Register

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

Bits	Description	
[31:0]	**DATA**	**CRC Input** The string of bytes to perform CRC calculation on. When MODE aaa 0, CRC performs calculation byte by byte in the order DATA[31:24], DATA[23:16], DATA[15:8], DATA[7:0]. When MODE aaa 1, CRC performs calculation byte by byte in the order DATA[7:0], DATA[15:8], DATA[23:16], DATA[31:24]. If number of input bytes exceeds CRC Packet Length (CRC_CTL[8:0]+1), any additional input bytes will be ignored. The CRC generator takes four clock cycles to process the CRC input. Software must ensure that at least four clock cycles occur between writes of CRC_DAT. Compiled assembly language can be examined to ensure this requirement is met.

CRC Output

Register	Offset	R/W	Description	Reset Value
CRC_CHECKSUM	CRC_BA+0x08	R	CRC Output Register	0x0000_FFFF

Table 5-122 CRC Output Register

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**
CHECKSUM[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHECKSUM[7:0]							

Bits	Description	
[15:0]	**CHECKSUM**	**CRC Output** The result of CRC computation. The result is valid four clock cycles after last CRC_DAT input data is written to CRC generator.

5.15 PDMA Controller

5.15.1 Overview

The ISD9100 series 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, ADC 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 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, ADC 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, and wrap-around addressing.

5.15.3 Block Diagram

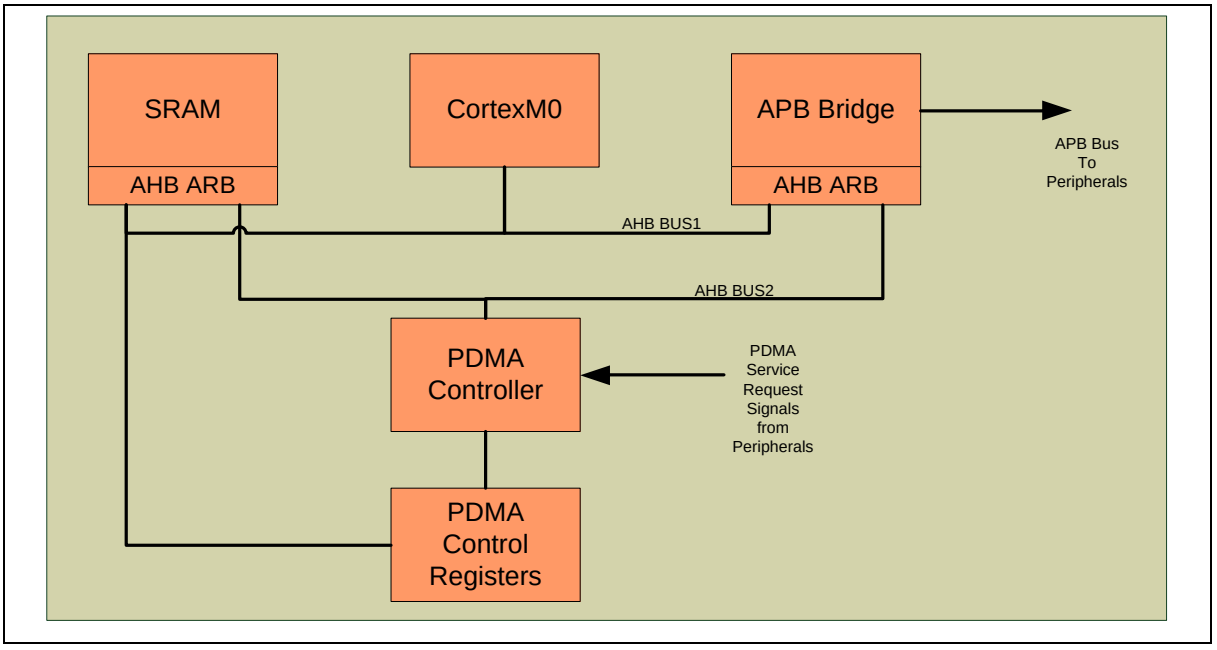


Figure 5-64 PDMA Controller Block Diagram

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_GLOCTL.CHCKEN
- Enable PDMA channel *n* by setting PDMA_DSCTn_CTL.CHEN
- Set source address in PDMA_DSCTn_ENDSA
- Set destination address in PDMA_DSCTn_ENDDA
- Set the transfer count in PDMA_TXBCCHn
- Set transfer mode and address increment mode in PDMA_DSCTn_CTL
- Route peripheral PDMA request signal to channel *n* in service selection register.
- Trigger transfer PDMA_DSCTn_CTL.TXEN

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

5.15.5 PDMA Controller Register Map

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

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMA_BA =** 0x5000_8000+(0x100*x)				
PDMA_DSCT0_CTL	PDMA_BA+0x00	R/W	PDMA Control Register of Channel 0	0x0000_0000
PDMA_DSCT0_ENDSA	PDMA_BA+0x04	R/W	PDMA Transfer Source Address Register of Channel 0	0x0000_0000
PDMA_DSCT0_ENDDA	PDMA_BA+0x08	R/W	PDMA Transfer Destination Address Register of Channel 0	0x0000_0000
PDMA_TXBCCH0	PDMA_BA+0x0C	R/W	PDMA Transfer Byte Count Register of Channel 0	0x0000_0000
PDMA_INLBPCH0	PDMA_BA+0x10	R	PDMA Internal Buffer Pointer Register of Channel 0	0xXXXX_XX00
PDMA_CURSACH0	PDMA_BA+0x14	R	PDMA Current Source Address Register of Channel 0	0x0000_0000
PDMA_CURDACH0	PDMA_BA+0x18	R	PDMA Current Destination Address Register of Channel 0	0x0000_0000
PDMA_CURBCCH0	PDMA_BA+0x1C	R	PDMA Current Byte Count Register of Channel 0	0x0000_0000
PDMA_INTENCH0	PDMA_BA+0x20	R/W	PDMA Interrupt Enable Control Register of Channel 0	0x0000_0001
PDMA_CH0IF	PDMA_BA+0x24	R/W	PDMA Interrupt Status Register of Channel 0	0x0000_0000
PDMA_DSCT1_CTL	PDMA_BA+0x100	R/W	PDMA Control Register of Channel 1	0x0000_0000
PDMA_DSCT1_ENDSA	PDMA_BA+0x104	R/W	PDMA Transfer Source Address Register of Channel 1	0x0000_0000
PDMA_DSCT1_ENDDA	PDMA_BA+0x108	R/W	PDMA Transfer Destination Address Register of Channel 1	0x0000_0000
PDMA_TXBCCH1	PDMA_BA+0x10C	R/W	PDMA Transfer Byte Count Register of Channel 1	0x0000_0000
PDMA_INLBPCH1	PDMA_BA+0x110	R	PDMA Internal Buffer Pointer Register of Channel 1	0xXXXX_XX00
PDMA_CURSACH1	PDMA_BA+0x114	R	PDMA Current Source Address Register of Channel 1	0x0000_0000
PDMA_CURDACH1	PDMA_BA+0x118	R	PDMA Current Destination Address Register of Channel 1	0x0000_0000
PDMA_CURBCCH1	PDMA_BA+0x11C	R	PDMA Current Byte Count Register of Channel 1	0x0000_0000
PDMA_INTENCH1	PDMA_BA+0x120	R/W	PDMA Interrupt Enable Control Register of Channel 1	0x0000_0001
PDMA_CH1IF	PDMA_BA+0x124	R/W	PDMA Interrupt Status Register of Channel 1	0x0000_0000
PDMA_DSCT2_CTL	PDMA_BA+0x200	R/W	PDMA Control Register of Channel 2	0x0000_0000
PDMA_DSCT2_ENDSA	PDMA_BA+0x204	R/W	PDMA Transfer Source Address Register of	0x0000_0000

			Channel 2	
PDMA_DSCT2_ENDDA	PDMA_BA+0x208	R/W	PDMA Transfer Destination Address Register of Channel 2	0x0000_0000
PDMA_TXBCCH2	PDMA_BA+0x20C	R/W	PDMA Transfer Byte Count Register of Channel 2	0x0000_0000
PDMA_INLBPCH2	PDMA_BA+0x210	R	PDMA Internal Buffer Pointer Register of Channel 2	0xXXXX_XX00
PDMA_CURSACH2	PDMA_BA+0x214	R	PDMA Current Source Address Register of Channel 2	0x0000_0000
PDMA_CURDACH2	PDMA_BA+0x218	R	PDMA Current Destination Address Register of Channel 2	0x0000_0000
PDMA_CURBCCH2	PDMA_BA+0x21C	R	PDMA Current Byte Count Register of Channel 2	0x0000_0000
PDMA_INTENCH2	PDMA_BA+0x220	R/W	PDMA Interrupt Enable Control Register of Channel 2	0x0000_0001
PDMA_CH2IF	PDMA_BA+0x224	R/W	PDMA Interrupt Status Register of Channel 2	0x0000_0000
PDMA_DSCT3_CTL	PDMA_BA+0x300	R/W	PDMA Control Register of Channel 3	0x0000_0000
PDMA_DSCT3_ENDSA	PDMA_BA+0x304	R/W	PDMA Transfer Source Address Register of Channel 3	0x0000_0000
PDMA_DSCT3_ENDDA	PDMA_BA+0x308	R/W	PDMA Transfer Destination Address Register of Channel 3	0x0000_0000
PDMA_TXBCCH3	PDMA_BA+0x30C	R/W	PDMA Transfer Byte Count Register of Channel 3	0x0000_0000
PDMA_INLBPCH3	PDMA_BA+0x310	R	PDMA Internal Buffer Pointer Register of Channel 3	0xXXXX_XX00
PDMA_CURSACH3	PDMA_BA+0x314	R	PDMA Current Source Address Register of Channel 3	0x0000_0000
PDMA_CURDACH3	PDMA_BA+0x318	R	PDMA Current Destination Address Register of Channel 3	0x0000_0000
PDMA_CURBCCH3	PDMA_BA+0x31C	R	PDMA Current Byte Count Register of Channel 3	0x0000_0000
PDMA_INTENCH3	PDMA_BA+0x320	R/W	PDMA Interrupt Enable Control Register of Channel 3	0x0000_0001
PDMA_CH3IF	PDMA_BA+0x324	R/W	PDMA Interrupt Status Register of Channel 3	0x0000_0000
PDMA_GLOCTL	PDMA_BA+0xF00	R/W	PDMA Global Control Register	0x0000_0000
PDMA_SVCSEL	PDMA_BA+0xF04	R/W	PDMA Service Selection Control Register	0xFFFF_FFFF
PDMA_GLOBALIF	PDMA_BA+0xF0C	R	PDMA Global Interrupt Status Register	0x0000_0000

5.15.6 PDMA Control Register Description

PDMA Control TXENl and Status Register (PDMA_DSCTn_CTL)(n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_CTL	PDMA_BA+0x00	R/W	PDMA Control Register of Channel 0	0x0000_0000
PDMA_DSCT1_CTL	PDMA_BA+0x100	R/W	PDMA Control Register of Channel 1	0x0000_0000
PDMA_DSCT2_CTL	PDMA_BA+0x200	R/W	PDMA Control Register of Channel 2	0x0000_0000
PDMA_DSCT3_CTL	PDMA_BA+0x300	R/W	PDMA Control Register of Channel 3	0x0000_0000

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

Table 5-123 PDMA Control and Status Register (PDMA_DSCTn_CTL, address 0x5000_8000 + *n* * 0x100)

Bits	Description	
[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.
[20:19]	**TWIDTH**	**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).

[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 BYTECNT aaa 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 BYTECNT aaa 32 then an interrupt could be generated when 32 bytes were sent and PDMA wraps around. x1x1: Both half and w interrupts generated.
[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_CURBCCHn (Current Byte Count) equals zero, the PDMA_CURDACHn (Current Destination Address) and PDMA_CURBCCHn registers will be reloaded from the PDMA_DSCTn_ENDDA (Destination Address) and PDMA_TXBCCHn (Byte Count) registers automatically and PDMA will start another transfer. Cycle continues until software sets PDMACKEN=0. When PDMACKEN 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_CURBCCHn (Current Byte Count) equals zero, the PDMA_CURSACHn (Current Source Address) and PDMA_CURBCCHn registers will be reloaded from the PDMA_DSCTn_ENDSA (Source Address) and PDMA_TXBCCHn (Byte Count) registers automatically and PDMA will start another transfer. Cycle continues until software sets PDMACKEN aaa 0. When PDMACKEN 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_DSCTn_ENDSA)(n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_ENDSA	PDMA_BA+0x04	R/W	PDMA Transfer Source Address Register of Channel 0	0x0000_0000
PDMA_DSCT1_ENDSA	PDMA_BA+0x104	R/W	PDMA Transfer Source Address Register of Channel 1	0x0000_0000
PDMA_DSCT2_ENDSA	PDMA_BA+0x204	R/W	PDMA Transfer Source Address Register of Channel 2	0x0000_0000
PDMA_DSCT3_ENDSA	PDMA_BA+0x304	R/W	PDMA Transfer Source Address Register of Channel 3	0x0000_0000

Table 5-124 PDMA Source Address Register (PDMA_DSCTn_ENDSA, address 0x5000_8004 + *n**0x100)

Bits	Description	
[31:0]	**ENDSA**	**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_DSCTn_ENDDA)(n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_ENDDA	PDMA_BA+0x08	R/W	PDMA Transfer Destination Address Register of Channel 0	0x0000_0000
PDMA_DSCT1_ENDDA	PDMA_BA+0x108	R/W	PDMA Transfer Destination Address Register of Channel 1	0x0000_0000
PDMA_DSCT2_ENDDA	PDMA_BA+0x208	R/W	PDMA Transfer Destination Address Register of Channel 2	0x0000_0000
PDMA_DSCT3_ENDDA	PDMA_BA+0x308	R/W	PDMA Transfer Destination Address Register of Channel 3	0x0000_0000

Table 5-125 PDMA Destination Address Register (PDMA_DSCTn_ENDDA, address 0x5000_8008 + *n**0x100)

Bits	Description	
[31:0]	**ENDDA**	**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_TXBCCHn)(n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_TXBCCH0	PDMA_BA+0x0C	R/W	PDMA Transfer Byte Count Register of Channel 0	0x0000_0000
PDMA_TXBCCH1	PDMA_BA+0x10C	R/W	PDMA Transfer Byte Count Register of Channel 1	0x0000_0000
PDMA_TXBCCH2	PDMA_BA+0x20C	R/W	PDMA Transfer Byte Count Register of Channel 2	0x0000_0000
PDMA_TXBCCH3	PDMA_BA+0x30C	R/W	PDMA Transfer Byte Count Register of 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**
BYTECNT [15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BYTECNT [7:0]							

Table 5-126 PDMA Transfer Byte Count Register (PDMA_TXBCCHn, address 0x5000_800C + *n**0x100)

Bits	Description	
[31:24]	Reserved	Reserved
[15:0]	**BYTECNT**	**PDMA Transfer Byte Count Register** This register controls the transfer byte count of PDMA. Maximum value is 0xFFFF. Note: When in memory-to-memory (PDMA_DSCTn_CTL.MODESEL aaa 00b) mode, the transfer byte count must be word aligned, that is multiples of 4bytes.

PDMA Internal Buffer Pointer Register (PDMA_INLBPCHn)(n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_INLBPCH0	PDMA_BA+0x10	R	PDMA Internal Buffer Pointer Register of Channel 0	0xXXXX_XX00
PDMA_INLBPCH1	PDMA_BA+0x110	R	PDMA Internal Buffer Pointer Register of Channel 1	0xXXXX_XX00
PDMA_INLBPCH2	PDMA_BA+0x210	R	PDMA Internal Buffer Pointer Register of Channel 2	0xXXXX_XX00
PDMA_INLBPCH3	PDMA_BA+0x310	R	PDMA Internal Buffer Pointer Register of Channel 3	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				**BUFPTR**			

Table 5-127 PDMA Internal Buffer Point Register (PDMA_INLBPCHn, address 0x5000_8010 + *n**0x100)

Bits	Description	
[31:4]	Reserved	Reserved
[3:0]	**BUFPTR**	**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_CURSACHn) (n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_CURSACH0	PDMA_BA+0x14	R	PDMA Current Source Address Register of Channel 0	0x0000_0000
PDMA_CURSACH1	PDMA_BA+0x114	R	PDMA Current Source Address Register of Channel 1	0x0000_0000
PDMA_CURSACH2	PDMA_BA+0x214	R	PDMA Current Source Address Register of Channel 2	0x0000_0000
PDMA_CURSACH3	PDMA_BA+0x314	R	PDMA Current Source Address Register of Channel 3	0x0000_0000

Table 5-128 PDMA Current Source Address Register (PDMA_CURSACHn, address 0x5000_8014 + *n**0x100)

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

PDMA Current Destination Address Register (PDMA_CURDACHn) (n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_CURDACH0	PDMA_BA+0x18	R	PDMA Current Destination Address Register of Channel 0	0x0000_0000
PDMA_CURDACH1	PDMA_BA+0x118	R	PDMA Current Destination Address Register of Channel 1	0x0000_0000
PDMA_CURDACH2	PDMA_BA+0x218	R	PDMA Current Destination Address Register of Channel 2	0x0000_0000
PDMA_CURDACH3	PDMA_BA+0x318	R	PDMA Current Destination Address Register of Channel 3	0x0000_0000

Table 5-129 PDMA Current Destination Address Register (PDMA_CURDACHn, address 0x5000_8018 + *n**0x100)

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

PDMA Current Byte Count Register (PDMA_CURBCCHn) (n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_CURBCCH0	PDMA_BA+0x1C	R	PDMA Current Byte Count Register of Channel 0	0x0000_0000
PDMA_CURBCCH1	PDMA_BA+0x11C	R	PDMA Current Byte Count Register of Channel 1	0x0000_0000
PDMA_CURBCCH2	PDMA_BA+0x21C	R	PDMA Current Byte Count Register of Channel 2	0x0000_0000
PDMA_CURBCCH3	PDMA_BA+0x31C	R	PDMA Current Byte Count Register of Channel 3	0x0000_0000

Table 5-130 PDMA Current Byte Count Register (PDMA_CURBCCHn, address 0x5000_801C + *n**0x100)

Bits	Description	
[31:16]	Reserved	Reserved
[15:0]	**CURBC**	**PDMA Current Byte Count Register (Read Only)** This field indicates the current remaining byte count of PDMA transfer. This register is initialized with PDMA_TXBCCHn register when PDMA is triggered or when a wraparound occurs

PDMA Interrupt Enable Control Register (PDMA_INTENCHn) (n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_INTENCH0	PDMA_BA+0x20	R/W	PDMA Interrupt Enable Control Register of Channel 0	0x0000_0001
PDMA_INTENCH1	PDMA_BA+0x120	R/W	PDMA Interrupt Enable Control Register of Channel 1	0x0000_0001
PDMA_INTENCH2	PDMA_BA+0x220	R/W	PDMA Interrupt Enable Control Register of Channel 2	0x0000_0001
PDMA_INTENCH3	PDMA_BA+0x320	R/W	PDMA Interrupt Enable Control Register of Channel 3	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					WAINTEN	TXOKIEN	TXABTIEN

Table 5-131 PDMA Interrupt Enable Control Register (PDMA_INTENCHn, address 0x5000_8020 + *n**0x100)

Bits	Description	
[31:3]	Reserved	Reserved
[2]	**WAINTEN**	**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]	**TXOKIEN**	**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]	**TXABTIEN**	**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_CHnIF) (n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_CH0IF	PDMA_BA+0x24	R/W	PDMA Interrupt Status Register of Channel 0	0x0000_0000
PDMA_CH1IF	PDMA_BA+0x124	R/W	PDMA Interrupt Status Register of Channel 1	0x0000_0000
PDMA_CH2IF	PDMA_BA+0x224	R/W	PDMA Interrupt Status Register of Channel 2	0x0000_0000
PDMA_CH3IF	PDMA_BA+0x324	R/W	PDMA Interrupt Status Register of Channel 3	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				WAIF			
7	6	5	4	3	2	1	0
Reserved						TXOKIF	TXABTIF

Table 5-132 PDMA Interrupt Enable Status Register (PDMA_CHnIF, address 0x5000_8024 + *n**0x100)

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]	**WAIF**	**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 aaa Current transfer finished flag (CURBC aaaaaa 0). 0100 aaa Current transfer half complete flag (CURBC aaaaaa BYTECNT/2).
[1]	**TXOKIF**	**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]	**TXABTIF**	**PDMA Read/Write Target Abort Interrupt Flag** This flag indicates a Target Abort interrupt condition has occurred. This condition can happen if attempt is made to read/write from invalid or non-existent memory space. It occurs when PDMA controller receives a bus error from AHB master. Upon occurrence PDMA will stop transfer and go to idle state. To resume, software must reset PDMA channel and initiate transfer again. 0 = No bus ERROR response received. 1 = Bus ERROR response received. NOTE: This bit is cleared by writing 1 to itself.

PDMA Global Control Register (PDMA_GLOCTL)

Register	Offset	R/W	Description	Reset Value
PDMA_GLOCTL	PDMA_BA+0xF00	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				CHCKEN			
7	6	5	4	3	2	1	0
Reserved							SWRST

Table 5-133 PDMA Global Control Register (PDMA_GLOCTL, address 0x5000_8F00)

Bits	Description	
[31:17]	Reserved	Reserved
[11:8]	**CHCKEN**	**PDMA Controller Channel Clock Enable Control** To enable clock for channel *n* CHCKEN[*n*] must be set. CHCKEN[*n*] aaa 1: Enable Channel *n* clock CHCKEN[*n*] aaa 0: Disable Channel *n* 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 (PDMA_SVCSEL)

Register	Offset	R/W	Description	Reset Value
PDMA_SVCSEL	PDMA_BA+0xF04	R/W	PDMA Service Selection Control Register	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
UARTXSEL				UARTRXSEL			
15	14	13	12	11	10	9	8
DPWMTXSEL				ADCRXSEL			
7	6	5	4	3	2	1	0
SPITXSEL				SPIRXSEL			

Table 5-134 PDMA Service Selection Control Register (PDMA_SVCSEL, address 0x5000_8F04)

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]	UARTXSEL	**PDMA UART0 Transmit Selection** This field defines which PDMA channel is connected to UART0 peripheral transmit (PDMA destination) request.
[19:16]	UARTRXSEL	**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]	ADCRXSEL	**PDMA ADC Receive Selection** This field defines which PDMA channel is connected to ADC peripheral receive (PDMA source) request.
[7:4]	SPITXSEL	**PDMA SPI0 Transmit Selection** This field defines which PDMA channel is connected to SPI0 peripheral transmit (PDMA destination) request.
[3:0]	SPIRXSEL	**PDMA SPI0 Receive Selection** This field defines which PDMA channel is connected to SPI0 peripheral receive (PDMA source) request.

PDMA Global Interrupt Status Register (PDMA_GLOBALIF)

Register	Offset	R/W	Description	Reset Value
PDMA_GLOBALIF	PDMA_BA+0xF0C	R	PDMA Global Interrupt Status Register	0x0000_0000

Table 5-135 PDMA Global Interrupt Status Register (PDMA_GLOBALIF, address 0x5000_8F0C)

Bits	Description	
[3:0]	**GLOBALIF**	**Interrupt Pin Status (Read Only)** GLOBALIF[*n*] is the interrupt status of PDMA channel *n.*

6 FLASH MEMORY CONTROLLER (FMC)

6.1 Overview

The ISD9100 series is available with 141K 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 sector address boundary. Erasable sector size is 1K Byte. This boundary is also specified in the CONFIG memory.

LDROM is a fixed 4K Byte in size, but if not required can be incorporated into the APROM address space of the 141K Byte device for a total device memory of 145K Byte.

6.2 Features

- AHB interface compatible
- Runs up to 50 MHz with zero wait-state for continuous address read access
- 141KB application program memory (APROM)
- 4KB in system programming (ISP) boot loader program memory (LDROM)
- Configurable data flash with 1k Bytes sector erase unit
- Programmable data flash start address.
- In System Program (ISP) capability to update on chip Flash EEPROM

6.3 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:

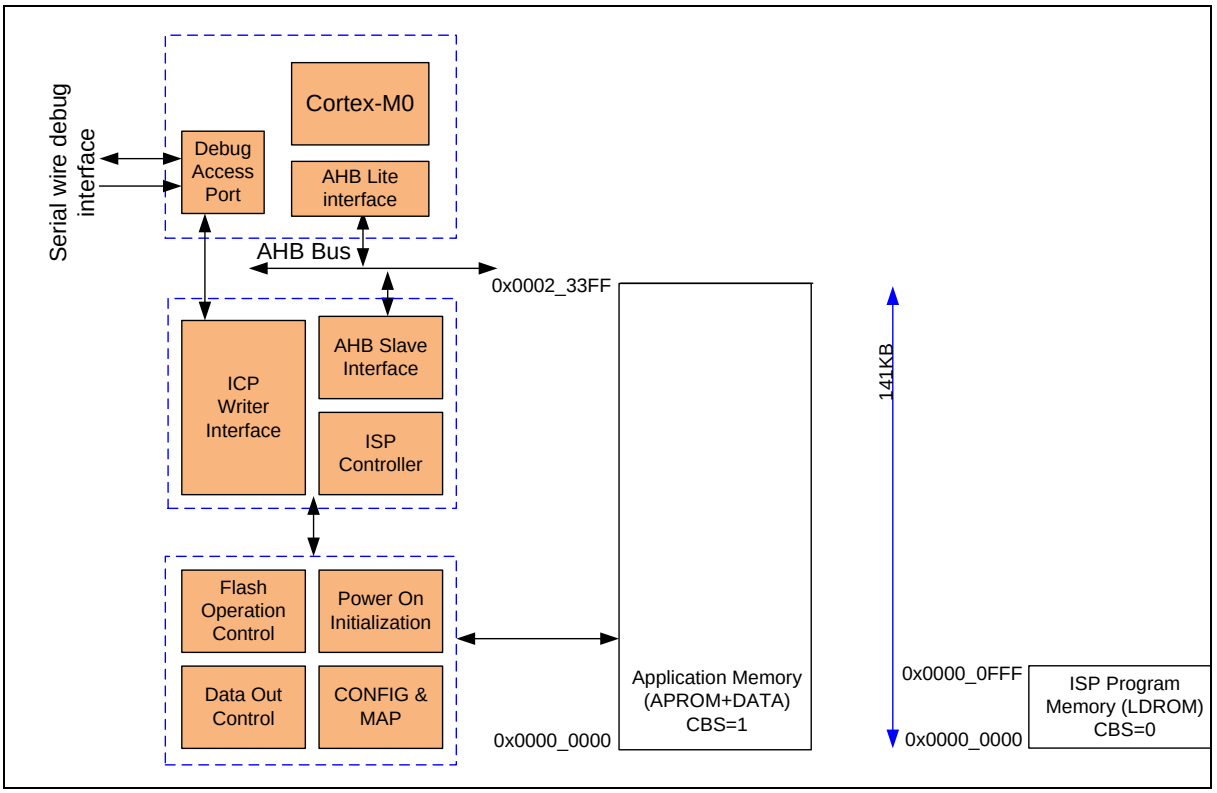


Figure 6-1 Flash Memory Control Block Diagram

6.4 Flash Memory Organization

The ISD9100 series flash memory consists of Application Program (APROM) memory (141KB), 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. An additional 504Bytes are available in CONFIG memory for the user to store custom configuration data. The first two CONFIG words are loaded from CONFIG memory at power-on into device control registers to initialize certain chip functions. The data flash start address (FMC_DFBA) is defined in CONFIG memory and determines the relative size of the APROM and DATAF partitions.

Table 6-1 Memory Address Map

Block Name	Size	Start Address	End Address
APROM	141 KB	0x0000_0000	0x0002_33FF (141KB) OR DFBADR-1 if DFEN!=0
DATAF	User Configurable	DFBADR	0x0002_33FF (141KB)
LDROM	4 KB	0x0010_0000	0x0010_0FFF
CONFIG	512B	0x0030_0000	0x0030_01FF

The Flash memory organization is shown as below:

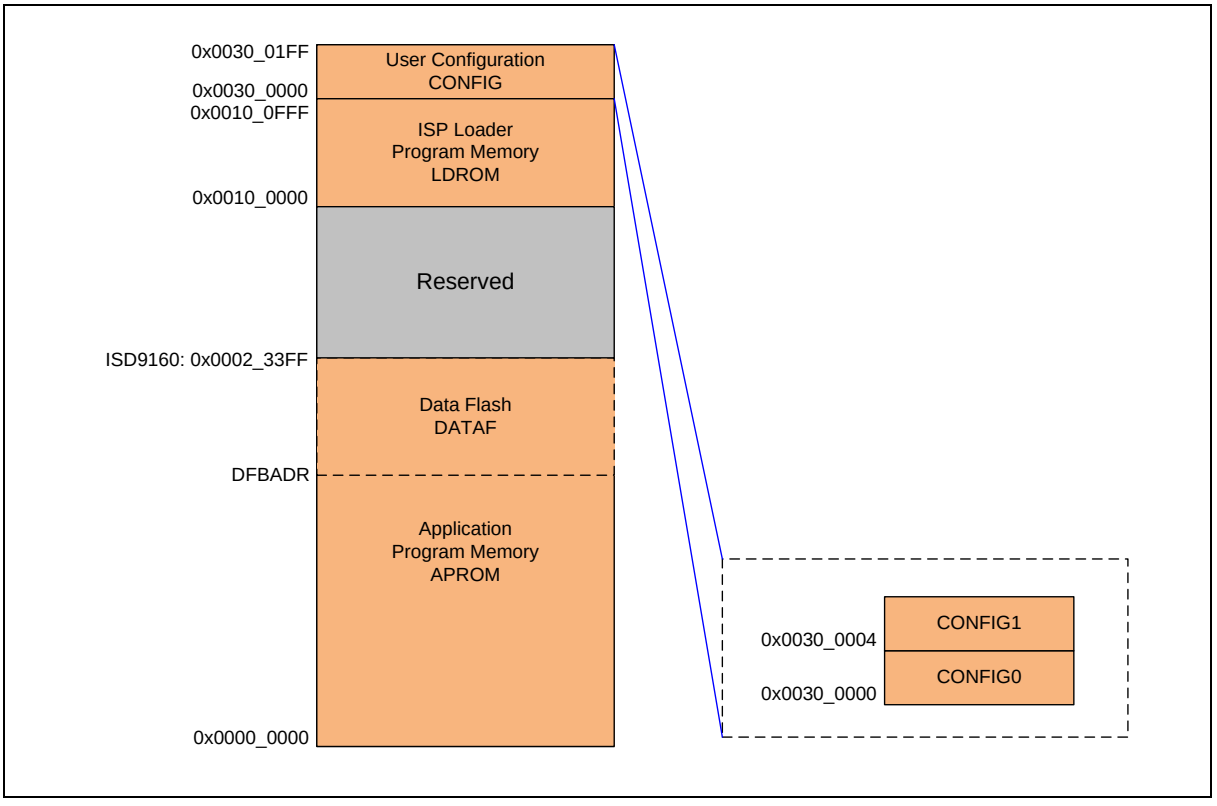


Figure 6-2 Flash Memory Organization

6.5 Boot Selection

The ISD9100 series 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 4KB boot loader program memory is used to store ISP firmware. The user customizes this firmware to implement a protocol specific to their system to download updated application code. This firmware could utilize device peripherals such as UART, SPI or I2C to fetch new application code. The memory area from which the ISD9100 series boots is controlled by the CBS bit in Config0 register.

6.6 Data Flash (DATAF)

The ISD9100 series 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 sector is 1Kbyte and minimum write size is one word (4Bytes). An erase operation resets all memory in sector to value 0xFF. A write operation can only change a '1' bit to a '0' bit. If a subset of the sector needs to be changed, the entire 1KB sector must be copied to another page or into SRAM in advance as entire sector 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 FMC_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 (141KB) and N is number of Kbytes (sectors) reserved for data flash. In addition, for the 141KB device, the LDROM partition can be disabled and included in APROM/DATAF memory by setting the LDROM_EN configuration bit low allowing a total of 145KB of memory available to APROM/DATAF.

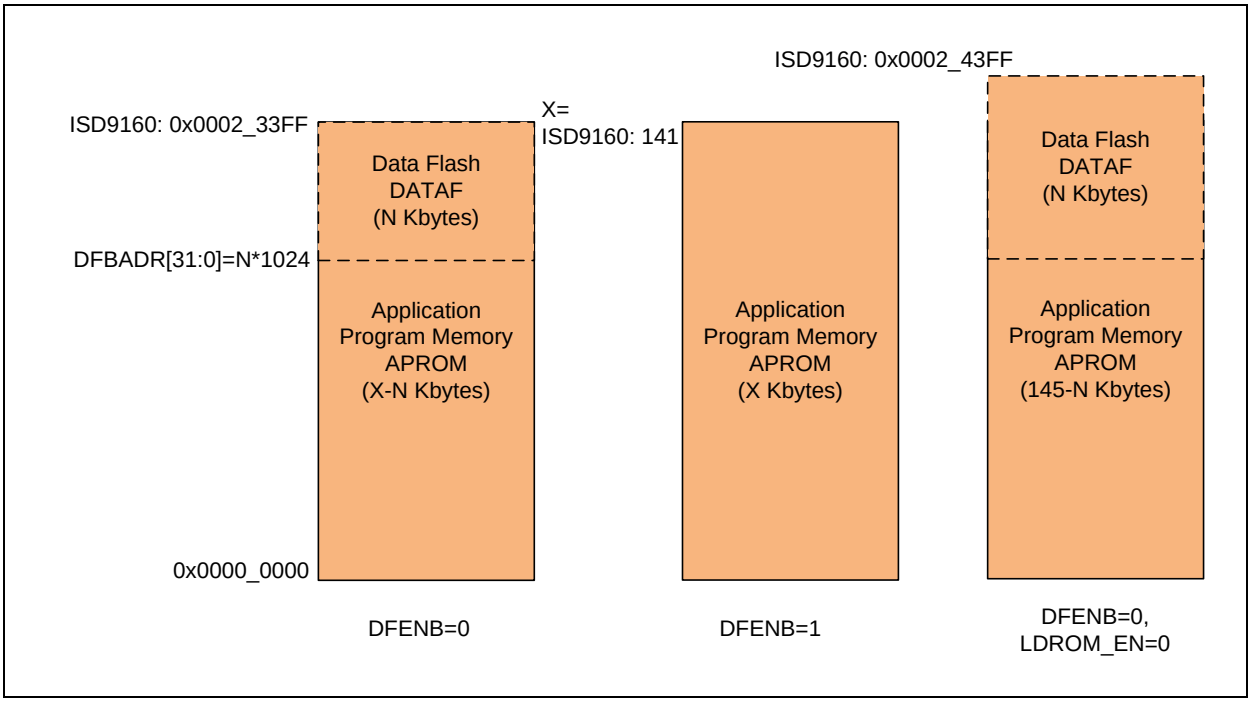


Figure 6-3 Flash Memory Structure

6.7 User Configuration (CONFIG)

6.7.1 CONFIG0 (ISP Address = 0x0030_0000)

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

Table 6-2 User Configuration Register 0 (Config0, address 0x0030_0000 accessible through ISP only)

Config0	Address = 0x0030_0000	
Bits	**Description**	
[31:23]	Reserved	Reserved
[23]	**CBODEN**	**Brown Out Detector Enable** If set to '1' the Brown Out Detector (BOD) will be enabled after power up. It will be configured at lowest voltage (2.1V) and if brown out condition detected will trigger the NMI interrupt to processor. 0=Disable brown out detect after power on 1= Enable
[22:8]	Reserved	Reserved
[7]	**CBS**	**Configuration Boot Selection** 0 = Chip will boot from LDROM, 1 = Chip will boot from APROM
[6:3]	Reserved	Reserved
[2]	**LDROMEN**	LDROM Control Bit 0=disable 1= enable
[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

6.7.2 CONFIG1 (Address = 0x0030_0004)

Table 6-3 User Configuration Register 1 (Config1, address 0x0030_0004 accessible through ISP only)

Config1	**Address = 0x0030_0004**	
Bits	**Description**	
[31:20]	Reserved	Reserved It is mandatory to program 0x00 to these Reserved bits
[19:0]	**DFBADR**	**Data Flash Base Address** This pointer sets the address for the start of data flash memory. Address must be on a 1KB sector boundary so DFBADR[9:0] must be 0x000.

6.8 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 ISD9100 series 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.

6.8.1 ISP Procedure

The ISD9100 series 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 (see Protected Register Lock Key Register (SYS_REGLCTL)) to gain access. This procedure is to protect the flash memory from unintentional access.

To enable ISP functionality software must first ensure the ISP clock (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.

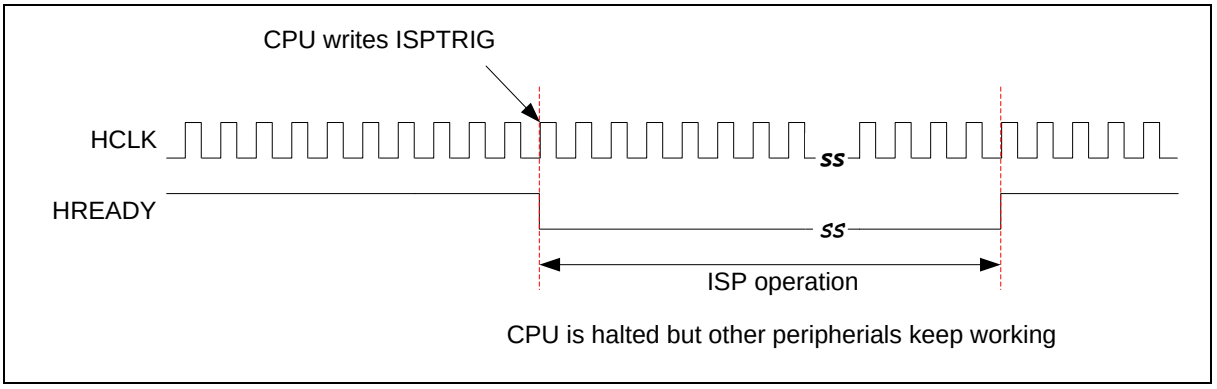


Figure 6-4 ISP Operation Timing

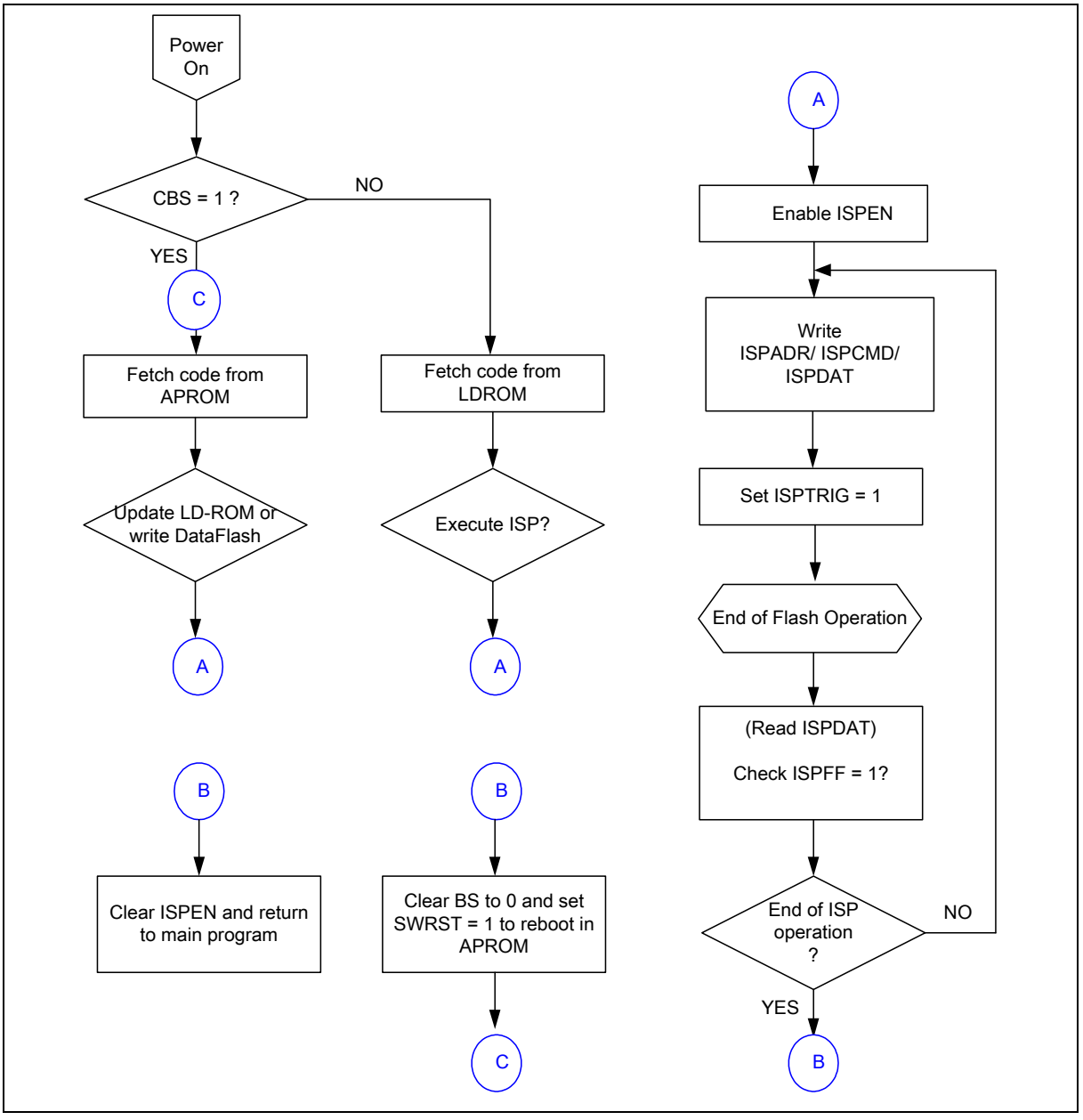


Figure 6-5 Boot Sequence and ISP Procedure

The ISP command set is shown in Table 6-4. 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.

There is an ISP command to read the device ID register. This register returns a code that reports the memory configuration of the ISD9100 series part as given in Table 6-5.

Table 6-4 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	0x1D00_01nn. See Table 6-5
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

Table 6-5 Device ID Memory Size

DID[7:4]	**Flash Size (KB)**
8	145

DID[3:0]	**RAM Size (KB)**
3	12

6.9 Flash Control Register Map

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

Register	Offset	R/W	Description	Reset Value
FMC Base Address: **FMC_BA=0x5000_C000**				
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0000_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

6.10 Flash Control Register Description

ISP Control Register (FMC_ISPCTL)

The FMC_ISPCTL register is a protected register, user must first follow the unlock sequence (see Protected Register Lock Key Register (SYS_REGLCTL)) to gain access.

Register	Offset	R/W	Description	Reset Value
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0000_0000

7	6	5	4	3	2	1	0
SWRST	ISPFF	LDUEN	CFGUEN	-	-	BS	ISPEN

Table 6-6 ISP Control Register (FMC_ISPCTL, address 0x5000_C000)

Bits	Description	
[7]	SWRST	**Software Reset** Writing 1 to this bit will initiate a software reset. It is cleared by hardware after reset.
[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.
[3:2]	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

Table 6-7 ISP Address Register (FMC_ISPADDR, address 0x5000_C004)

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 ISPARD[1:0] must be 00b for correct ISP operation.

ISP Data Register (FMC_ISPDAT)

Register	Offset	R/W	Description	Reset Value
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000

Table 6-8 ISP Data Register (FMC_ISPDAT, address 0x5000_C008)

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

Table 6-9 ISP Data Register (FMC_ISPCMD, address 0x5000_C00C)

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 (see Protected Register Lock Key Register (SYS_REGLCTL)) to gain access.

Register	Offset	R/W	Description	Reset Value
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000

Table 6-10 ISP Trigger Control Register (FMC_ISPTRG, address 0x5000_C010)

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 (see Protected Register Lock Key Register (SYS_REGLCTL)) 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

Table 6-11 Data Flash Base Address Register (FMC_DFBA, address 0x5000_C014)

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's configuration; register content is loaded from Config1 when chip is reset.

7 ANALOG SIGNAL PATH BLOCKS

This section describes the functional blocks that perform analog signal functions on the ISD9100 series. This includes the ADC, DPWM Speaker Driver, PGA Gain Amplifier, Automatic Gain Control and a variety of auxiliary analog functional blocks.

7.1 Audio Analog-to-Digital Converter (ADC)

7.1.1 Functional Description

The ISD9100 series includes a 2nd Order Delta-Sigma Audio Analog-to-Digital converter providing SNR >85dB and THD >70dB. 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. This provides support for standard audio sampling rates from 8kHz to 48kHz.

7.1.2 Features

- Front-end PGA providing gain range of -12dB – 35dB.
- Boost Gain stage of 0dD or 26dB.
- Configurable OSR (Over Sampling Ratio) of 64/128/192/384
- Configurable clock rate through master oscillator integer division.
- Decimation signal can be used directly or passed to biquad filter for further filtering.
- Audio data buffered to 8 word FIFO, accessible via APB and PDMA.

7.1.3 Block Diagram

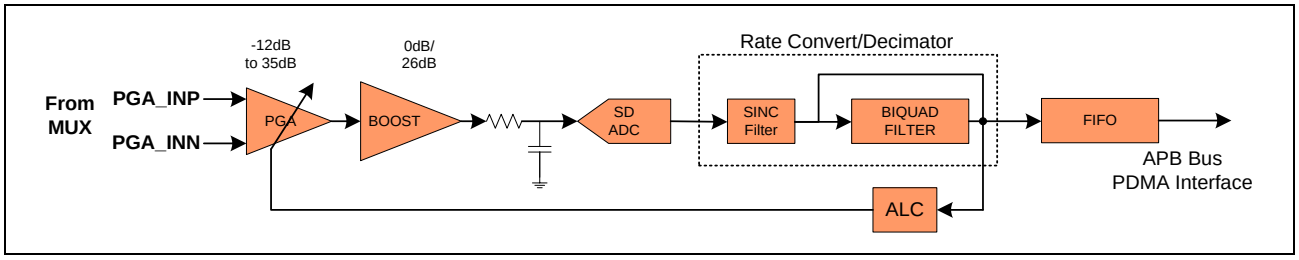


Figure 7-1 ADC Signal Path Block Diagram

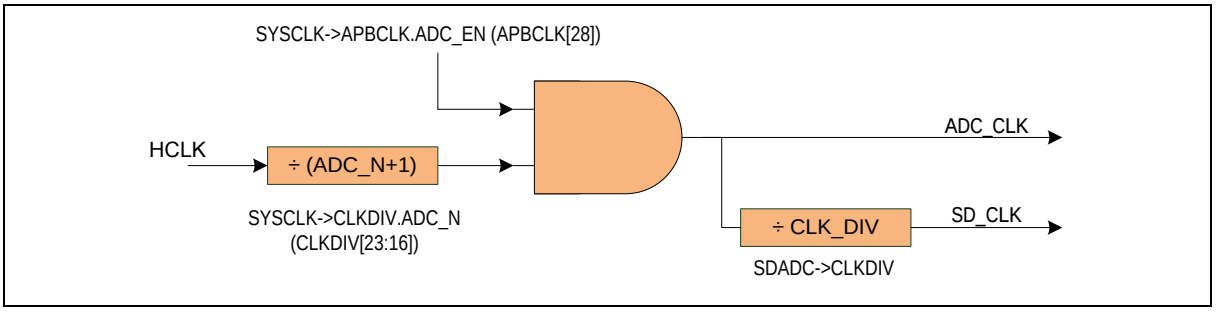


Figure 7-2 ADC Clock Control

7.1.4 Operation

The ADC is an Audio Delta-Sigma 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 SDCLK, and the oversampling ratio.

The data stream generated by the ADC 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.

If FIFO is not serviced then oldest data is over-written such that the FIFO always contains the eight most recent samples.

7.1.4.1 *Determining Sample Rate*

The maximum clock rate of the Delta-Sigma Converter is 6.144MHz. Best performance is gained with clocks rates between 1.024MHz and 4.096MHz. Sample rate is given by the following formula:

$$F_s = HCLK \div CLK\_DIV \div OSR$$

Tables of common audio sample rates are provided below.

Table 7-1 Sample Rates for HCLK=49.152MHz

HCLK=49.152MHz	SD_CLK	Sample Rate (Hz) for OSR			
ADC CLKDIV		64	128	192	384
8	6,144,000	96,000	48,000	32,000	16,000
16	3,072,000	48,000	24,000	16,000	8,000
24	2,048,000	32,000	16,000	10,667	5,333
32	1,536,000	24,000	12,000	8,000	4,000
48	1,024,000	16,000	8,000	5,333	2,667

Table 7-2 Sample Rates for HCLK=32.768MHz

HCLK=32.768MHz	SD_CLK	Sample Rate (Hz) for OSR			
ADC CLKDIV		64	128	192	384
8	4,096,000	64,000	32,000	21,333	10,667
16	2,048,000	32,000	16,000	10,667	5,333
24	1,365,333	21,333	10,667	7,111	3,556
32	1,024,000	16,000	8,000	5,333	2,667

Table 7-3 Sample Rates for HCLK=24.576MHz

HCLK=24.576MHz	SD_CLK	Sample Rate (Hz) for OSR			
ADC CLKDIV		64	128	192	384
8	3,072,000	48,000	24,000	16,000	8,000
12	2,048,000	32,000	16,000	10,667	5,333
16	1,536,000	24,000	12,000	8,000	4,000
24	1,024,000	16,000	8,000	5,333	2,667

Table 7-4 Sample Rates for HCLK=16.384MHz

HCLK=16.384MHz	SD_CLK	Sample Rate (Hz) for OSR			
ADC CLKDIV		64	128	192	384
4	4,096,000	64,000	32,000	21,333	10,667
8	2,048,000	32,000	16,000	10,667	5,333
16	1,024,000	16,000	8,000	5,333	2,667
24	682,667	10,667	5,333	3,556	1,778
32	512,000	8,000	4,000	2,667	1,333

7.1.4.2 *Configuring Analog Path*

To operate the ADC the entire analog path from analog input to ADC needs to be configured for correct operation. This involves:

- Selecting and powering up VMID reference.
- Powering up modulator and reference buffers.
- Selecting an input source with the analog MUX.
- Configure sample rate and ADC clock source.

7.1.4.3 *Interrupt Sources*

The ADC 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 ADC FIFO output to generate interrupts when set levels are exceeded.

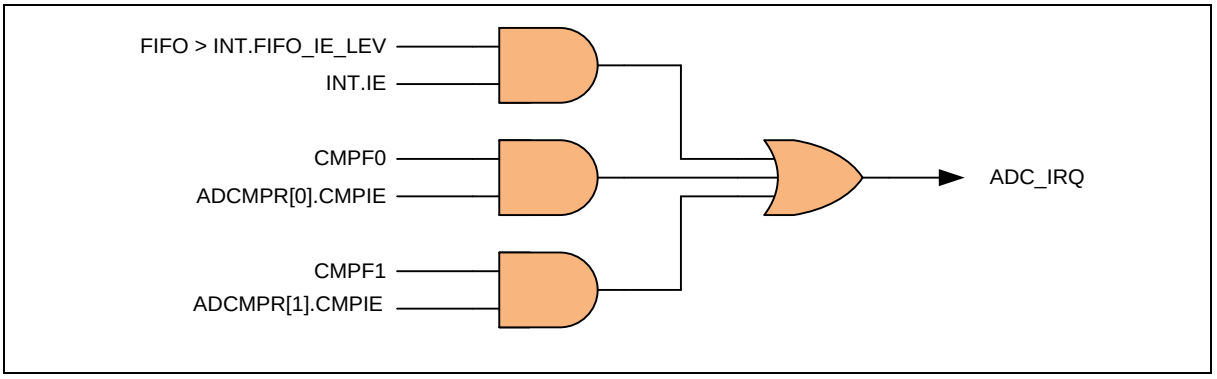


Figure 7-3 SDADC Controller Interrupt

7.1.4.4 *Peripheral DMA Request*

Normal use of the ADC 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.

7.1.5 ADC 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
ADC Base Address: **ADC_BA = 0x400E_0000**				
ADC_DAT	ADC_BA+0x00	R	ADC FIFO Data Out.	0x0000_XXXX
ADC_CHEN	ADC_BA+0x04	R/W	ADC Enable Register	0x0000_0000
ADC_CLKDIV	ADC_BA+0x08	R/W	ADC Clock Divider Register	0x0000_0000
ADC_DCICTL	ADC_BA+0x0C	R/W	ADC Decimation Control Register	0x0000_0000
ADC_INTCTL	ADC_BA+0x10	R/W	ADC Interrupt Control Register	0x0000_0000
ADC_PDMACTL	ADC_BA+0x14	R/W	ADC PDMA Control Register	0x0000_0000
ADC_CMP0	ADC_BA+0x18	R/W	ADC Comparator 0 Control Register	0x0000_0000
ADC_CMP1	ADC_BA+0x1C	R/W	ADC Comparator 1 Control Register	0x0000_0000

7.1.6 ADC Register Description

FIFO Audio Data Register (ADC_DAT)

Register	Offset	R/W	Description	Reset Value
ADC_DAT	ADC_BA+0x00	R	ADC FIFO Data Out.	0x0000_XXXX

Table 7-5 FIFO Audio Data Register (ADC_DAT, address 0x400E_0000)

Bits	Description	
[31:16]	Reserved	**Reserved**
[15:0]	**RESULT**	**ADC Audio 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 FIFOINTLV interrupt to determine if valid data is present in FIFO.

ADC Enable Register (ADC_CHEN)

Register	Offset	R/W	Description	Reset Value
ADC_CHEN	ADC_BA+0x04	R/W	ADC Enable Register	0x0000_0000

Table 7-6 ADC Enable Register (ADC_CHEN, address 0x400E_0004)

Bits	Description	
[31:1]	Reserved	Reserved
[0]	**CHEN**	**ADC Enable** 0 = Conversion stopped and ADC is reset including FIFO pointers. 1 = ADC Conversion enabled.

ADC Clock Division Register (ADC_CLKDIV)

Register	Offset	R/W	Description	Reset Value
ADC_CLKDIV	ADC_BA+0x08	R/W	ADC Clock Divider Register	0x0000_0000

Table 7-7 ADC Clock Divider Register (ADC_CLKDIV, address 0x400E_0008)

Bits	Description	
[31:8]	Reserved	Reserved
[7:0]	**CLKDIV**	**ADC Clock Divider** This register determines the clock division ration between the incoming ADC_CLK (aaa HCLK by default) and the Delta-Sigma sampling clock of the ADC. This together with the over-sampling ratio (OSR) determines the audio sample rate of the converter. CLKDIV should be set to give a SD_CLK frequency in the range of 1.024-6.144MHz. CLKDIV must be greater than 2. SD_CLK frequency aaa HCLK / CLKDIV

ADC Decimation Control Register (ADC_DCICTL)

Register	Offset	R/W	Description	Reset Value
ADC_DCICTL	ADC_BA+0x0C	R/W	ADC Decimation Control Register	0x0000_0000

Table 7-8 ADC Decimation Control Register (ADC_DCICTL, address 0x400E_000C)

Bits	Description	
[19:16]	**GAIN**	**CIC Filter Additional Gain** This should normally remain default 0. Can be set to non-zero values to provide additional digital gain from the decimation filter. An additional gain is applied to signal of GAIN/2.
[3:0]	**OVSPLRAT**	**Decimation Over-Sampling Ratio** This term determines the over-sampling ratio of the decimation filter. Valid values are: 0: OVSPLRAT aaa 64 1: OVSPLRAT aaa 128 2: OVSPLRAT aaa 192 3: OVSPLRAT aaa 384

ADC Interrupt Control Register (ADC_INTCTL)

Register	Offset	R/W	Description	Reset Value
ADC_INTCTL	ADC_BA+0x10	R/W	ADC Interrupt Control Register	0x0000_0000

Table 7-9 ADC Interrupt Control Register (ADC_INTCTL, address 0x400E_0010)

Bits	Description	
[31]	**INTEN**	**Interrupt Enable** If set to '1' an interrupt is generated whenever FIFO level exceeds that set in FIFOINTLV.
[2:0]	**FIFOINTLV**	**FIFO Interrupt Level** Determines at what level the ADC FIFO will generate a servicing interrupt to the CPU. Interrupt will be generated when number of words present in ADC FIFO is > FIFOINTLV.

ADC PDMA Control Register (ADC_PDMACTL)

Register	Offset	R/W	Description	Reset Value
ADC_PDMACTL	ADC_BA+0x14	R/W	ADC PDMA Control Register	0x0000_0000

Table 7-10 ADC PDMA Control Register (ADC_PDMACTL, address 0x400E_0014)

Bits	Description	
[0]	**RXDMAEN**	**Enable ADC PDMA Receive Channel** Enable ADC PDMA. If set, then ADC will request PDMA service when data is available.

A/D Compare Register 0(ADCMPR0)

Register	Offset	R/W	Description	Reset Value
ADC_CMP0	ADC_BA+0x18	R/W	ADC Comparator 0 Control Register	0x0000_0000

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

Table 7-11 ADC Comparator Control Registers (ADC_CMP0, address 0x400E_0018)

Bits	Description	
[31:16]	**CMPDAT**	**Comparison Data** 16 bit value to compare to FIFO output word.
[11:8]	**CMPMCNT**	**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 (CMPMCNT +1), the CMPFLAG bit will be set.
[7]	**CMPFLAG**	**Compare Flag** When the conversion result meets condition in ADCMPR0 this bit is set to 1. It is cleared by writing 1 to self.
[2]	**CMPCOND**	**Compare Condition** 0= Set the compare condition that result is less than CMPDAT 1= Set the compare condition that result is greater or equal to CMPDAT Note: When the internal counter reaches the value (CMPMCNT +1), the CMPFLAG bit will be set.
[1]	**ADCMPIE**	**Compare Interrupt Enable** 0 = Disable compare function interrupt. 1 = Enable compare function interrupt. If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMCNT, CMPFLAG bit will be asserted, if ADCMPIE is set to 1, a compare interrupt request is generated.
[0]	**ADCMPEN**	**Compare Enable** 0 = Disable compare. 1 = Enable compare. Set this bit to 1 to enable compare CMPDAT with FIFO data output.

A/D Compare Register 1 (ADCMPR1)

Register	Offset	R/W	Description	Reset Value
ADC_CMP1	ADC_BA+0x1C	R/W	ADC Comparator 1 Control Register	0x0000_0000

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

Table 7-12 ADC Comparator Control Registers (ADC_CMP1, address 0x400E_001C)

Bits	Description	
[31:16]	CMPDAT	**Comparison Data** 16 bit value to compare to FIFO output word.
[11:8]	CMPMCNT	**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 (CMPMCNT +1), the CMPFLAG bit will be set.
[7]	CMPFLAG	**Compare Flag** When the conversion result meets condition in ADCMPR0 this bit is set to 1. It is cleared by writing 1 to self.
[2]	CMPCOND	**Compare Condition** 0= Set the compare condition that result is less than CMPDAT 1= Set the compare condition that result is greater or equal to CMPDAT Note: When the internal counter reaches the value (CMPMCNT +1), the CMPFLAG bit will be set.
[1]	ADCMPIE	**Compare Interrupt Enable** 0 = Disable compare function interrupt. 1 = Enable compare function interrupt. If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMCNT, CMPFLAG bit will be asserted, if ADCMPIE is set to 1, a compare interrupt request is generated.
[0]	ADCMPEN	**Compare Enable** 0 = Disable compare. 1 = Enable compare. Set this bit to 1 to enable compare CMPDAT with FIFO data output.

7.2 Audio Class D Speaker Driver (DPWM)

7.2.1 Functional Description

The ISD9100 series includes a differential Class D (PWM) speaker driver capable of delivering 1W 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.

7.2.2 Features

- Differential Bridge-Tied-Load structure to directly drive 8Ω Speaker.
- Power delivery up to 1W @5V into 8Ω.
- Power efficiency of up to 85%.
- Configurable input sample rate.
- 16 Sample FIFO for audio output.
- PDMA data channel for streaming of PCM audio data.

7.2.3 Block Diagram

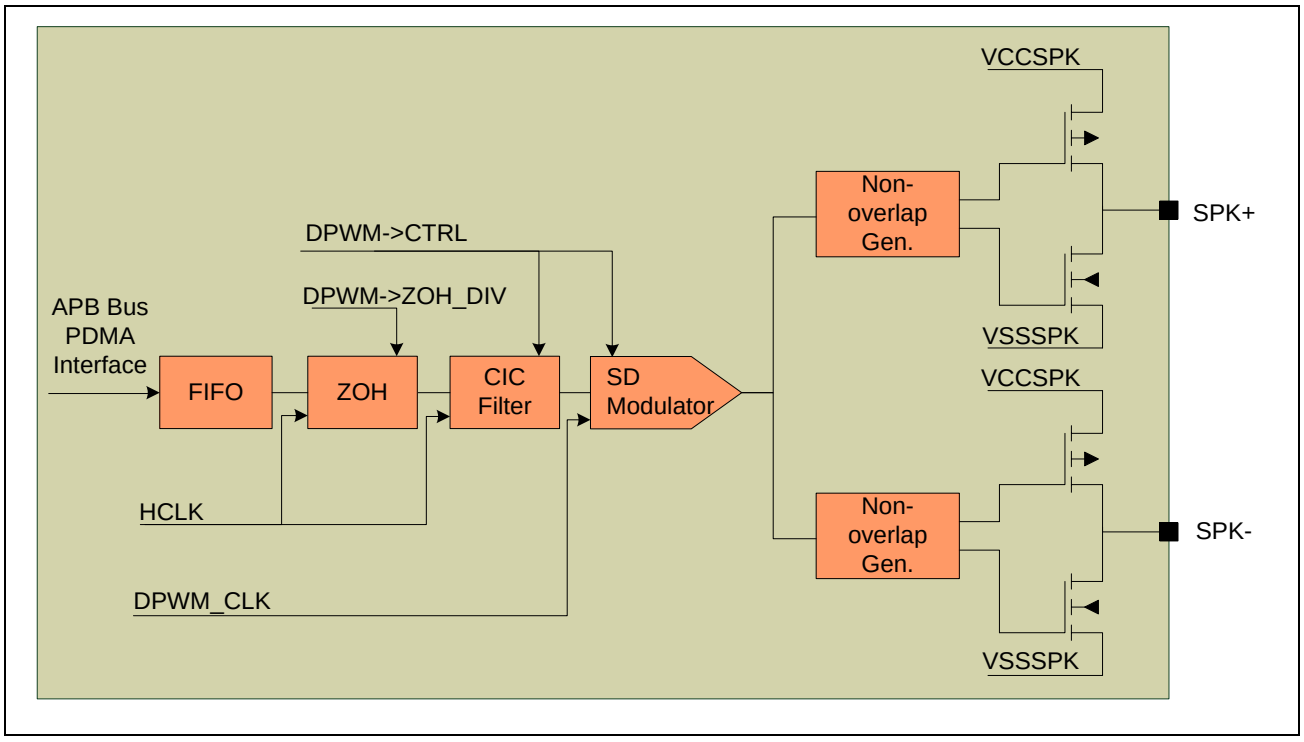


Figure 7-4 DPWM Block Diagram

7.2.4 Operation

The DPWM block receives audio data by writing 16bit 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 is generated by the internal oscillator (OSC48M) and operates at the frequency of OSC48M or 2x the frequency of OSC48M (See CLK_CLKSEL1 register Table 5-36). Ultimate SNR (Signal-to-Noise Ratio) is determined by the time resolution of the master clock.

7.2.4.1 *Determining Sample Rate*

The sample rate at which the DPWM block consumes audio data is given by:

$$F_s = HCLK \div ZOH\_DIV \div 64$$

Where HCLK is the master CPU clock rate and ZOHDIV is the divider control register. A table of common audio sample rates is provided below.

Table 7-13 DPWM Sample Rates for Various HCLK

HCLK (MHz)	ZOHDIV	Sample Rate (Hz)
49.152	24	32,000
49.152	48	16,000
49.152	96	8,000
32.768	16	32,000
32.768	32	16,000
32.768	64	8,000
24.576	12	32,000
24.576	24	16,000
24.576	48	8,000

7.2.4.2 *Configuring Speaker Driver*

To operate the speaker driver the following configuration is recommended:

- Enable DPWM clock source (CLK_APBCLK0.DPWMCKEN Table 5-33, CLK_CLKSEL1.DPWMCKSEL Table 5-36).
- Reset DPWM IP block. (SYS_IPRST1.DPWM_RST Table 5-4)
- Select sample rate based on current HCLK frequency.
- Setup PDMA channel to provide data to DPWM.
- Enable PDMA Request.
- Enable Driver.

7.2.4.3 *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.

7.2.5 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 = 0x4007_0000**				
DPWM_CTL	DPWM_BA+0x00	R/W	DPWM Control Register	0x0000_0000
DPWM_STS	DPWM_BA+0x04	R	DPWM 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 FIFO Input	0x0000_0000
DPWM_ZOHDIV	DPWM_BA+0x10	R/W	DPWM Zero Order Hold Division Register	0x0000_0030

7.2.6 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

7	6	5	4	3	2	1	0
Reserved	**DPWMEN**	**DITHEREN**		**DEADTIME**	**MODUFRQ**		

Table 7-14 DPWM Control Register (DPWM_CTL, address 0x4007_0000)

Bits	Description	
[6]	**DPWMEN**	**DPWM Enable** 0= Disable DPWM, SPK pins are tri-state, CIC filter is reset, FIFO pointers are reset (FIFO data is not reset). 1= Enable DPWM, SPK pins are enabled and driven, data is taken from FIFO.
[5:4]	**DITHEREN**	**DPWM Signal Dither Control** To prevent structured noise on PWM output due to DC offsets in the input signal it is possible to add random dither to the PWM signal. These bits control the dither: 0 = No dither. 1 = +/- 1 bit dither 3 = +/- 2 bit dither
[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.
[2:0]	**MODUFRQ**	**DPWM Modulation Frequency** This parameter controls the carrier modulation frequency of the PWM signal as a proportion of DPWM_CLK. MODUFRQ : DPWM_CLK Division : Frequency for DPWM_CLK aaa 98.304MHZ 0 : 228 : 431158 1 : 156 : 630154 2 : 76 : 1293474 3 : 52 : 1890462 4 : 780 : 126031 5 : 524 : 187603 6 : 396 : 248242 7 : 268 : 366806

DPWM FIFO Status Register (DPWM_STS)

Register	Offset	R/W	Description	Reset Value
DPWM_STS	DPWM_BA+0x04	R	DPWM FIFO Status Register	0x0000_0002

Table 7-15 DPWM FIFO Status Register (DPWM_STS, address 0x4007_0004)

Bits	Description	
[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

Table 7-16 DPWM PDMA Control Register (DPWM_DMACTL, address 0x4007_0008)

Bits	Description	
[31:8]	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 FIFO Input	0x0000_0000

Table 7-17 DPWM FIFO Input (DPWM_DATA, address 0x4007_000C)

Bits	Description	
[15:0]	**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.

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_0030

Table 7-18 DPWM Zero Order Hold Division Register (DPWM_ZOHDIV, address 0x4007_0010)

Bits	Description	
[7:0]	**ZOHDIV**	**DPWM Zero Order Hold, Down-Sampling Divisor** The input sample rate of the DPWM is set by HCLK frequency and the divisor set in this register by the following formula: Fs aaa HCLK/ZOHDIV/64 Valid range is 1 to 255. Default is 48, which gives a sample rate of 16kHz for a 49.152MHz (default) HCLK.

7.3 Analog Comparator

7.3.1 Functional Description

ISD9100 series contains two analog comparators. The comparator output is a logical one when positive input greater than negative input, otherwise the output is a zero. Each comparator can be configured to cause an interrupt when the comparator output value changes. The block diagram is shown in Figure 7-5.

Note that the analog input port pins must be configured as input type or analog alternate function before Analog Comparator function is enabled.

7.3.2 Features

- Analog input voltage range: 0~5.0V
- Comparator 0 multiplexed to all analog enabled GPIO (GPIOB[7:0]).
- Comparator 0 can compare against VBG or VMID.
- Comparator 1 can compare GPIOB[7] to GPIOB[6] or VBG.
- Single comparator interrupt requested by either comparator.
- Can be used in conjunction with Capacitive Touch Sensing block for capacitive touch sensing.

7.3.3 Block Diagram

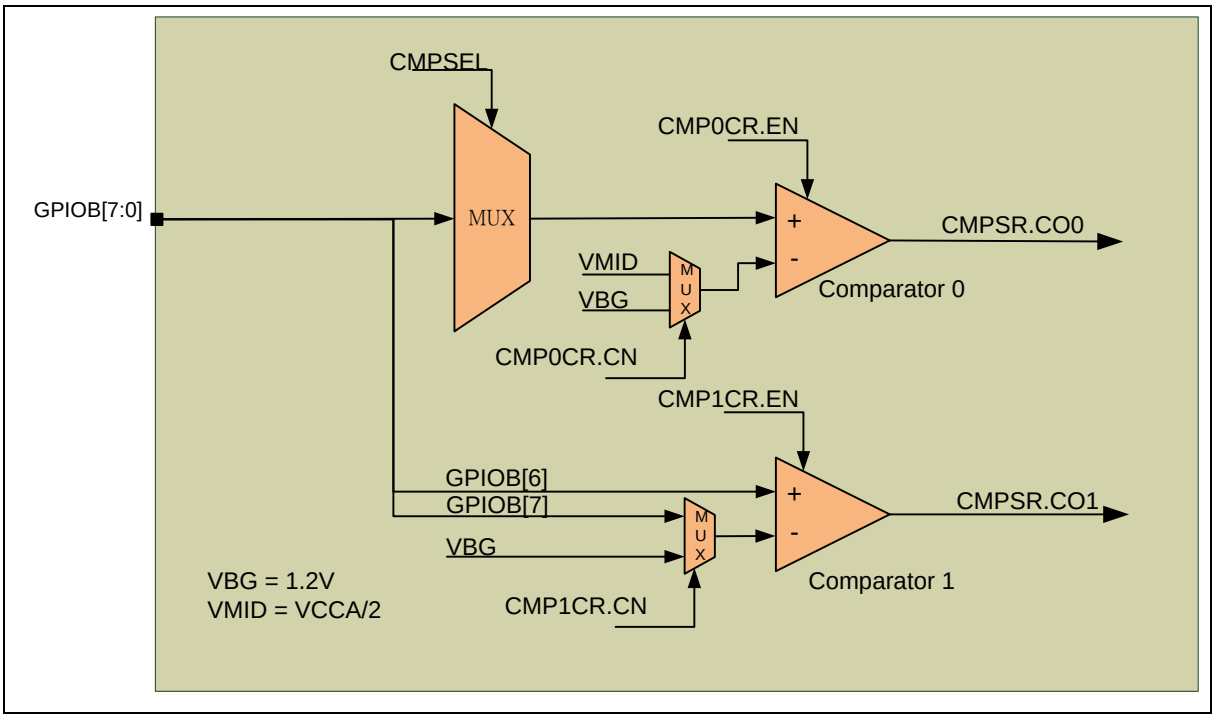


Figure 7-5 Analog Comparator Block Diagram

7.3.4 Operational Procedure

Setup Procedure

To use the Analog Comparator block, use the following sequence:

1. Configure GPIO for use as analog input by setting type to input.
2. Enable the peripheral clock (CLK_APBCLK0.ACMPCKEN)
3. Reset the Comparator block (SYS_IPRST1.ACMPRST, Table 5-4)
4. If using VMID ensure that VMID block is powered up (Section 7.4.4)
5. Select comparison sources with CMPnCR and ACMP_POSSEL.
6. Enable comparators and appropriate interrupts with CMPnCR.
7. Enables system interrupt if appropriate (e.g. NVIC_EnableIRQ(ACMP_IRQn);)

Interrupt Sources

The comparator generates an output CO*n* (n=0,1) which is reported in ACMP_STATUS register. If CMP*n*CR.IE bit is set then a state change on the comparator output CO*n* will cause comparator flag CMPF*n* to go high and the comparator interrupt is requested. Software can write a one to CMPF*n* to clear flag and interrupt request.

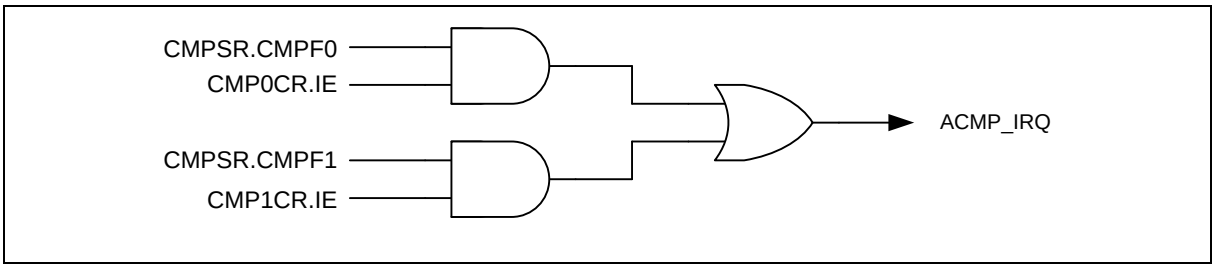


Figure 7-6 Comparator Controller Interrupt Sources

7.3.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
ACMP Base Address: **ACMP_BA = 0x400D_0000**				
ACMP_CTL0	ACMP_BA+0x00	R/W	Analog Comparator 0 Control Register	0x0000_0000
ACMP_CTL1	ACMP_BA+0x04	R/W	Analog Comparator 1 Control Register	0x0000_0000
ACMP_STATUS	ACMP_BA+0x08	R/W	Comparator Status Register	0x0000_00XX
ACMP_POSSEL	ACMP_BA+0x0C	R/W	Comparator Select Register	0x0000_0000

7.3.6 Register Description

Comparator 0 Control Register (ACMP_CTL0)

Register	Offset	R/W	Description	Reset Value
ACMP_CTL0	ACMP_BA+0x00	R/W	Analog Comparator 0 Control Register	0x0000_0000

7	6	5	4	3	2	1	0
-	-	-	**NEGSEL**	-		**ACMPIE**	**ACMPEN**

Table 7-19 Comparator 0 Control Register (ACMP_CTL0, address 0x400D_0000).

Bits	Description	
[4]	**NEGSEL**	**Comparator0 Negative Input Select** 0 = VBG, Bandgap reference voltage aaa 1.2V 1 = VMID reference voltage aaa VCCA/2
[1]	**ACMPIE**	**CMP0 Interrupt Enable** 0 = Disable CMP0 interrupt function 1 = Enable CMP0 interrupt function
[0]	**ACMPEN**	**Comparator Enable** 0 = Disable 1 = Enable

Comparator 1 Control Register (ACMP_CTL1)

Register	Offset	R/W	Description	Reset Value
ACMP_CTL1	ACMP_BA+0x04	R/W	Analog Comparator 1 Control Register	0x0000_0000

7	6	5	4	3	2	1	0
-	-	-	**NEGSEL**	-		**ACMPIE**	**ACMPEN**

Table 7-20 Comparator 1 Control Register (ACMP_CTL1, address 0x400D_0004).

Bits	Description	
[4]	**NEGSEL**	**Comparator1 Negative Input Select** 0 = GPIOB[7] 1 = VBG, Bandgap reference voltage aaa 1.2V
[1]	**ACMPIE**	**CMP1 Interrupt Enable** 0 = Disable CMP1interrupt function 1 = Enable CMP1 interrupt function
[0]	**ACMPEN**	**Comparator Enable** 0 = Disable 1 = Enable

CMP Status Register (ACMP_STATUS)

Register	Offset	R/W	Description	Reset Value
ACMP_STATUS	ACMP_BA+0x08	R/W	Comparator Status Register	0x0000_00XX

7	6	5	4	3	2	1	0
Reserved				ACMPO1	ACMPO0	ACMPIF1	ACMPIF0

Table 7-21 CMP Status Register (ACMP_STATUS, address 0x400D_0008).

Bits	Description	
[3]	ACMPO1	**Comparator1 Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (CMP1EN aaa 0).
[2]	ACMPO0	**Comparator0 Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (CMP0EN aaa 0).
[1]	ACMPIF1	**Compare 1 Flag** This bit is set by hardware whenever the comparator output changes state. This bit will cause a hardware interrupt if enabled. This bit is cleared by writing 1 to itself.
[0]	ACMPIF0	**Compare 0 Flag** This bit is set by hardware whenever the comparator output changes state. This bit will cause a hardware interrupt if enabled. This bit is cleared by writing 1 to itself.

CMP Select Register (ACMP_POSSEL)

Register	Offset	R/W	Description	Reset Value
ACMP_POSSEL	ACMP_BA+0x0C	R/W	Comparator Select Register	0x0000_0000

Table 7-22 CMP Select Register (ACMP_POSSEL, address 0x400D_000C).

Bits	Description	
[2:0]	**POSSEL**	**Comparator0 GPIO Selection** GPIOB[POSSEL] is the active analog GPIO input selected to Comparator 0 positive input.

7.4 Analog Functional Blocks

7.4.1 Overview

The ISD9100 series 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.

7.4.2 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.
- Microphone Bias generator.
- Analog Multiplexor.
- Programmable Gain Amplifier (PGA).
- OSC48M Frequency Control.
- Capacitive Touch Sensing Relaxation Oscillator.
- Oscillator Frequency Measurement block.

7.4.3 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_CURCTL0	ANA_BA+0x08	R/W	Current Source Control Register	0x0000_0000
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 Select Register	0x0000_0000
ANA_MICBEN	ANA_BA+0x2C	R/W	Microphone Bias Enable Register	0x0000_0000
ANA_MUXCTL	ANA_BA+0x50	R/W	Analog Multiplexer Control Register	0x0000_0000
ANA_PGACTL	ANA_BA+0x60	R/W	PGA Enable Register	0x0000_0000
ANA_SIGCTL	ANA_BA+0x64	R/W	Signal Path Control Register	0x0000_0000
ANA_PGAGAIN	ANA_BA+0x68	R/W	PGA Gain Select Register	0x0000_0010
ANA_TRIM	ANA_BA+0x84	R/W	Oscillator Trim Register	0x0000_XXXX
ANA_CAPSCTL	ANA_BA+0x8C	R/W	Capacitive Touch Sensing Control Register	0x0000_0000
ANA_CAPSCNT	ANA_BA+0x90	R	Capacitive Touch Sensing Count Register	0x0000_0000

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

7.4.4 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 Figure 7-7.

Before using the ADC, 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.

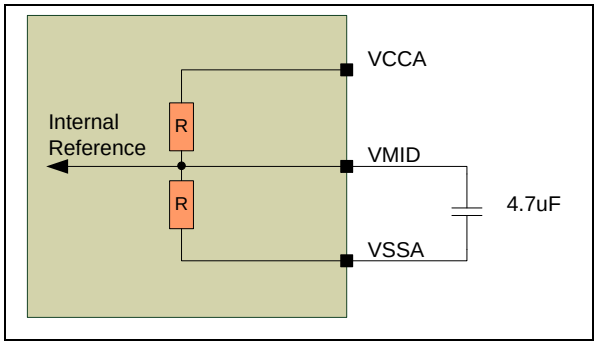


Figure 7-7 VMID Reference Generation

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

7	6	5	4	3	2	1	0
Reserved					**PDHIRES**	**PDLORES**	**PULLDOWN**

Table 7-23 VMID Control Register (ANA_VMID, address 0x4008_0000).

Bits	Description	
[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.

7.4.5 GPIO Current Source Generation

The GPIOB port consists of analog enabled GPIO. One of the features of these pins is the ability to route a current source to the pin. This is useful for a variety of purposes such as providing a current load to a sensor such as a photo-transistor or CDS cell. It can also be used to do capacitive touch sensing in combination with the relaxation oscillator control circuit.

The current generation block consists of a programmable current source controlled by ANA_CURCTL0.VALSEL and individual switches to each of the GPIOB pins as shown in Figure 7-8. Power control for this block is merged with the analog comparator, this block must be enabled to use current source (ACMP_CTL0.ACMPEN=1).

Analog peripheral clock must be enabled before register can be written. At least one of the analog comparators must be enabled to enable current source.

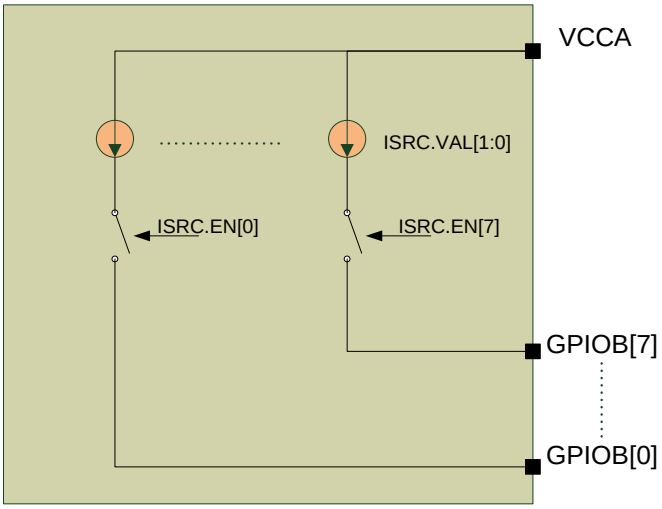


Figure 7-8 GPIOB Current Source Generation

Current Source Control Register (ANA_CURCTL0)

Register	Offset	R/W	Description	Reset Value
ANA_CURCTL0	ANA_BA+0x08	R/W	Current Source Control Register	0x0000_0000

15	14	13	12	11	10	9	8
Reserved						**VALSEL**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CURSRCEN							

Table 7-24 GPIO Current Source Control Register (ANA_CURCTL0, address 0x4008_0008).

Bits	Description	
[31:10]	Reserved	Reserved
[9:8]	**VALSEL**	**Current Source Value** Select master current for source generation 0= 0.5 uA 1= 1 uA 2= 2.5 uA 3= 5 uA
[7:0]	**CURSRCEN**	**Enable Current Source to GPIOB[x]** Individually enable current source to GPIOB pins. Each GPIOB pin has a separate current source. 0 = Disable 1 = Enable current source to pin GPIOB[x]

7.4.6 LDO Power Domain Control

The ISD9100 series 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 2.4-3.3V regulated supply to operate the GPIOA[7:0] domain and external loads up to 30mA. The supply pin for the LDO is the VCCLDO pin which should be connected to VCCD. If the LDO is not used, both VCCLDO 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).

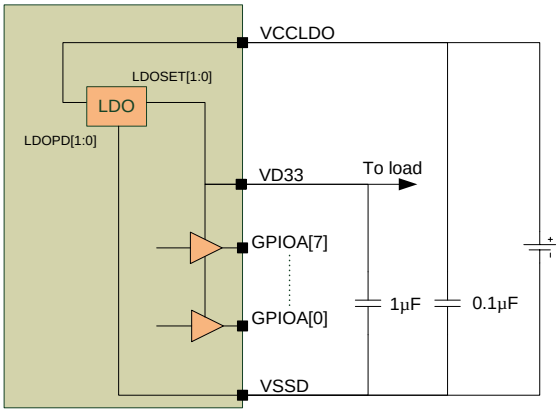


Figure 7-9 LDO Power Domain

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

7	6	5	4	3	2	1	0
Reserved						**LDOSEL**	

Table 7-25 LDO Voltage Control Register (ANA_LDOSEL, address 0x4008_0020).

Bits	Description	
[31:2]	Reserved	Reserved
[1:0]	**LDOSEL**	**Select LDO Output Voltage** Note that maximum I/O pad operation speed only specified for voltage >2.4V. 0= 3.0V 1= 1.8V 2= 2.4V 3= 3.3V

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

7	6	5	4	3	2	1	0
Reserved						**DISCHAR**	**PD**

Table 7-26 LDO Power Down Control Register (ANA_LDOPD, address 0x4008_0024).

Bits	Description	
[1]	**DISCHAR**	**Discharge** 0 = No load on 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.

7.4.7 Microphone Bias Generator

The ISD9100 series provides a microphone bias generator (MICBIAS) for improved recording quality. The MICBIAS can provide a maximum current of 1mA with a -60dB power supply rejection. The MICBIAS output voltage can be configured with ANA_MICBSEL[1:0] to select bias voltages from 50% to 90% of the VCCA supply voltage (see description below). The user should consider the microphone manufacturers specification in deciding on the optimum MICBIAS voltage to use. Generally, a microphone will require a current of 0.1mA to a maximum 0.5mA and a voltage of 1V to 3V across it.

Referring to the application diagram of Figure 7-11, external resistor R_1 and R_2 values are selected to limit the current to a maximum that can be provided by MICBIAS; 1mA. On the ISD9100 series, the minimum total resistance $(R_1 + R_2)$ is 4Kohms. MICBIAS output voltage should be such that the following condition is met:

$$V_{MICBIAS} > V_S + (R_1 + R_2) \times I_{MIC}$$

where V_S is the desired voltage across the microphone from specification and I_{MIC} is the current through the microphone (0.1-0.5mA)

From Figure 7-11, MIC_IN1 and MIC_IN2 are AC coupled to the ISD9100 series MIC+ and MIC- respectively for differential inputs. In single-ended operation, MIC_IN1 should go to MIC- of the ISD9100 series. C_1 and C_2 are AC coupling capacitors. In single-ended application, R_2 can be removed and R_1 increased to at least 4Kohms. For improved performance, it is recommended to keep R_2 to provide additional rejection from ground noise.

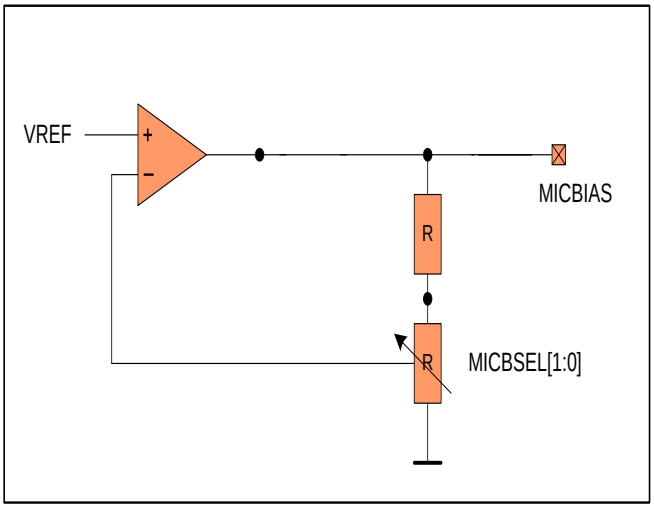


Figure 7-10 MICBIAS Block Diagram

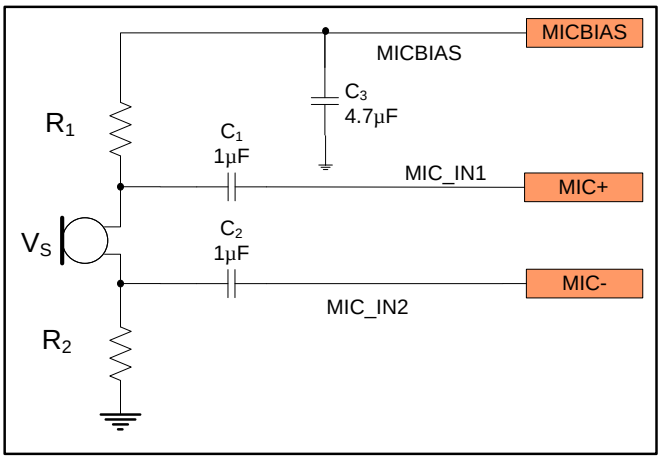


Figure 7-11 MICBIAS Application Diagram

Microphone Bias Select (ANA_MICBSEL)

Register	Offset	R/W	Description	Reset Value
ANA_MICBSEL	ANA_BA+0x28	R/W	Microphone Bias 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					**REFSEL**	**VOLSEL**	

Table 7-27 Microphone Bias Selection Register (ANA_MICBSEL, address 0x4008_0028).

Bits	Description	
[31:3]	**Reserved**	Reserved
[2]	**REFSEL**	**Select Reference Source For MICBIAS Generator** VMID provides superior noise performance for MICBIAS generation and should be used unless fixed voltage is absolutely necessary, then noise performance can be sacrificed and bandgap voltage used as reference. 0= VMID aaa VCCA/2 is reference source. 1= VBG (bandgap voltage reference) is reference source.
[1:0]	**VOLSEL**	**Select Microphone Bias Voltage** MICBMODE aaa 0 0: 90% VCCA 1: 65% VCCA 2: 75% VCCA 3: 50% VCCA MICBMODE aaa 1 0: 2.4V 1: 1.7V 2: 2.0V 3: 1.3V

Microphone Bias Enable Register (ANA_MICBEN)

Register	Offset	R/W	Description	Reset Value
ANA_MICBEN	ANA_BA+0x2C	R/W	Microphone Bias Enable Register	0x0000_0000

7	6	5	4	3	2	1	0
Reserved							**MUXEN**

Table 7-28 Microphone Bias Enable Register (ANA_MICBEN, address 0x4008_002C)

Bits	Description	
[0]	**MICBEN**	**Enable Microphone Bias Generator** 0 = Powered Down. 1 = Enabled.

7.4.8 Analog Multiplexer

The ISD9100 series provides an analog multiplexer (AMUX) which allows the PGA input to be switched from the dedicated MICP/MICN analog inputs to any of the analog enabled GPIO (GPIOB[7:0]). The negative input of the PGA connects to GPIOB[7:0], while the positive PGA input connects to the odd numbered GPIOB[7:1]. Figure 7-12 shows the multiplexer block diagram and Table 7-29 shows the multiplexer control signals.

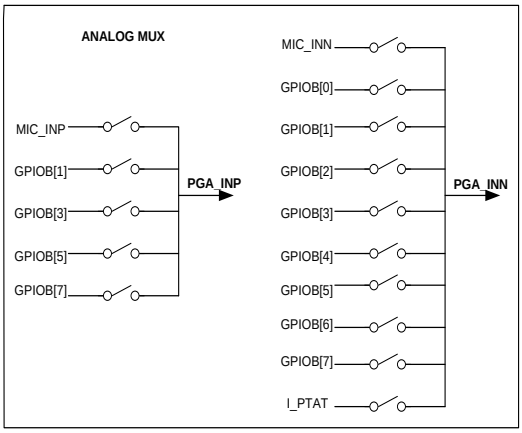


Figure 7-12 Analog Multiplexer Block Diagram

Analog Multiplexer Control Register (ANA_MUXCTL)

Register	Offset	R/W	Description	Reset Value
ANA_MUXCTL	ANA_BA+0x50	R/W	Analog Multiplexer Control Register	0x0000_0000

15	14	13	12	11	10	9	8
Reserved	**MUXEN**	**PGAINSEL**	**PTATCUR**	**POSINSEL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NEGINSEL							

Table 7-29 Analog Multiplexer Control Register (ANA_MUXCTL, address 0x4008_0050).

Bits	Description	
[31:15]	**Reserved**	Reserved
[14]	**MUXEN**	**Enable The Analog Multiplexer** 0 = All channels disabled 1 = Selection determined by register setting.
[13]	**PGAINSEL**	**Select MICP/MICN To PGA Inputs**
[12]	**PTATCUR**	**Select PTAT Current** I_PTAT, to PGA_INN, negative input to PGA, for temperature measurement.
[11:8]	**POSINSEL**	**Selects Connection Of GPIOB[7,5,3,1] To PGA_INP, Positive Input Of PGA** 1000b: GPIOB[7] connected to PGA_INP 0100b: GPIOB[5] connected to PGA_INP 0010b: GPIOB[3] connected to PGA_INP 0001b: GPIOB[1] connected to PGA_INP
[7:0]	**NEGINSEL**	**Selects Connection Of GPIOB[7:0] To PGA_INN, Negative Input Of PGA** If NEGINSEL[*n*] aaa 1 then GPIOB[*n*] is connected to PGA_INN.

Temperature Sensor Measurement

In addition, the multiplexer can route a PTC (positive temperature coefficient) current, PTAT current, to the ADC to perform temperature measurements. To configure the signal path to do temperature measurement, configure the ADC path as follows:

1) Enable the multiplexer, PGA, IPBOOST, and sigma-delta modulator. (See Section 7.4.9, Section 7.1).
2) Have the multiplexer select I_PTAT current as input and choose VBG (bandgap voltage) as reference (REFSEL).
3) Set the 6-bit ANA_PGAGAIN[5:0] gain value to hex 0x17 and choose 0dB gain setting for IPBOOST gain block.
4) The temperature can be inferred by the information given in Table 7-30 and equation below.

$$T\ (°C) = 27 + (ADC\_VAL - 0x42EA)/50. \quad \text{(Equation 7-1)}$$

The settings corresponding to this configuration are:

ANA_SIGCTL=0x1E, ANA_PGACTL=0x07, ANA_MUXCTL=0x5000, ANA_PGAGAIN=0x17

Table 7-30 Temperature Sensor Measurement.

Parameter	Specification (Reference)				Test Condition
	Min.	Typ.	Max.	Unit	
Temperature Sensor Output		0x42EA		Code	At 27° C
Temperature Sensor Delta Coefficient (number of bits per degree °C)**		50		LSB/°C	Relative to 27°C

**LSB is the least significant bit of a 16-bit ADC with a defined full-scale RMS input voltage of 0.77V

7.4.9 Programmable Gain Amplifier

The ISD9100 series provides a Programmable Gain Amplifier (PGA) as the front-end to the ADC to allow the adjustment of signal path gain. It is used in conjunction with the ALC block to provide automatic level control of incoming audio signals. Figure 7-13 shows the signal path diagram. The PGA provides a gain from -12dB to 35.25dB in increments of 0.75dB steps using a 6-bit control, ANA_PGAGAIN[5:0]. The gain is monotonically increasing with 0x00 for lowest gain (-12dB) and 0x3f for the maximum gain (35.25dB). The signal path is enabled by powering up the gain elements (PUPGA, PUBOOST). The PGA and IP BOOST blocks can be muted with the ANA_SIGCTL register. Input to the PGA can be either differential or single-ended on the PGA_INN input. The Analog MUX controls connection of the signal path to external pins. PGA input impedance varies based on the gain setting. Table 7-31 shows a table of input impedance for different gain setting.

The IP BOOST block can provide 0dB or 26dB of gain to provide a maximum gain of 61dB in the signal path. Front-end anti-alias filtering for the sigma-delta ADC is also provided by PGA/IP-BOOST blocks with an attenuation of -45dB at 6MHz frequency. The signal path defaults to have VCCA/2 as the reference voltage.

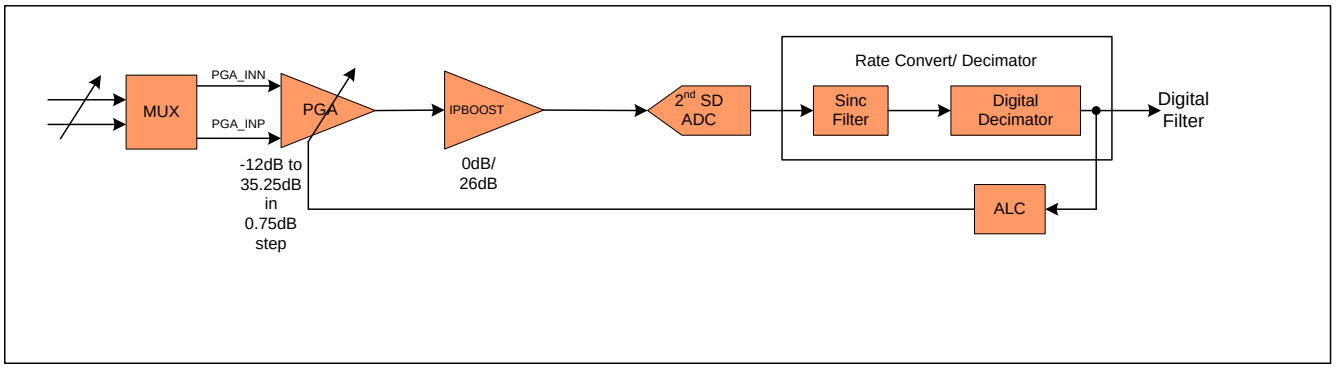


Figure 7-13 PGA Signal Path Block Diagram

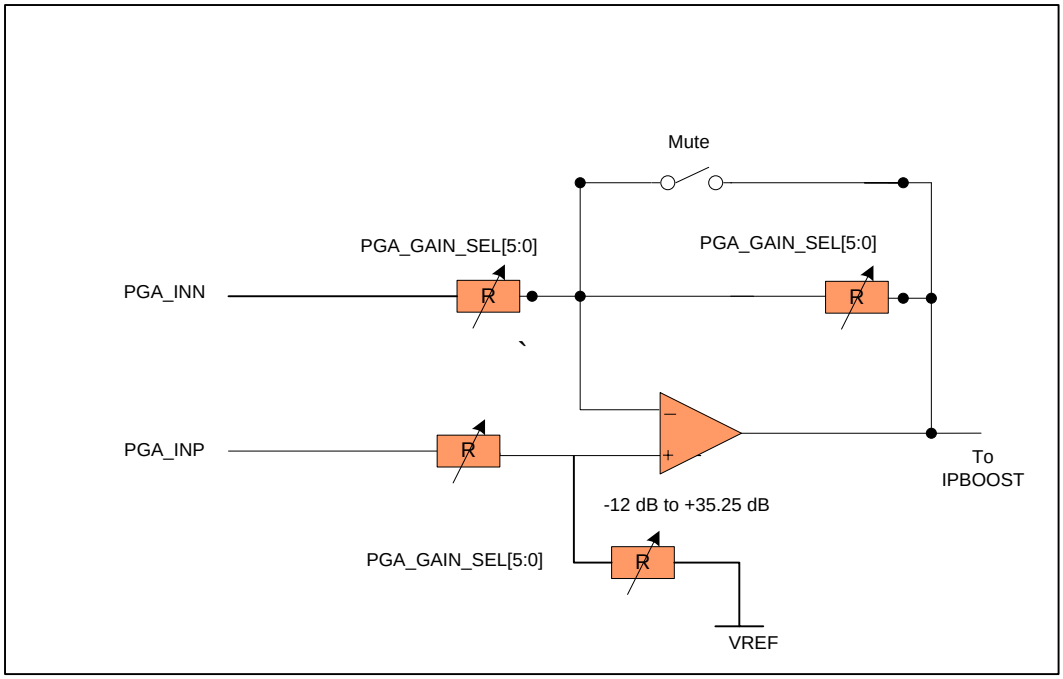


Figure 7-14 PGA Structure

Table 7-31 PGA Input Impedance Variation with Gain Setting

Gain (dB)	-12	-9	-6	-3	0	3	6	9	12	18	30	35.2
MICN Impedance (kΩ)	75	69	63	55	47	35	31	25	19	11	2.9	1.6
MICP Impedance (kΩ)	94	94	94	94	94	94	94	94	94	94	94	94

PGA Enable Register (ANA_PGACTL)

Register	Offset	R/W	Description	Reset Value
ANA_PGACTL	ANA_BA+0x60	R/W	PGA Enable Register	0x0000_0000

7	6	5	4	3	2	1	0
Reserved				**BSTGAIN**	**PUBOOST**	**PUPGA**	**REFSEL**

Table 7-32 PGA Enable and Control Register (ANA_PGACTL, address 0x4008_0060)

Bits	Description	
[3]	**BSTGAIN**	**Boost Stage Gain Setting** 0 = Gain aaa 0dB. 1 = Gain aaa 26dB
[2]	**PUBOOST**	**Power Up Control For Boost Stage Amplifier** This amplifier must be powered up for signal path operation. 0 = Power Down. 1 = Power up.
[1]	**PUPGA**	**Power Up Control For PGA Amplifier** This amplifier must be powered up for signal path operation. 0 = Power Down. 1 = Power up.
[0]	**REFSEL**	**Select Reference For Analog Path** Signal path is normally referenced to VMID (VCCA/2). To use an absolute reference this can be set to VBG aaa 1.2V. 0 = Select VMID voltage as analog ground reference. 1 = Select Bandgap voltage as analog ground reference.

Signal Path Control Register (ANA_SIGCTL)

Register	Offset	R/W	Description	Reset Value
ANA_SIGCTL	ANA_BA+0x64	R/W	Signal Path Control Register	0x0000_0000

7	6	5	4	3	2	1	0
Reserved	MUTEBST	MUTEPGA	PUADCOP	PUCURB	PUBUFADC	PUBUFPGA	PUZCDCMP

Table 7-33 Signal Path Mute Control Register (**ANA_SIGCTL**, address 0x4008_0064)

Bits	Description	
[6]	MUTEBST	**Boost Stage Mute Control** 0 = Normal. 1 = Signal Muted.
[5]	MUTEPGA	**PGA Mute Control** 0 = Normal. 1 = Signal Muted.
[4]	PUADCOP	**Power Up ADC ΣΔ Modulator** This block must be powered up for ADC operation. 0 = Power down. 1 = Power up.
[3]	PUCURB	**Power Up Control For Current Bias Generation** This block must be powered up for signal path operation. 0 = Power down. 1 = Power up.
[2]	PUBUFADC	**Power Up Control For ADC Reference Buffer** This block must be powered up for signal path operation. 0 = Power down. 1 = Power up.
[1]	PUBUFPGA	**Power Up Control For PGA Reference Buffer** This block must be powered up for signal path operation. 0 = Power down. 1 = Power up.
[0]	PUZCDCMP	**Power Up And Enable Control For Zero Cross Detect Comparator** When enabled PGA gain settings will only be updated when ADC input signal crosses zero signal threshold. To operate ZCD the ALC peripheral clock (CLK_APBCLK0.BFALCKEN) must also be enabled and BIQ_CTL.DLCOEFF aaa 1 to allow ZCD clocks to be generated. 0 = Power down. 1 = Power up and enable zero cross detection.

PGA GAIN Control Register (ANA_PGAGAIN)

Register	Offset	R/W	Description	Reset Value
ANA_PGAGAIN	ANA_BA+0x68	R/W	PGA Gain Select 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		GAINREAD					
7	6	5	4	3	2	1	0
Reserved		GAINSET					

Table 7-34 PGA Gain Control Register (ANA_PGAGAIN, address 0x4008_0068)

Bits	Description	
[13:8]	GAINREAD	**Current PGA Gain** Read Only. May be different from GAIN register when AGC is enabled and is controlling the PGA gain.
[5:0]	GAINSET	**Select 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.

7.4.10 Capacitive Touch Sensing Relaxation Oscillator/Counter

The ISD9100 series provides a functional unit that is used with analog GPIO functions to form a relaxation oscillator. The major application of this function is to measure the capacitive load on a GPIO pin. This measurement allows the user to implement a capacitive touch sensing scheme. With appropriate touch sensor design, the capacitance of the sensor will change appreciably in the presence of a finger, and the Capacitive Touch Sensing Relaxation Oscillator can measure this.

This block us used in conjunction with the analog comparator block and current source block to form a relaxation oscillator and counter circuit that can sense capacitance changes. A block diagram of the system is shown in Figure 7-15.

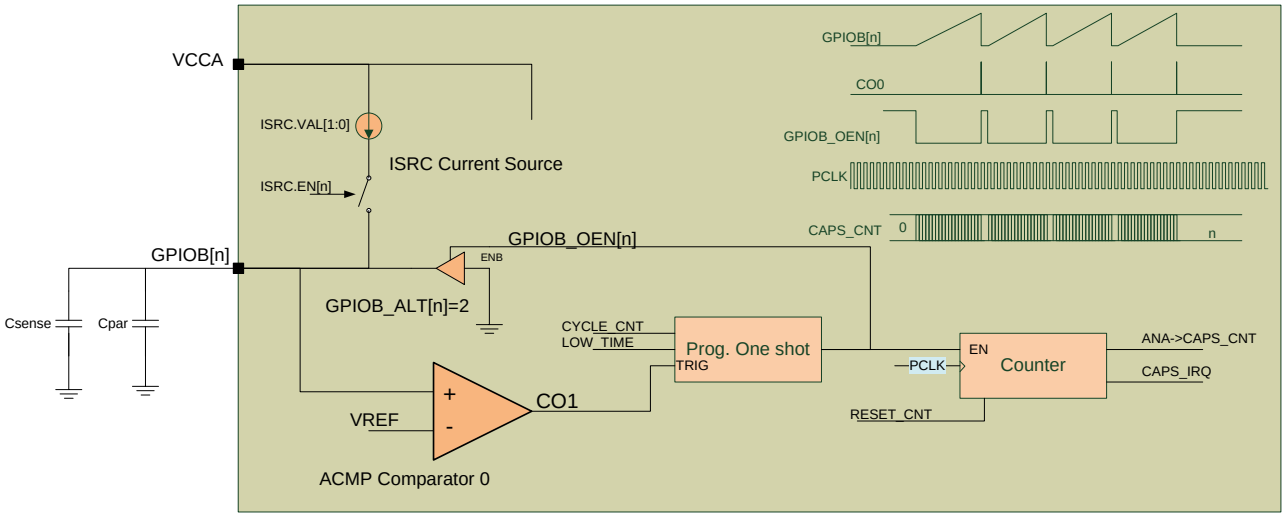


Figure 7-15 Capacitive Touch Sensing Function Block Diagram

7.4.10.1 *Functional Description*

The principle behind the operation of this block is that a certain capacitance is present on one of the analog enabled GPIO (GPIOB[7:0]). This capacitance consists of a certain parasitic capacitance $C_{par}$and the capacitor that is to be sensed C_{sense}. The GPIO is configured into the Capacitive Touch Sensing mode by setting SYS_GPB_MFP.GPBn = 2 and enabling a current source to this pin (ANA_CURCTL0.CURSRCEN = 2^n). The Analog Comparator 0 is also setup to compare the voltage at the pin to a reference voltage (ACMP_POSSEL = n, ACMP_CTL0.ACMPEN = 1).

In this configuration the circuit will charge the total capacitance with current ANA_CURCTL0.VALSEL = 0.5µA-5µA. When the voltage reaches the reference voltage (normally set to VBG=1.2V), the Capacitive Touch Sensing block will reset the GPIO pin to 0V. The circuit can be configured to do this 2^CYCLECNT times before generating an interrupt. While the capacitor is charging, a 24bit counter is also enabled such that the total charge time is recorded. After completion of 2^CYCLES_CNT cycles the software can read the ANA_CAPSCNT register to get a value proportional to the total capacitance on the pin. Once this is done, the count can be reset with RSTCNT and a new measurement started either on the same GPIO or selecting a different GPIO.

7.4.10.2 *Design Considerations*

Selecting parameters for capacitive touch sensing measurement is a trade-off between speed and accuracy/noise immunity. The higher the current source setting, the faster the oscillation but lower the resolution. The higher the cycle count the slower the measurement but the higher the accuracy and noise immunity.

7.4.10.3 *Register Descriptions*

Capacitive Touch Sensing Control Register (ANA_CAPSCTL)

Register	Offset	R/W	Description	Reset Value
ANA_CAPSCTL	ANA_BA+0x8C	R/W	Capacitive Touch Sensing Control Register	0x0000_0000

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

Table 7-35 Capacitive Touch Sensing Control Register (ANA_CAPSCTL, address 0x4008_008C).

Bits	Description	
[31]	CAPSEN	**Enable** 0 = Disable/Reset block. 1 = Enable Block.
[30]	INTEN	**Interrupt Enable** 0 = Disable/Reset CAPS_IRQ interrupt. 1 = Enable CAPS_IRQ interrupt.
[29]	RSTCNT	**Reset Count** 0: Release/Activate ANA_CAPSCNT 1: Set high to reset ANA_CAPSCNT.
[15:8]	CLKDIV	**Reference Clock Divider** Circuit can be used to generate a reference clock output of SDCLK/2/(CLKDIV+1) instead of a Capacitive Touch Sensing reset signal.
[5]	CLKMODE	**Reference Clock Mode** 0 = Capacitive Touch Sensing Mode. 1 = Circuit is in Reference clock generation mode.
[4:2]	CYCLECNT	**Number of Relaxation Cycles** Peripheral performs 2^(CYCLECNT) relaxation cycles before generating interrupt.

[1:0]	**LOWTIME**	**Output Low Time** Number of PCLK cycles to discharge external capacitor. 0=1cycle 1=2cycles 2=8cycles 3=16cycles

Capacitive Touch Sensing Count Register (ANA_CAPSCNT)

Register	Offset	R/W	Description	Reset Value
ANA_CAPSCNT	ANA_BA+0x90	R	Capacitive Touch Sensing Count Register	0x0000_0000

Table 7-36 Capacitive Touch Sensing Count Register (ANA_CAPSCNT, address 0x4008_0090).

Bits	Description	
[23:0]	**CAPSCNT**	**Counter Read Back Value Of Capacitive Touch Sensing Block**

7.4.11 Oscillator Frequency Measurement and Control

The ISD9100 series 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 7-16.

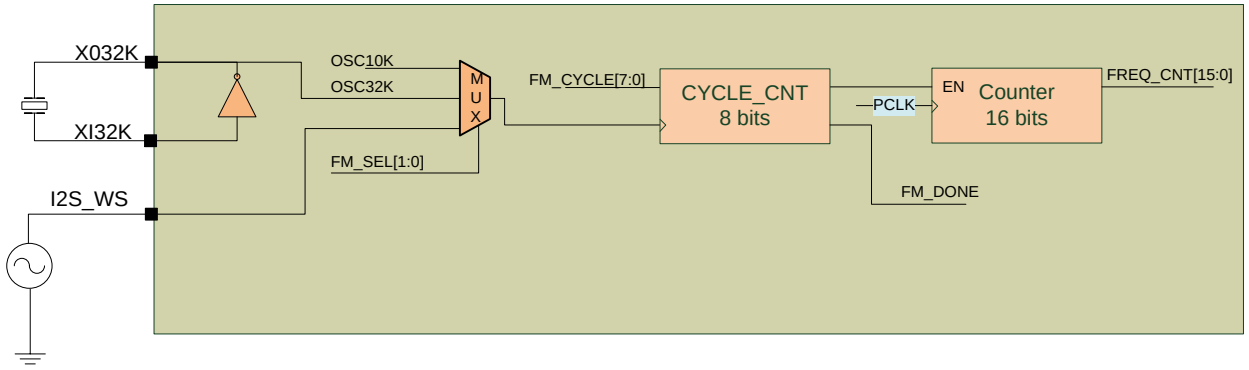


Figure 7-16 Oscillator Frequency Measurement Block Diagram

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 16kHz 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;
```

To adjust the oscillator the user can write to the SYS_IRCTCTL register (see Table 5-11). In addition, to obtain frequencies in between SYS_IRCTCTL trim settings a SUPERFINE function is available. The SUPERFINE function dithers the trim setting between the current setting and FINE trim settings above and below the current setting. An example of how the SUPERFINE trim register can adjust the measured oscillator frequency is shown in the figure below

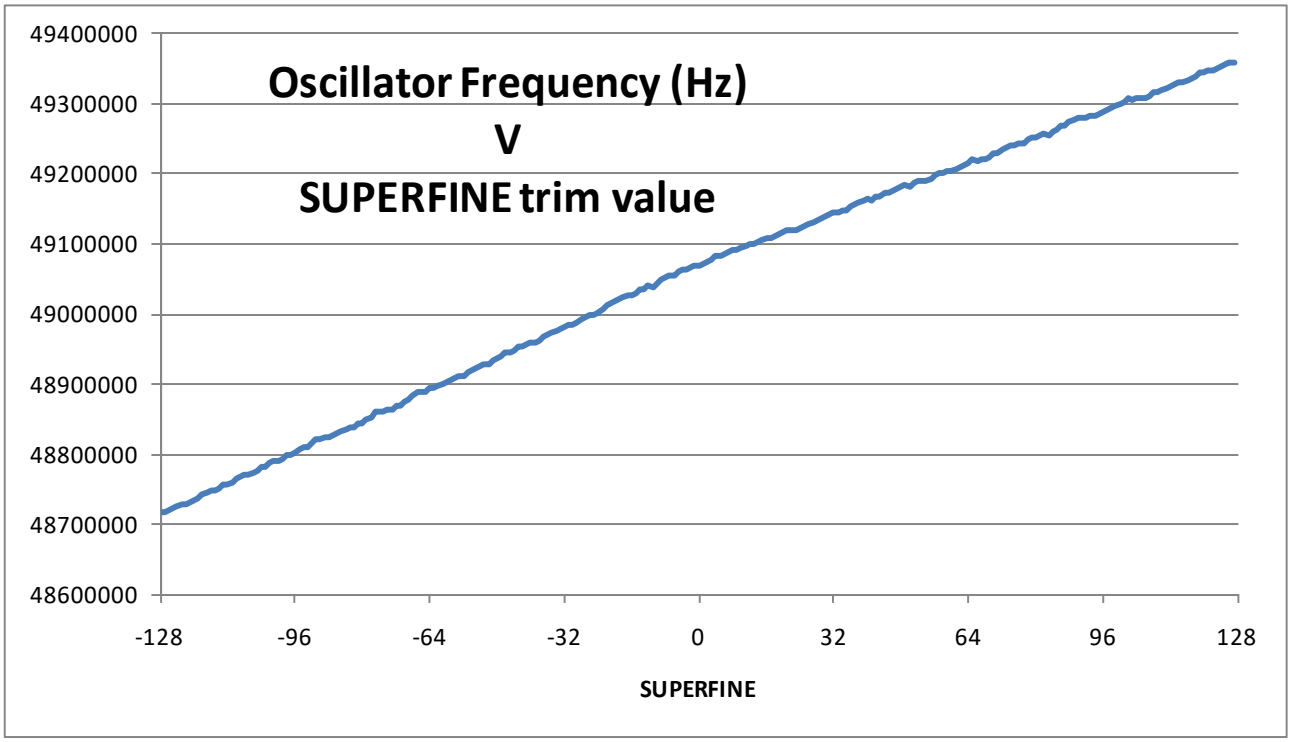


Figure 7-17 Example SUPERFINE Trim Frequency Adjustment.

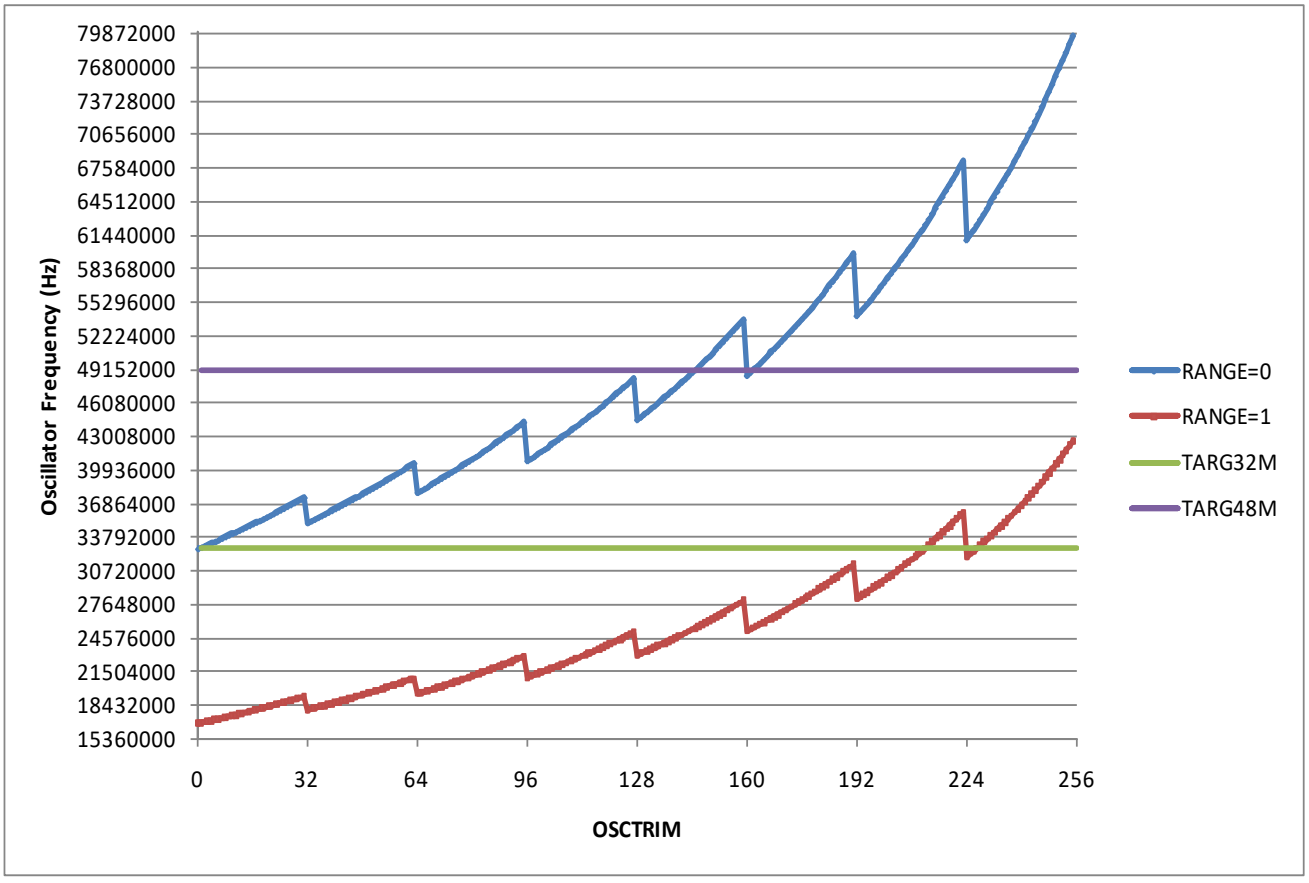


Figure 7-18 Typical Oscillator Frequency versus SYS_IRCTCTL Setting.

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**
SUPERFINE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
COARSE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OSCTRIM							

Table 7-37 Oscillator Trim Register (ANA_TRIM, address 0x4008_0084).

Bits	Description	
[23:16]	**SUPERFINE**	**Superfine** The superfine trim setting is an 8bit signed integer. It adjusts the master oscillator by dithering the FINE trim setting between the current setting and one setting above (values 1,127) or below (values -1, -128) the current trim setting. Each step effectively moves the frequency 1/128th of the full FINE trim step size.
[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	

Table 7-38 Frequency Measurement Control Register (ANA_FQMMCTL, address 0x4008_0094).

Bits	Description	
[31]	FQMMEN	**FQMMEN** 0 = Disable/Reset block. 1 = Start Frequency Measurement.
[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.
[2]	MMSTS	**Measurement Done** 0 = Measurement Ongoing. 1 = Measurement Complete.
[1:0]	CLKSEL	**Reference Clock Source** 00b: OSC16K, 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

Table 7-39 Frequency Measurement Count Register (ANA_FQMMCNT, address 0x4008_0098).

Bits	Description	
[15:0]	**FQMMCNT**	**Frequency Measurement Count** When MMSTS aaa 1 and G0 aaa 1, this is number of PCLK periods counted for frequency measurement. The frequency will be PCLK aaa FQMMCNT * Fref /(CYCLESEL+1) Hz Maximum resolution of measurement is Fref /(CYCLESEL+1)*2 Hz

7.5 Automatic Level Control (ALC)

7.5.1 Overview and Features

The ALC seeks to control the PGA gain such that the PGA output maintains a constant envelope. This helps to prevent clipping at the input of the sigma delta ADC while maximizing the full dynamic range of the ADC. The ALC monitors the output of the ADC biquad output when that filter is enabled in the ADC path, or the output of the SINC filter otherwise. The ADC 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 control, which is fed back to the PGA.

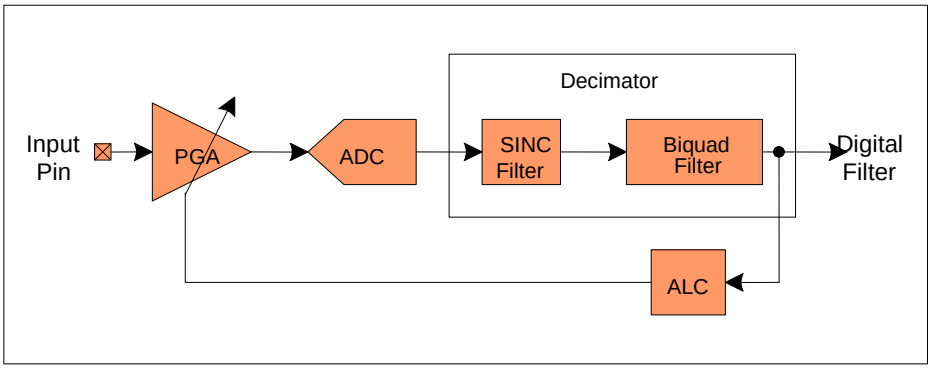


Figure 7-19 ALC Block Diagram

The ALC is enabled by setting ALCEN. The ALC shares a clock source with the Biquad filter so CLK_APBCLK0.BFALCKEN must be set to operate ALC. The ALC has two functional modes, which is set by MODESEL.

- Normal mode (MODESEL = LOW)
- Peak Limiter mode (MODESEL = HIGH)

When the ALC is disabled, the input PGA returns to the PGA gain setting held in ANA_PGAGAIN.GAINSET. In order to have a smooth transition when disabling the ALC, the user may prefer to fetch the ALC trained gain setting from ANA_PGAGAIN.GAINREAD and write that value to ANA_PGAGAIN.GAINSET prior to disabling the ALC. An input gain update must be made by writing to GAINSET[5:0]. A digital peak detector monitors the input signal amplitude and compares it to a register defined threshold level TARGETLV[3:0].

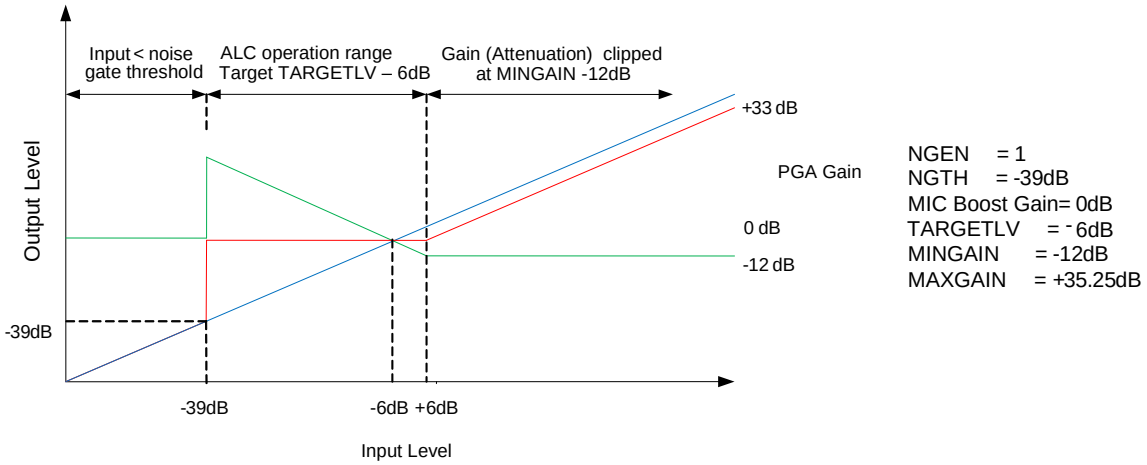


Figure 7-20: ALC Response Graph

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.

7.5.1.1 *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.

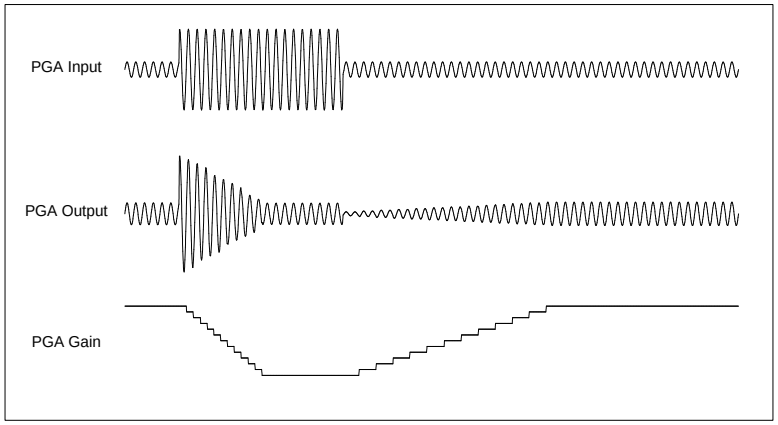


Figure 7-21: ALC Normal Mode Operation

7.5.1.2 *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.

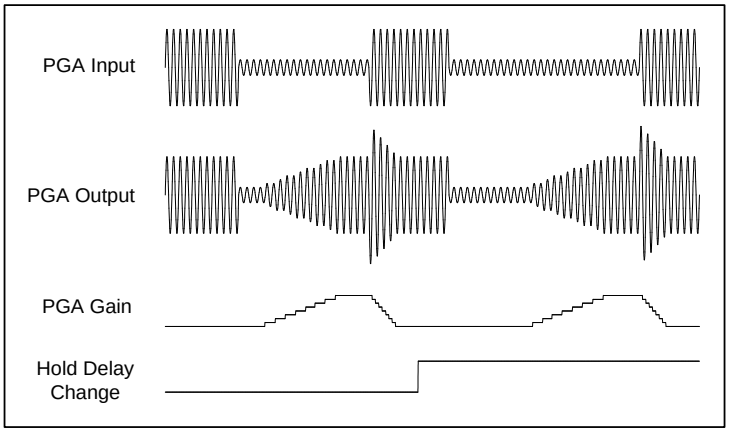


Figure 7-22: ALC Hold Time

7.5.1.3 *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.

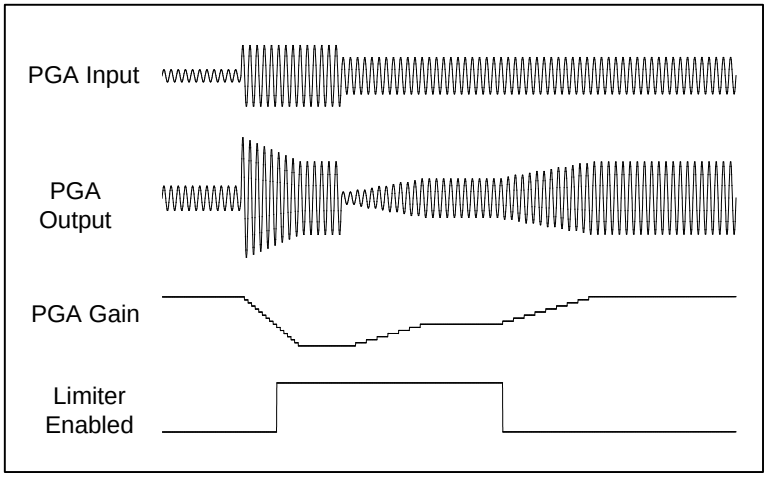


Figure 7-23: ALC Limiter Mode Operations

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 ADC output level has been reduced below the threshold. This limits ADC clipping if there is a sudden increase in the input signal level.

7.5.1.4 *Attack Time*

When the absolute value of the ADC 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 ADC output value when the absolute value of the ADC 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.

7.5.1.5 *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.

7.5.1.6 *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 NGTHBST 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 ADC – PGA gain – MIC Boost gain) < NGTHBST (dB)

Levels at the extremes of the range may cause inappropriate operation, so care should be taken when setting up the function.

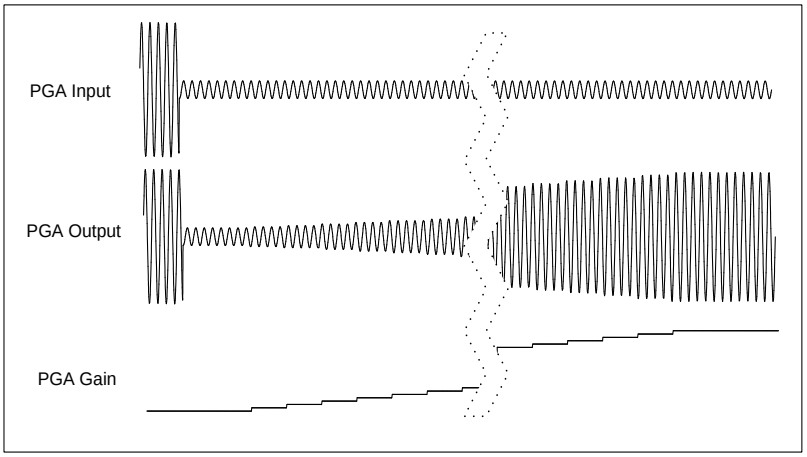


Figure 7-24: ALC Operation with Noise Gate disabled

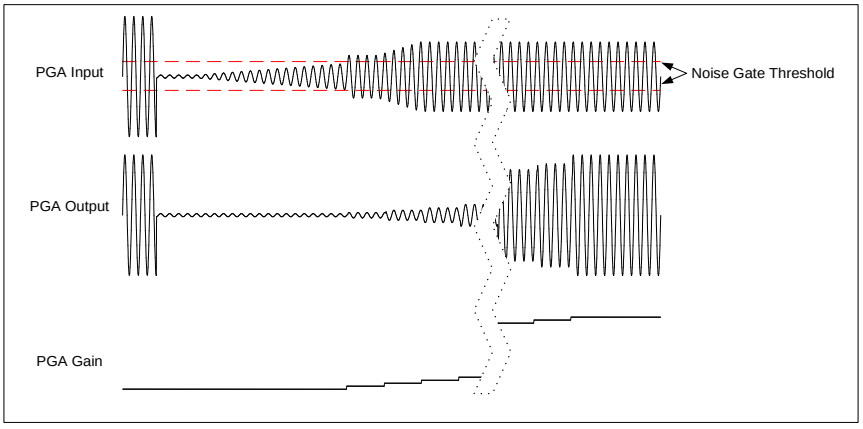


Figure 7-25: ALC Operation with Noise Gate Enabled

7.5.1.7 *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.
- Register ANA_SIGCTL.PUZCDCMP – is only relevant when the ALC is disabled.

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.

7.5.2 ALC Control 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_0048**				
ALC_CTL	ALC_BA+0x00	R/W	ALC Control Register	0x0E01_6320
ALC_STS	ALC_BA+0x04	R	ALC status register	0x0000_0000
ALC_INTSTS	ALC_BA+0x08	R/W	ALC interrupt register	0x0000_0000
ALC_INTCTL	ALC_BA+0x0C	R/W	ALC interrupt enable register	0x0000_0000

7.5.3 ALC Control Register Description

ALC Control Register (ALC_CTL)

Register	Offset	R/W	Description	Reset Value
ALC_CTL	ALC_BA+0x00	R/W	ALC Control Register	0x0E01_6320

Table 7-40 ALC Control Register (ALC_CTL, address 0x400B_0048)

31	30	29	28	27	26	25	24
PKLIMEN	PKSEL	NGPKSEL	ALCEN	MAXGAIN			MINGAIN
23	22	21	20	19	18	17	16
MINGAIN		ZCEN	HOLDTIME				TARGETLV[3]
15	14	13	12	11	10	9	8
TARGETLV[2:0]			MODESEL	DECAYSEL			
7	6	5	4	3	2	1	0
ATKSEL				NGEN	NGTHBST		

Bits	Description	
[31]	**PKLIMEN**	**ALC peak limiter enable** 0 = enable fast decrement when signal exceeds 87.5% of full scale (default) 1 = disable fast decrement when signal exceeds 87.5% of full scale
[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]	**ZCEN**	**ALC Zero Crossing** 0 = zero crossing disabled 1 = zero crossing enabled
[20:17]	**HOLDTIME**	**ALC Hold Time** (Value: 0~10). Hold Time aaa (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 aaa 0, Range: 125us to 128ms When MODESEL aaa 1, Range: 31us to 32ms (time doubles with every step)

[7:4]	**ATKSEL**	**ALC Attack Time** (Value: 0~10) When MODESEL aaa 0, Range: 500us to 512ms When MODESEL aaa 1,Range: 125us to 128ms (Both ALC time doubles with every step)
[3]	**NGEN**	**Noise Gate Enable** 0 = Noise gate disabled 1 = Noise gate enabled
[2:0]	**NGTHBST**	**Noise Gate Threshold** Boost disabled: Threshold aaa (-81+6xNGTHBST) dB Boost enabled: Threshold aaa (-87+6xNGTHBST) dB

ALC Status Register (ALC_STS)

Register	Offset	R/W	Description	Reset Value
ALC_STS	ALC_BA+0x04	R	ALC status register	0x0000_0000

Table 7-41 ALC Status Register (ALC_STS, address 0x400B_004C)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				PEAKVAL[8:5]			
15	14	13	12	11	10	9	8
PEAKVAL[4:0]					P2PVAL[8:6]		
7	6	5	4	3	2	1	0
P2PVAL[5:0]						NOISEF	CLIPFLAG

Bits	Description	
[31:19]	Reserved	Reserved
[18:11]	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]	CLIPFLAG	**Clipping Flag** Asserted when signal level is detected to be above 87.5% of full scale

ALC Interrupt Register (ALC_INTSTS)

Register	Offset	R/W	Description	Reset Value
ALC_INTSTS	ALC_BA+0x08	R/W	ALC interrupt register	0x0000_0000

Table 7-42 ALC Interrupt Register (ALC_INTSTS, address 0x400B_0050)

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							**INTFLAG**

Bits	Description	
[0]	**INTFLAG**	**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.

ALC Interrupt Enable Register (ALC_INTCTL)

Register	Offset	R/W	Description	Reset Value
ALC_INTCTL	ALC_BA+0x0C	R/W	ALC interrupt enable register	0x0000_0000

Table 7-43 ALC Interrupt Enable Register (ALC_INTCTL, address 0x400B_0054)

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							INTEN

Bits	Description	
[0]	INTEN	**ALC Interrupt Enable** 0 = ALC INT disabled 1 = ALC INT enabled

7.6 Biquad Filter (BIQ)

7.6.1 Overview and Features

A coefficient programmable 3-stage Biquad filter (6th-Order IIR filter) is available which can be used on either ADC 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.

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Upon power on reset or when the BIQ_CTL.DLCOEFF=0 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 at 7/16 Fs (Sample Rate).

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.

The operating Sample Rate of the filter can be setup by the following registers: The default value of BIQ_CTL.SRDIV (sample rate divider) is 3071, when the chip is running at HCLK=49.152Mhz, the operating SR of BIQ can be calculated by equation HCLK/(SRDIV+1) = 16Khz. The processor can change the operating sample rate (SR) by changing the SRDIV register.

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 3. The final BIQ sampling rate for DPWM path is based on both SRDIV and BIQ_CTL.DPWMPUSR registers which is equal to SR* (BIQ_CTL.DPWMPUSR+1) . So the default DPWM operating sample rate is 16*(3+1) = 64Khz.

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 ADC 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. Turn on BIQ_CTL.BIQEN. BIQ will start filter function.

7.6.2 BIQ Control Register Map

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_d010
BIQ_COEFF1	BIQ_BA+0x004	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0001_c020
BIQ_COEFF2	BIQ_BA+0x008	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0001_c020
BIQ_COEFF3	BIQ_BA+0x00c	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0001_ad66
BIQ_COEFF4	BIQ_BA+0x010	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_d1dc
BIQ_COEFF5	BIQ_BA + 0x14	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_c1d0
BIQ_COEFF6	BIQ_BA+0x018	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0001_83a0
BIQ_COEFF7	BIQ_BA+0x01c	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_c1d0
BIQ_COEFF8	BIQ_BA+0x020	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0001_7445
BIQ_COEFF9	BIQ_BA+0x024	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_92f6
BIQ_COEFF10	BIQ_BA + 0x28	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_b3cc
BIQ_COEFF11	BIQ_BA+0x02c	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0001_6798
BIQ_COEFF12	BIQ_BA+0x030	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_b3cc
BIQ_COEFF13	BIQ_BA+0x034	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0001_595d
BIQ_COEFF14	BIQ_BA+0x038	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_75d2
BIQ_CTL	BIQ_BA+0x040	R/W	BIQ Control Register	0x0BFF_0030

7.6.3 Register Description

BIQ Control Register (BIQ_CTL)

Register	Offset	R/W	Description	Reset Value
BIQ_CTL	BIQ_BA+0x040	R/W	BIQ Control Register	0x0BFF_0030

Table 7-44 BIQ Control Register (BIQ_CTL, address 0x400B_0040)

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

Bits	Description	
[28:16]	SRDIV	**Sample Rate Divider** This register is used to program the operating sampling rate of the biquad filter. The sample rate is defined as HCLK/(SRDIV+1). Default to 3071 so the sampling rate is 16K when HCLK is 49.152MHz.
[6:4]	DPWMPUSR	**DPWM Path Up Sample Rate (From SRDIV Result)** This register is only used when PATHSEL is set to 1. The operating sample rate for the biquad filter will be (DPWMPUSR+1)*HCLK/(SRDIV+1). Default value for this register is 3.
[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]	PRGCOEFF	**Programming Mode Coefficient Control Bit** 0 = Coefficient RAM is in normal mode. 1 = coefficient RAM is under programming mode. This bit must be turned off when BIQEN in on.
[1]	PATHSEL	**AC Path Selection For BIQ** 0 = used in ADC path 1 = used in DPWM path

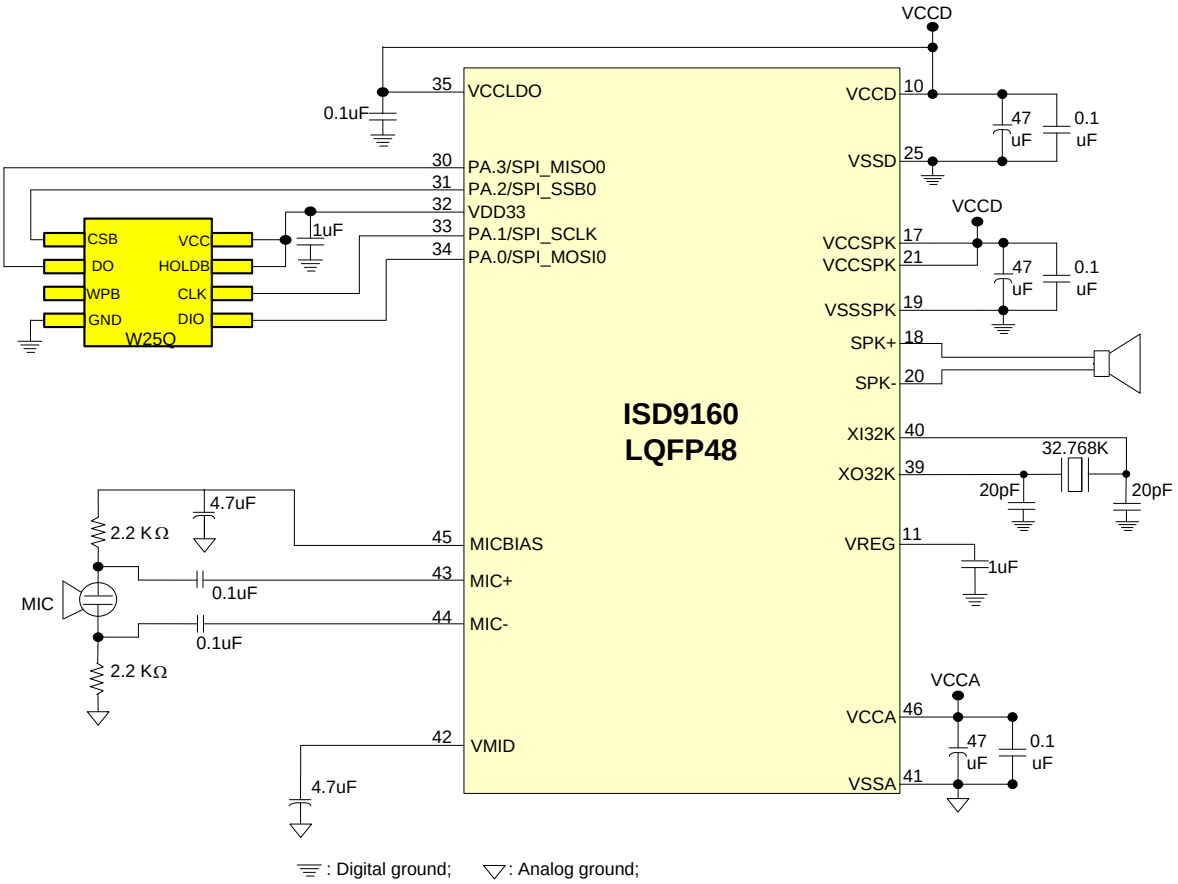
[0]	**BIQEN**	**BIQ Filter Start To Run** 0 = BIQ filter is not processing 1 = BIQ filter is on.

BIQ Coefficient (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_d010
BIQ_COEFF1	BIQ_BA+0x004	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0001_c020
BIQ_COEFF2	BIQ_BA+0x008	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0001_c020
BIQ_COEFF3	BIQ_BA+0x00c	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0001_ad66
BIQ_COEFF4	BIQ_BA+0x010	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_d1dc
BIQ_COEFF5	BIQ_BA + 0x14	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_c1d0
BIQ_COEFF6	BIQ_BA+0x018	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0001_83a0
BIQ_COEFF7	BIQ_BA+0x01c	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_c1d0
BIQ_COEFF8	BIQ_BA+0x020	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0001_7445
BIQ_COEFF9	BIQ_BA+0x024	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 2nd stage BIQ Coefficients	0x0000_92f6
BIQ_COEFF10	BIQ_BA + 0x28	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_b3cc
BIQ_COEFF11	BIQ_BA+0x02c	R/W	Coefficient b1 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0001_6798
BIQ_COEFF12	BIQ_BA+0x030	R/W	Coefficient b2 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_b3cc
BIQ_COEFF13	BIQ_BA+0x034	R/W	Coefficient a1 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0001_595d
BIQ_COEFF14	BIQ_BA+0x038	R/W	Coefficient a2 In H(z) Transfer Function (3.16 format) - 3rd stage BIQ Coefficients	0x0000_75d2

Bits	Description	
[31:0]	**COEFFDAT**	**Coefficient Data**

8 APPLICATION DIAGRAM

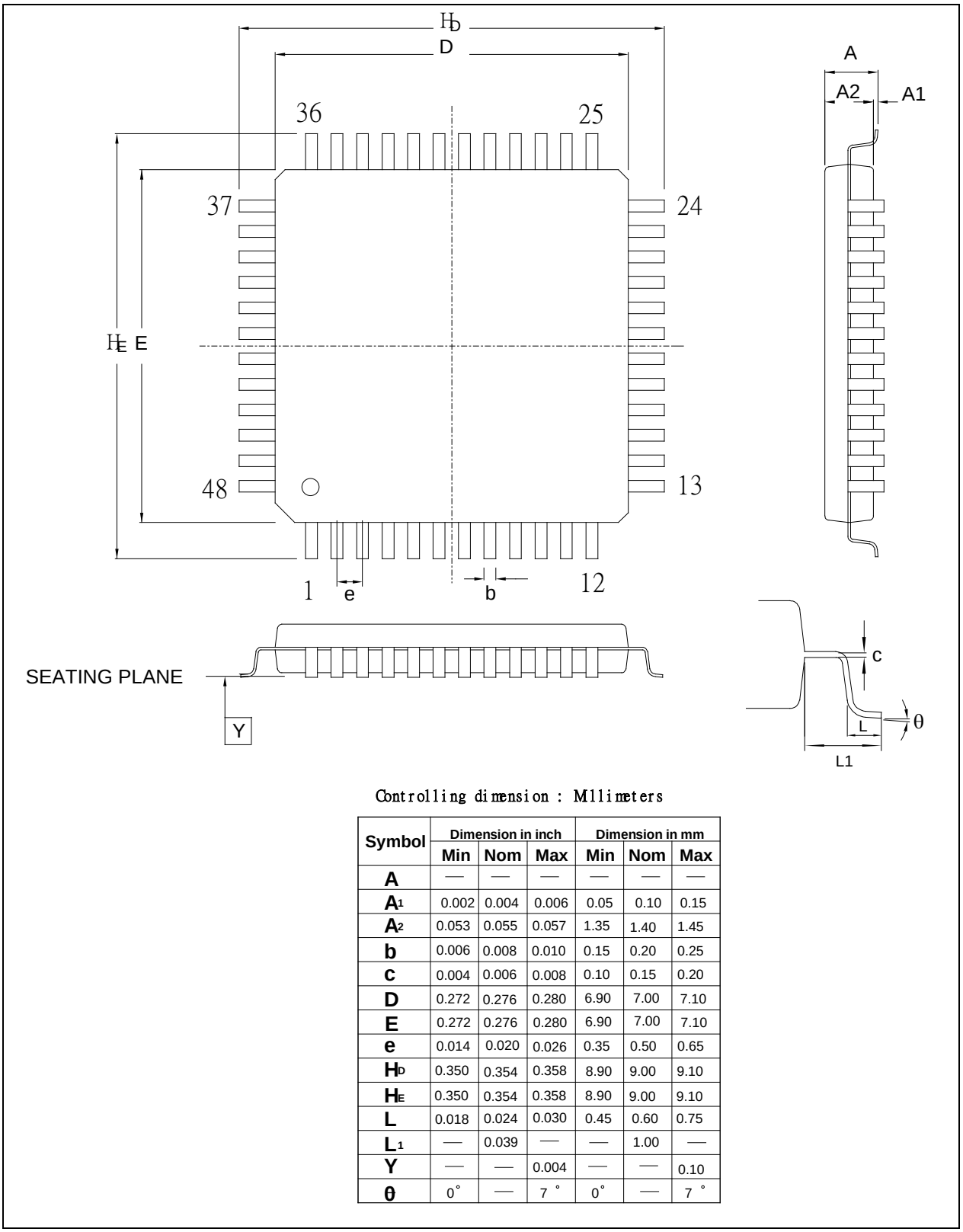


9 ELECTRICAL CHARACTERISTICS

For information on ISD9100 series electrical characteristics, please refer to ISD Cortex®-M0 Chipcorder ISD9100 series Datasheet.

10 PACKAGE DIMENSIONS

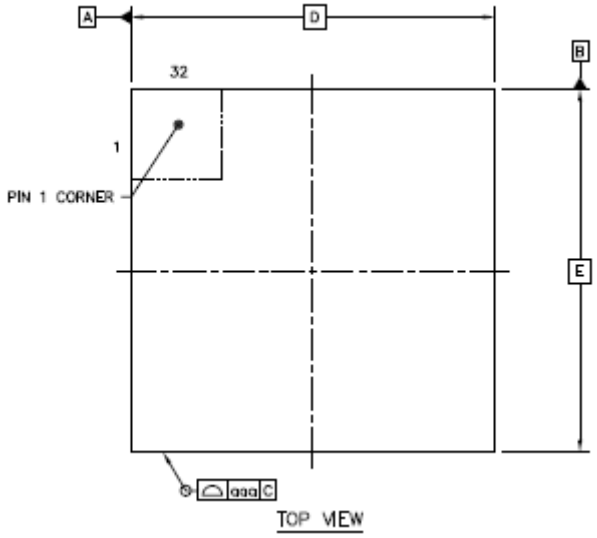
10.1.1 48L LQFP (7x7x1.4mm footprint 2.0mm)

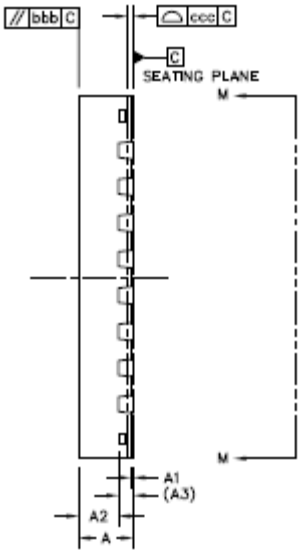


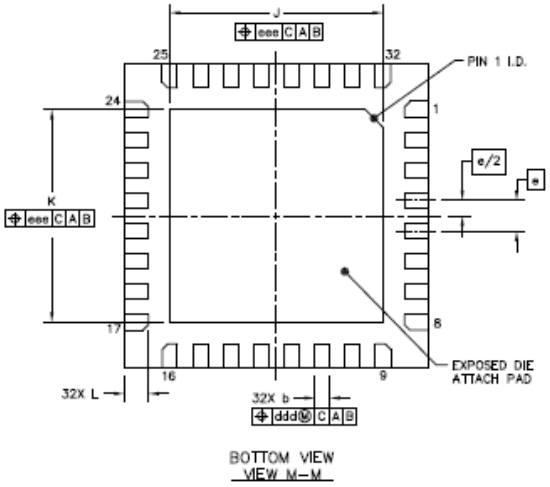
Controlling dimension : Millimeters

Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	—	—	—	—
A_1	0.002	0.004	0.006	0.05	0.10	0.15
A_2	0.053	0.055	0.057	1.35	1.40	1.45
b	0.006	0.008	0.010	0.15	0.20	0.25
c	0.004	0.006	0.008	0.10	0.15	0.20
D	0.272	0.276	0.280	6.90	7.00	7.10
E	0.272	0.276	0.280	6.90	7.00	7.10
e	0.014	0.020	0.026	0.35	0.50	0.65
H_D	0.350	0.354	0.358	8.90	9.00	9.10
H_E	0.350	0.354	0.358	8.90	9.00	9.10
L	0.018	0.024	0.030	0.45	0.60	0.75
L_1	—	0.039	—	—	1.00	—
Y	—	—	0.004	—	—	0.10
θ	0°	—	7°	0°	—	7°

10.1.2 33-pin QFN (5x5x0.8mm footprint 2.5mm)

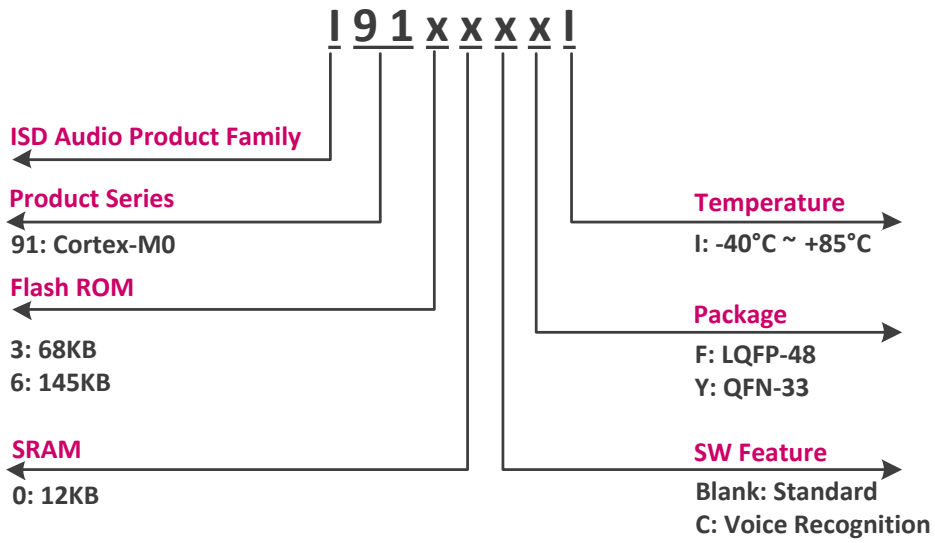






		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.7	0.75	0.8
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.55	0.57
L/F THICKNESS		A3		0.203 REF	
LEAD WIDTH		b	0.2	0.25	0.3
BODY SIZE	X	D		5 BSC	
	Y	E		5 BSC	
LEAD PITCH		e		0.5 BSC	
EP SIZE	X	J	3.4	3.5	3.6
	Y	K	3.4	3.5	3.6
LEAD LENGTH		L	0.35	0.4	0.45
PACKAGE EDGE TOLERANCE		aaa		0.1	
MOLD FLATNESS		bbb		0.1	
COPLANARITY		ccc		0.08	
LEAD OFFSET		ddd		0.1	
EXPOSED PAD OFFSET		eee		0.1	

11 ORDERING INFORMATION



12 REVISION HISTORY

VERSION	DATE	PAGE/ CHAP.	DESCRIPTION
V1.01	Sep 01, 2014	-	First Release.
V1.41	Mar 30, 2016	-	Add QFN33 pin package. Add ordering info.
V1.42	Aug 08, 2019	-	Add QFN33 pin description.
V1.5	Mar. 16, 2023	-	Added “Package is Halogen-free, RoHS-compliant and TSCA-compliant” in section 2 Remove Electrical characteristics section.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.