ISD9300 Series
Technical Reference Manual

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

www.nuvoton.com

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 GENERAL DESCRIPTION

The ISD9300 series are system-on-chip products optimized for low power, audio record and playback with an embedded ARM® Cortex™-M0 32-bit microcontroller core.

The series embeds a Cortex™-M0 core running up to 98MHz with 68K/100K/145K-byte of non-volatile flash memory and 16K-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, I2C, I2S), PWM modulators, GPIO, Analog Comparator, Low Voltage Detector and Brown-out detector.

The ISD9300 series 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 10µA.

For audio functionality the ISD9300 series includes a Sigma-Delta ADC with 90dB 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 of power to an 8Ω speaker.

The ISD9300 series provides in total 32 general purpose IO pins (GPIO), including 16 pins which are analog enabled. 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 sensing.

2 FEATURES

The equipped features are dependent on the product line and their sub products.

- Core
  - ARM® Cortex™-M0 core running up to 98.304MHz.
  - 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) support 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 power16kHz 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
  - 68/100/145K bytes Flash EPROM for program code and data storage.
  - Mini-cache to maintain near zero-wait state memory access.
  - 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
  - 16K 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 16-49.152MHz. Factory trimmed within 1% to settings of 49.152MHz 36.864MHz 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
  - 32 general purpose IO pins, including 16 analog enabled.
  - Four I/O modes:
    - Quasi-bidirectional
    - 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.
  - 90dB Signal-to-Noise (SNR) performance.
  - Programmable gain amplifier with 32 steps from -12 to 35.25dB in 0.75dB steps.
  - Boost gain stage of 26dB, giving maximum total gain of 61dB.
  - Input selectable from dedicated MIC pins or analog enabled GPIO.

- Programmable biquad filter to support multiple sample rates from 8-32kHz.
- DMA support for minimal CPU intervention.

- 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.
- Timer
  - Two timers with 8-bit pre-scaler and 24-bit resolution.
  - Counter auto reload.
- Watchdog 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
  - Six 16-bit PWM generators provide sixsingle ended PWM outputs or three 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
  - SPI Clock up to 24MHz.
  - SPI data rate in Quad mode of 98 Mbps.
  - Support MICROWIRE/SPI master/slave mode (SSP).
  - Full duplex synchronous serial data transfer.
  - Variable length of transfer data from 1 to 4 bytes.
  - MSB or LSB first data transfer.
  - 2 slave/device select lines when used in master mode.
  - Hardware CRC calculation module available for CRC calculation of data stream.
  - DMA support.
  - Quad/Dual SPI support.
- I^2C
  - 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 16 levels: 2.1V, 2,2V, 2.4V, 2.5V, 2.65V, 2.8V, 3.0V, 3.1V, 3.2V, 3.4V, 3.6V, 3.7V, 3.8V, 3.9V, 4.2V, 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.
  - Ten GPIO (GPIOA<7:0>& GPIOB<13:12>) 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: -40℃~85℃
- Packages:
  - All Green package (RoHS).
  - LQFP 64-pin.

3 ABBREVIATIONS

Acronym	Description
ACMP	Analog Comparator Controller
ADC	Analog-to-Digital Converter
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
BOD	Brown-out Detection
FIFO	First In, First Out
FMC	Flash Memory Controller
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
MPU	Memory Protection Unit
NVIC	Nested Vectored Interrupt Controller
PDMA	Peripheral Direct Memory Access
PLL	Phase-Locked Loop
PWM	Pulse Width Modulation
SPI	Serial Peripheral Interface
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
WDT	Watchdog Timer

Table 3-1 List of Abbreviations

4 PARTS INFORMATION LIST AND PIN CONFIGURATION

4.1 ISD9300 Product Selection Guide

Part number	Flash	SRAM	Max. CPU Freq	I/O	Package
I9331RI	68 KB	16 KB	49.152 MHz	up to 32	LQFP64
I9331VRI	68 KB	16 KB	49.152 MHz	up to 32	LQFP64
I9341RI	100 KB	16 KB	49.152 MHz	up to 32	LQFP64
I9341VRI	100 KB	16 KB	49.152 MHz	up to 32	LQFP64
I9361RI	145 KB	16 KB	49.152 MHz	up to 32	LQFP64
I9361VRI	145 KB	16 KB	49.152 MHz	up to 32	LQFP64

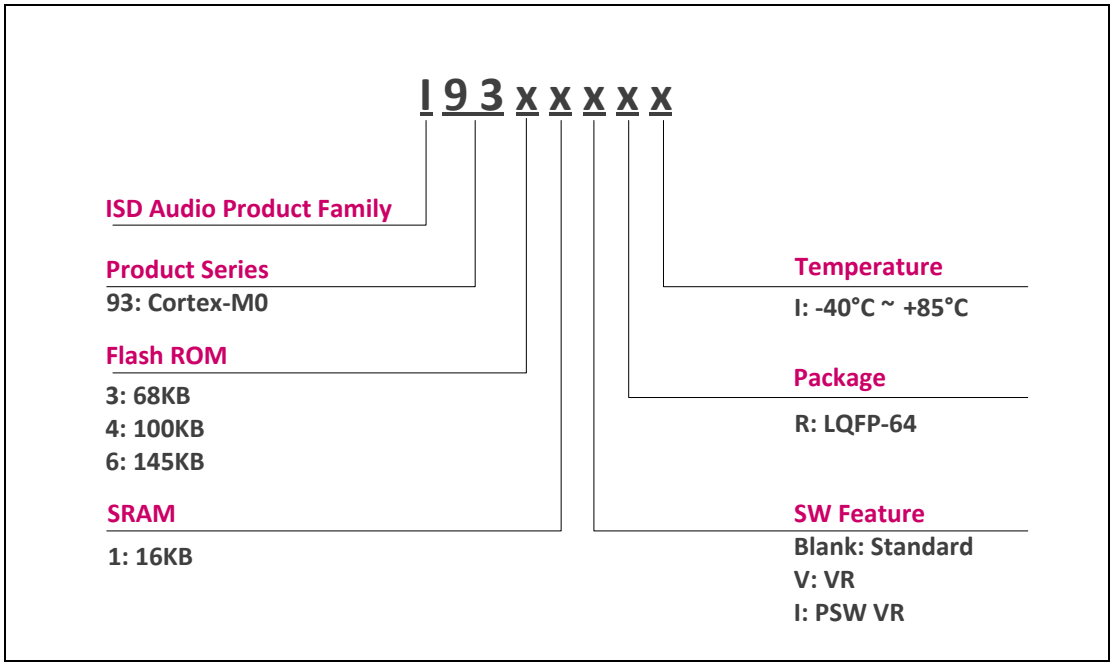


Figure 4-1 ISD9300 Series Selection Code

4.2 Pin Configuration

4.2.1 ISD9300 Pin Diagram

4.2.1.1 ISD9300 LQFP 64 pin

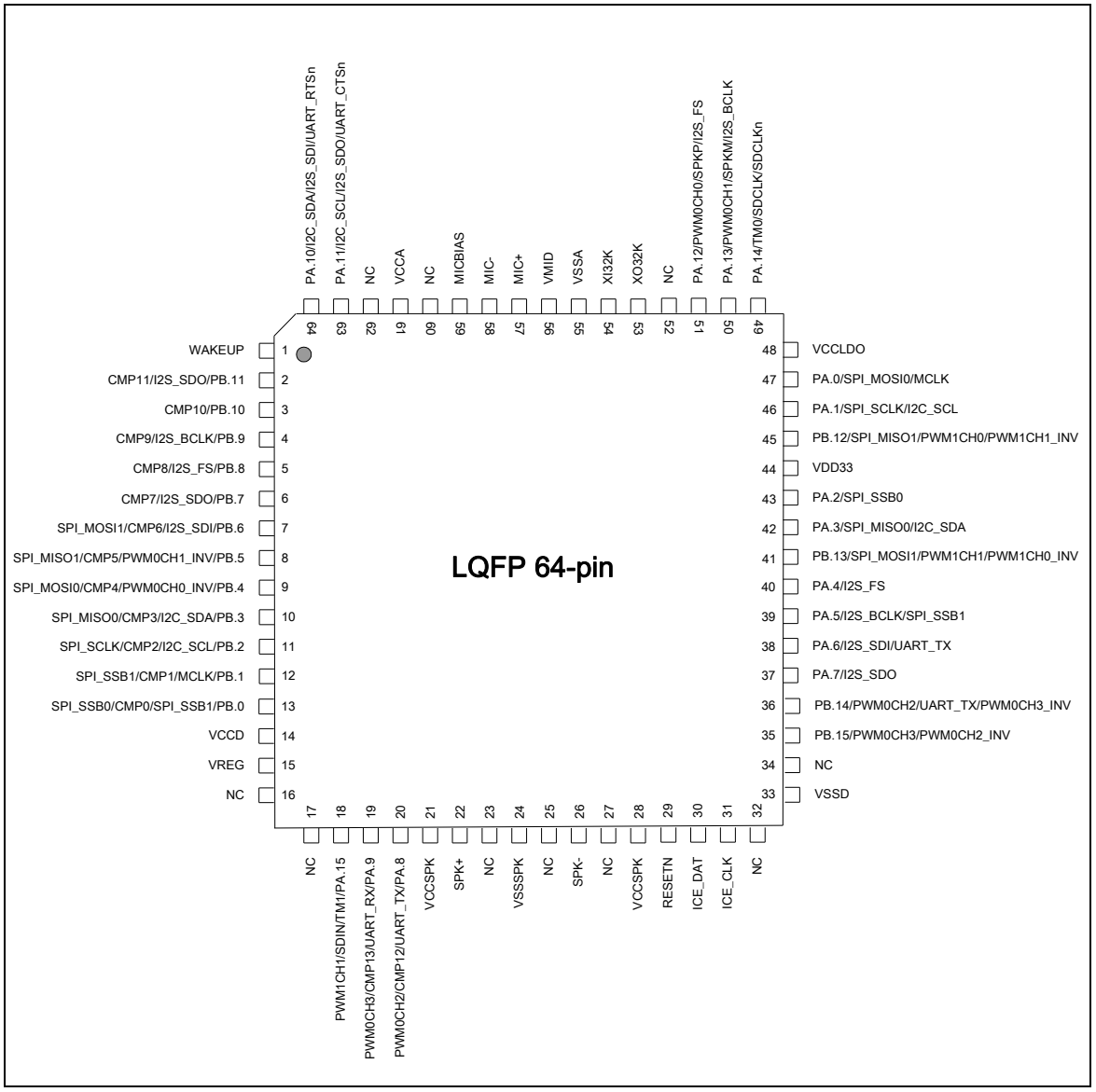


Figure 4-2 ISD9300 LQFP 64-pin Diagram

4.3 Pin Description

4.3.1 ISD9300 Pin Description

Pin No. (LQFP 64)	Pin Name	Pin Type	Alt CFG	Description
1	WAKEUP	**I**		Pull low to wake part from deep power down
2	PB.11	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 11
	I2S_SDO	**O**	1	I2S Serial Data out
	CMP11	**AIO**	2	Configure as relaxation oscillator for CapSense
3	PB.10	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 10
	CMP10	**AIO**	2	Configure as relaxation oscillator for CapSense
4	PB.9	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 9
	I2S_BCLK	**O**	1	I2S Bit Clock (master mode only)
	CMP9	**AIO**	2	Configure as relaxation oscillator for CapSense
5	PB.8	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 8
	I2S_FS	**O**	1	I2S Frame Sync (master mode only)
	CMP8	**AIO**	2	Configure as relaxation oscillator for CapSense
6	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 CapSense
7	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 CapSense
	SPI_MOSI1	**O**	3	Master Out, Slave In channel 1 for SPI interface
8	PB.5	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 5
	PWM0CH1_INV	**O**	1	Invert PWM0 channel 1 output pin
	CMP5	**AIO**	2	Configure as relaxation oscillator for CapSense
	SPI_MISO1	**I**	3	Master In, Slave Out channel 1 for SPI interface
9	PB.4	**A/I/O**	0	General purpose input/output pin, analog capable; Port B, bit 4
	PWM0CH0_INV	**O**	1	Invert PWM0 channel 0 output pin
	CMP4	**AIO**	2	Configure as relaxation oscillator for CapSense
	SPI_MOSI0	**O**	3	Master Out, Slave In channel 0 for SPI interface
10	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 CapSense

Pin No. (LQFP 64)	Pin Name	Pin Type	Alt CFG	Description
	SPI_MISO0	**I**	3	Master In, Slave Out channel 0 for SPI interface
11	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 CapSense
	SPI_SCLK	**I/O**	3	Serial Clock for SPI interface
12	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 CapSense
	SPI_SSB1	**O**	3	Slave Select Bar 1 for SPI interface
13	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**	3	Slave Select Bar 1 for SPI interface
	CMP0	**AIO**	2	Configure as relaxation oscillator for CapSense
	SPI_SSB0	**I/O**	3	Slave Select Bar 0 for SPI interface
14	VCCD	**P**		Main Digital Supply for Chip. Supplies all IO except analog, Speaker Driver and PA<7:0>
15	VREG	**P**		Logic regulator output decoupling pin. A 1µF capacitor returning to VSSD must be placed on this pin.
16	NC			Should remain unconnected.
17	NC			Should remain unconnected.
18	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
	PWM1CH1	**O**	3	Output of PWM1 channel 1
19	PA.9	**I/O**	0	General purpose input/output pin; Port A, bit 9
	UART_RX	**I**	1	Receive channel of UART
	CMP13	**AIO**	2	Configure as relaxation oscillator for CapSense
	PWM0CH3	**O**	3	Output of PWM0 channel3
20	PA.8	**I/O**	0	General purpose input/output pin; Port A, bit 8
	UART_TX	**O**	1	Transmit channel of UART
	CMP12	**AIO**	2	Configure as relaxation oscillator for CapSense
	PWM0CH2	**O**	3	Output of PWM0 channel2
21	VCCSPK	**P**		Power Supply for PWM Speaker Driver
22	SPK+	**O**		Positive Speaker Driver Output

Pin No. (LQFP 64)	Pin Name	Pin Type	Alt CFG	Description
24	VSSSPK	**P**		Ground for PWM Speaker Driver
26	SPK-	**O**		Negative Speaker Driver Output
28	VCCSPK	**P**		Power Supply for PWM Speaker Driver
29	RESETN	**I**		External reset input. Pull this pin low to reset device to initial state. Has internal weak pull-up.
30	ICE_DAT	**I/O**		Serial Wire Debug port data pin. Has internal weak pull-up.
31	ICE_CLK	**I**		Serial Wire Debug port clock pin. Has internal weak pull-up.
33	VSSD	**P**		Digital Ground.
35	PB.15	**I/O**	0	General purpose input/output pin; Port B, bit 15
	PWM0CH3	**O**	1	Output of PWM0 channel3
	PWM0CH2_INV	**O**	3	Invert PWM0 channel2 output pin.
36	PB.14	**I/O**	0	General purpose input/output pin; Port B, bit 14
	PWM0CH2	**O**	1	Output of PWM0 channel2
	UART_TX	**O**	2	UART transmit
	PWM0CH3_INV	**O**	3	Invert PWM0 channel3 output pin.
37	PA.7	**I/O**	0	General purpose input/output pin; Port A, bit 7
	I2S_SDO	**O**	1	Serial Data Out for I2S interface
38	PA.6	**I/O**	0	General purpose input/output pin; Port A, bit 6
	I2S_SDI	**I**	1	Serial Data In for I2S interface
	UART_TX	**O**	2	UART transmit
39	PA.5	**I/O**	0	General purpose input/output pin; Port A, bit 5
	I2S_BCLK	**I/O**	1	Bit Clock for I2S interface
	SPI_SSB1	**I/O**	3	SPI Slave Select 1
40	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
41	PB.13	**I/O**	0	General purpose input/output pin; Port B, bit 13
	SPI_MOSI1	**I/O**	1	Master Out, Slave In channel 1 for SPI interface
	PWM1CH1	**O**	2	Output of PWM1 channel1
	PWM1CH0_INV	**O**	3	Invert PWM1 channel0 output pin.
42	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
43	PA.2	**I/O**	0	General purpose input/output pin; Port A, bit 2

Pin No. (LQFP 64)	Pin Name	Pin Type	Alt CFG	Description
	SPI_SSB0	**I/O**	1	Slave Select Bar 0 for SPI interface
44	VDD33	**P**		LDO Regulator Output. If used, a 1µF capacitor must be placed to ground. If not used then tie to VCCD.
45	PB.12	**I/O**	0	General purpose input/output pin; Port B, bit 12
	SPI_MISO1	**I/O**	1	Master In, Slave Out channel 1 for SPI interface
	PWM1CH0	**O**	2	Output of PWM1 channel0
	PWM1CH1_INV	**O**	3	Invert PWM1 channel1 output pin.
46	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
47	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.
48	VCCLDO	**P**		Power Supply for LDO, should be connected to VCCD
49	PA.14	**I/O**	0	General purpose input/output pin; Port A, bit 14
	TM0	**I**	1	External input to Timer 0
	SDCLK	**O**	2	Clock output for digital microphone mode.
	SDCLKn	**O**	3	Inverse Clock output for digital microphone mode.
50	PA.13	**I/O**	0	General purpose input/output pin; Port A, bit 13
	PWM0CH1	**O**	1	PWM0 channel1 Output.
	SPKM	**O**	2	Equivalent to SPK-.
	I2S_BCLK	**I/O**	3	Bit Clock for I2S interface
51	PA.12	**I/O**	0	General purpose input/output pin; Port A, bit 12
	PWM0CH0	**O**	1	PWM0 channel0 Output.
	SPKP	**O**	2	Equivalent to SPK+
	I2S_FS	**I/O**	3	Frame Sync Clock for I2S interface
53	XO32K	**O**		32.768kHz Crystal Oscillator Output
54	XI32K	**I**		32.768kHz Crystal Oscillator Input. Max Voltage 1.8V
55	VSSA	**AP**		Ground for analog circuitry.
56	VMID	**O**		Mid rail reference. Connect 4.7µF to VSSA.
57	MIC+	**AI**		Positive microphone input.
58	MIC-	**AI**		Negative microphone input.
59	MICBIAS	**AO**		Microphone bias output.

Pin No. (LQFP 64)	Pin Name	Pin Type	Alt CFG	Description
61	VCCA	**AP**		Analog power supply.
63	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.
64	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.

Note: Pin Type I=Digital Input, O=Digital Output; AI=Analog Input; P=Power Pin; AP=Analog Power

5 BLOCK DIAGRAM

5.1 ISD9300 Block Diagram

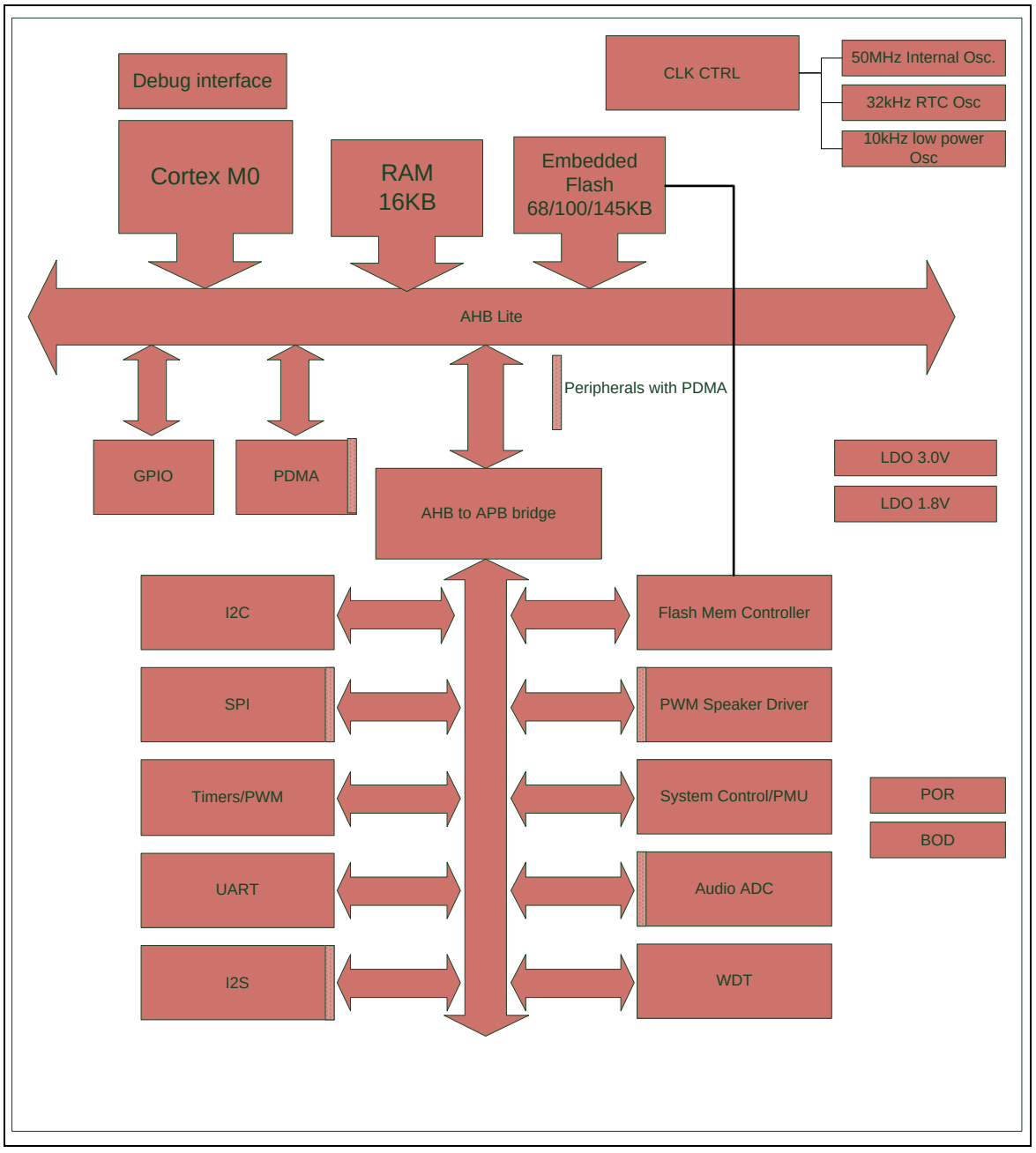


Figure 5-1 ISD9300 Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 ARM® Cortex™-M0 Core

The Cortex™-M0 processor is a configurable, multistage, 32-bit RISC processor, which has an AMBA AHB-Lite interface and includes an NVIC component. It also has optional hardware debug functionality. The processor can execute Thumb code and is compatible with other Cortex™-M profile processor. The profile supports two modes -Thread mode and Handler mode. Handler mode is entered as a result of an exception. An exception return can only be issued in Handler mode. Thread mode is entered on Reset, and can be entered as a result of an exception return.

Figure 6-1 shows the functional controller of processor.

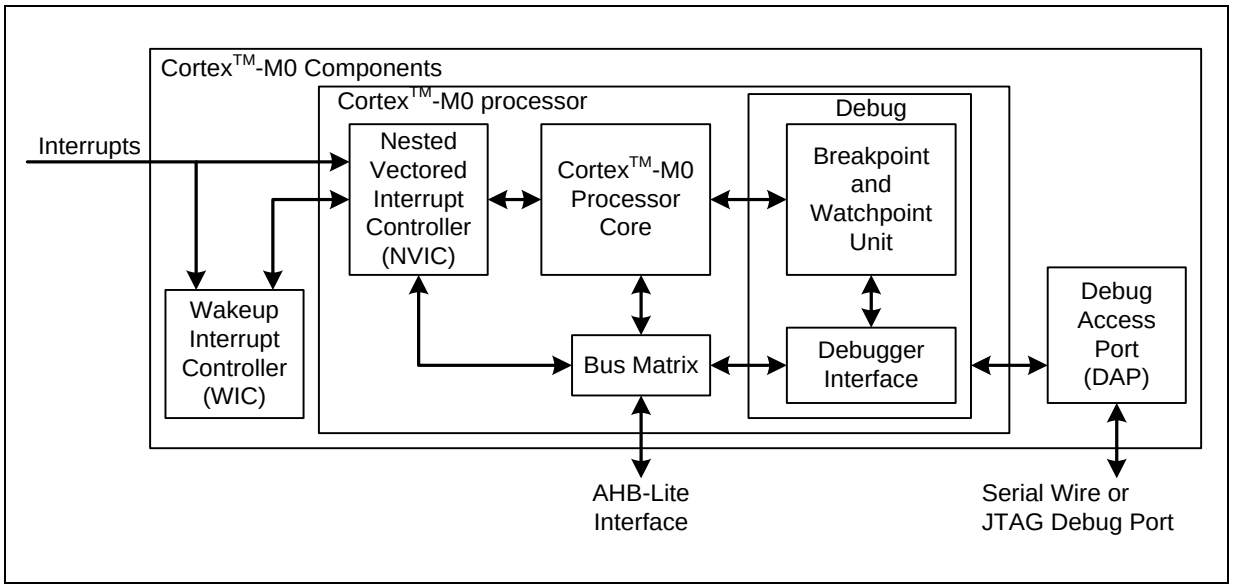


Figure 6-1 Functional Controller Diagram

The implemented device provides the following components and features:

- A low gate count processor:
  - ARMv6-M Thumb® instruction set
  - Thumb-2 technology
  - ARMv6-M compliant 24-bit SysTick timer
  - A 32-bit hardware multiplier
  - System interface supportedwith little-endian data accesses
  - Ability to have deterministic, fixed-latency, interrupt handling
  - Load/store-multiples and multicycle-multiplies that can be abandoned and restarted to facilitate rapid interrupt handling
  - C Application Binary Interface compliant exception model. This is the ARMv6-M, C Application Binary Interface (C-ABI) compliant exception model that enables the use of pure C functions as interrupt handlers
  - Low Power Sleepmode entry using Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or the return from interrupt sleep-on-exit feature

- NVIC:
  - 32 external interrupt inputs, each with four levels of priority
  - Dedicated Non-maskable Interrupt (NMI) input
  - Supports for both level-sensitive and pulse-sensitive interrupt lines
  - Supports Wake-up Interrupt Controller (WIC) and, providing Ultra-low Power Sleep mode
- 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)

6.2 System Manager

6.2.1 Overview

System management includes thefollowing sections:

- System Resets
- System Memory Map
- System management registersfor Product ID, chip reset and on-chip controllers reset , multi-functional pin control
- System Timer (SysTick)
- Nested Vectored Interrupt Controller (NVIC)
- Brown-Out and chip miscellaneous Control Register
- Combined peripheral interrupt source identify

6.2.2 System Reset

The system reset can be issued by one of the following listed events. For these reset event flags can be read by SYS_RSTSTS register.

- Power-on Reset
- 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.

6.2.3 System Power Distribution

The ISD9300 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>& GPIOB<13:12>.
- 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.

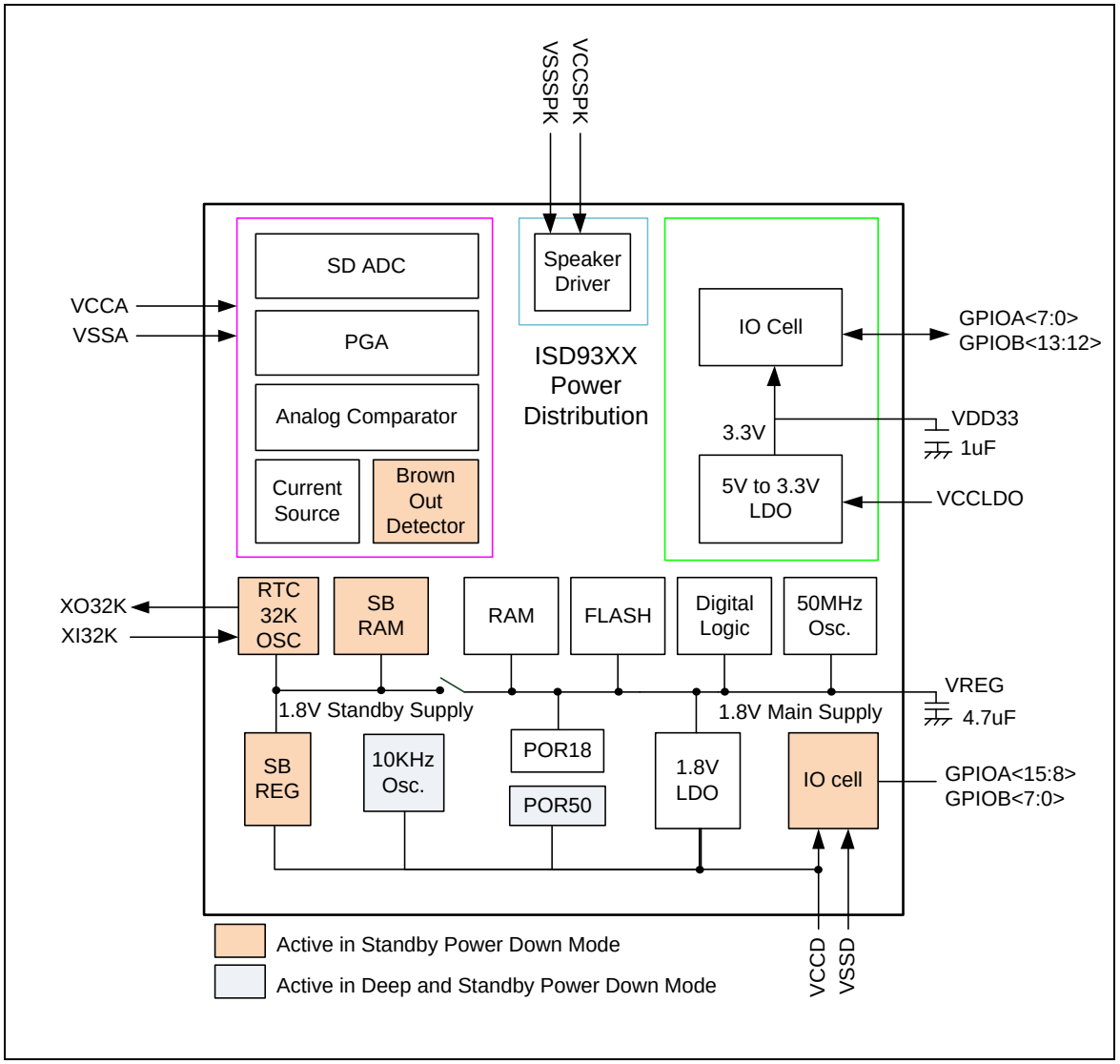


Figure 6-2 ISD9300 Power Distribution Diagram

6.2.4 System Memory Map

The ISD9300 series provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown in the following table. The detailed register definition, memory space, and programming detailed will be described in the following sections for each on-chip peripheral. The ISD9300 series only supports little-endian data format.

Address Space	Token	Controllers
Flash and SRAM Memory Space		
0x0000_0000 – 0x0000_FFFF	FLASH_BA	FLASH Memory Space (64KB)
0x0000_0000 – 0x0001_7FFF	FLASH_BA	FLASH Memory Space (96KB)
0x0000_0000 – 0x0002_33FF	FLASH_BA	FLASH Memory Space (141KB)
0x0000_0000 – 0x0002_43FF	FLASH_BA	FLASH Memory Space (145KB)
0x2000_0000 – 0x2000_3FFF	SRAM_BA	SRAM Memory Space (16KB)
AHB Controllers Space (0x5000_0000 – 0x501F_FFFF)		
0x5000_0000 – 0x5000_01FF	SYS_BA	System Global Control Registers
0x5000_0200 – 0x5000_02FF	CLK_BA	Clock Control Registers
0x5000_0300 – 0x5000_03FF	INT_BA	Interrupt Multiplexer Control Registers
0x5000_4000 – 0x5000_7FFF	GPIO_BA	GPIO Control Registers
0x5000_8000 – 0x5000_BFFF	PDMA_BA	Peripheral DMA Control Registers
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash Memory Control Registers
APB1 Controllers Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_7FFF	WDT_BA	Watchdog Timer Control Registers
0x4000_8000 – 0x4000_BFFF	RTC_BA	Real Time Clock (RTC) Control Register
0x4001_0000 – 0x4001_3FFF	TIMER0_BA	Timer0/Timer1 Control Registers
0x4002_0000 – 0x4002_3FFF	I2C0_BA	I^2C0 Interface Control Registers
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 Serial Interface Control Registers
0x4004_0000 – 0x4004_3FFF	PWM_BA	PWM0/1 Control Registers
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0 Control Registers
0x4006_0000 – 0x4007_3FFF	DPWM_BA	Differential Audio PWM Speaker Driver
0x4006_0000 – 0x4008_3FFF	ANA_ BA	Analog Block Control Registers
0x4006_0000 – 0x4008_7FFF	BODTALM_BA	Brown Out Detector Control Registers
0x4006_0000 – 0x400A_FFFF	I2S_BA	I2S Interface Control registers
0x4006_0000 – 0x400B_FFFF	BIQ_BA	Biquad Filter Control Registers
0x400D_0000 – 0x400D_3FFF	ACMP_BA	Analog Comparator Control Registers
0x400E_0000 – 0x400E_FFFF	ADC0_BA	Analog-Digital-Converter (ADC) Control Registers
0x400F_0000 – 0x400F_7FFF	SBRAM_BA	Standby RAM Block Address space

System Controllers Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SYSTICK_BA	System Timer Control Registers
0xE000_E100 – 0xE000_ECFF	SCS_BA	External Interrupt Controller Control Registers
0xE000_ED00 – 0xE000_ED8F	SYSINFO_BA	System Control Registers

Table 6-1 Address Space Assignments for On-Chip Controllers

6.2.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
SYS Base Address: **SYS_BA = 0x5000_0000**				
SYS_PDID	SYS_BA+0x00	R	Product ID	0xXXXX_XXXX
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Source Register	0x0000_00XX
SYS_IPRST0	SYS_BA+0x08	R/W	IP Reset Control Resister0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	IP Reset Control Resister1	0x0000_0000
SYS_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 alternatefunctions control register	0x0000_0000
SYS_GPB_MFP	SYS_BA+0x3C	R/W	GPIOB multiple alternate functions control register	0x0000_0000
SYS_WKCTL	SYS_BA+0x54	R/W	WAKEUP pin control register	0x0000_0006
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control	0x0000_0000
SYS_IRCTCTL	SYS_BA+0x110	R/W	Oscillator Frequency Adjustment control register	0xXXXX_XXXX

6.2.6 Register Description

Part Device ID Code Register (PDID)

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

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

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

Bits	Description	
[31:0]	**PDID**	**Product Identifier** Chip identifier for ISD9300 series.

System Reset Source Register (SYS_RSTSTS)

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

Register	Offset	R/W	Description	Reset Value
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Source Register	0x0000_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
					PORF	DPDRSTF	WKRSTF
7	6	5	4	3	2	1	0
CPURF	PMURSTF	SYSRF	Reserved	Reserved	WDTRF	Reserved	CORERSTF

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**PORF**	**Power On Reset Flag** The PORF flag is set by hardware if device has powered up from a power on reset condition or standby power down. 0=No detected. 1= A power on Reset has occurred. This bit is cleared by writing 1 to itself. Writing 1 to this bit will clear bits PORF, DPDRSTF, and WKRSTF
[9]	**DPDRSTF**	**Deep Power Down Reset Flag** The DPDRSTF flag is set by hardware if device has powered up due to the DPD timer function. 0=No detected. 1= A power on was triggered by DPD timer. This bit is cleared by writing 1 to itself. Writing 1 to this bit will clear bits PORF, DPDRSTF, and WKRSTF
[8]	**WKRSTF**	**Wakeup Pin Reset Flag** The WKRSTF flag is set by hardware if device has powered up from deep power down (DPD) due to action of the WAKEUP pin. 0=No detected. 1= A power on was triggered by WAKEUP pin. This bit is cleared by writing 1 to itself. Writing 1 to this bit will clear bits PORF, DPDRSTF, and WKRSTF
[7]	**CPURF**	**Reset Source From CPU** The CPURF flag is set by hardware if software writes CPURST (SYS_IPRST0[1]) with a "1" to reset Cortex-M0 CPU kernel and Flash memory controller (FMC). 0= No reset from CPU. 1= The Cortex-M0 CPU kernel and FMC has been reset by software setting CPURST to 1.

		This bit is cleared by writing 1 to itself.
[6]	**PMURSTF**	**Reset Source From PMU** The PMURSTF flag is set if the PMU. 0= No reset from PMU. 1= PMU reset the system from a power down/standby event. This bit is cleared by writing 1 to itself.
[5]	**SYSRF**	**Reset Source From MCU** The SYSRF flag is set if the previous reset source originates from the Cortex_M0 kernel. 0= No reset from MCU. 1= The Cortex_M0 MCU issued a reset signal to reset the system by software writing 1 to bit SYSRESTREQ(SYSCTL_AIRCTL[2], Application Interrupt and Reset Control Register) in system control registers of Cortex_M0 kernel. This bit is cleared by writing 1 to itself.
[4:3]	**Reserved**	Reserved.
[2]	**WDTRF**	**Reset Source From WDG** The WDTRF flag is set if pervious reset source originates from the Watch-Dog module. 0= No reset from Watch-Dog. 1= The Watch-Dog module issued the reset signal to reset the system. This bit is cleared by writing 1 to itself.
[1]	**Reserved**	Reserved.
[0]	**CORERSTF**	**Reset Source From CORE** The CORERSTFflag is set if the core has been reset. Possible sources of reset are a Power-On Reset (POR), RESETn Pin Reset or PMU reset. 0= No reset from CORE. 1= Core was reset by hardware block. This bit is cleared by writing 1 to itself.

IP Reset Control Register1(SYS_IPRST0)

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Controller Reset** Set “1” will generate a reset signal to the PDMA Block. User needs to set this bit to “0” to release from the reset state 0= Normal operation. 1= PDMA IP reset.
[1]	**CPURST**	**CPU Kernel One Shot Reset** Setting this bit will reset the CPU kernel and Flash Memory Controller(FMC), this bit will automatically return to “0” after the 2 clock cycles This bit is a protected bit, to program first issue the unlock sequence (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 Register1 (SYS_IPRST1)

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

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

31	30	29	28	27	26	25	24
Reserved		I2S_RST	ADC_RST	USBD_RST	Reserved	CAN1_RST	CAN0_RST
23	22	21	20	19	18	17	16
PS2_RST	ACMP_RST	PWM47_RST	PWM03_RST	Reserved	UART2_RST	UART1_RST	UART0_RST
15	14	13	12	11	10	9	8
SPI3_RST	SPI2_RST	SPI1_RST	SPI0_RST	Reserved		I2C1_RST	I2C0_RST
7	6	5	4	3	2	1	0
TMR1RST	TMR0RST	Reserved					

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.
[21]	**PWM1RST**	**PWM1 Controller Reset** 0=Normal Operation. 1=Reset.
[20]	**PWM0RST**	**PWM0 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]	**SPI0 RST**	**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							

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							

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, 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. Set to other values the pin is assigned to a peripheral as outlined in table below.

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP	SYS_BA+0x38	R/W	GPIOA multiple alternate functions control register	0x0000_0000

31	30	29	28	27	26	25	24
PA15MFP		PA14 MFP		PA13MFP		PA12MFP	
23	22	21	20	19	18	17	16
PA11MFP		PA10MFP		PA9MFP		PA8MFP	
15	14	13	12	11	10	9	8
PA7MFP		PA6MFP		PA5MFP		PA4MFP	
7	6	5	4	3	2	1	0
PA3MFP		PA2MFP		PA1MFP		PA0MFP	

Bits	Description	
[31:30]	**PA15MFP**	**Alternate Function Setting For PA15MFP** 00 = GPIO. 01 = TM1. 10 = SDIN. 11 = PWM1CH1.
[29:28]	**PA14MFP**	**Alternate Function Setting For PA14MFP** 00 = GPIO. 01 = TM0. 10 = SDCLK. 11 = PWM1CH0.
[27:26]	**PA13MFP**	**Alternate Function Setting For PA13MFP** 00 = GPIO. 01 = PWM0CH1. 10 = SPKM. 11 = I2S_BCLK.
[25:24]	**PA12MFP**	**Alternate Function Setting For PA12MFP** 00 = GPIO. 01 = PWM0CH0. 10 = SPKP. 11 = I2S_FS.
[23:22]	**PA11MFP**	**Alternate Function Setting For PA11MFP** 00 = GPIO.

		01 = I2C_SCL. 10 = CMP15. 11 = UART_CTSn.
[21:20]	**PA10MFP**	**Alternate Function Setting For PA10MFP** 00 = GPIO. 01 = I2C_SDA. 10 = CMP14. 11 = UART_RTSn.
[19:18]	**PA9MFP**	**Alternate Function Setting For PA9MFP** 00 = GPIO. 01 = UART_RX. 10 = CMP13. 11 = PWM0CH3.
[17:16]	**PA8MFP**	**Alternate Function Setting For PA8MFP** 00 = GPIO. 01 = UART_TX. 10 = CMP12. 11 = PWM0CH2.
[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. 10 = UART_TX.
[11:10]	**PA5MFP**	**Alternate Function Setting For PA5MFP** 00 = GPIO. 01 = I2S_BCLK. 11 = SPI_SSB1.
[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.

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

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

Bits	Description	
[31:30]	**PB15MFP**	**Alternate Function Setting For PB15MFP** 00 = GPIO. 01 = PWM0CH3. 11 = PWM0CH2_INV (Invert output of PWM0 channel2).
[29:28]	**PB14MFP**	**Alternate Function Setting For PB14MFP** 00 = GPIO. 01 = PWM0CH2. 10 = UART_TX. 11 = PWM0CH3_INV (Invert output of PWM0 channel3).
[27:26]	**PB13MFP**	**Alternate Function Setting For PB13MFP** 00 = GPIO. 01 = SPI_MOSI1. 10 = PWM1CH1. 11 = PWM1CH0_INV (Invert output of PWM1 channel0).
[25:24]	**PB12MFP**	**Alternate Function Setting For PB12MFP** 00 = GPIO. 01 = SPI_MISO1. 10 = PWM1CH0. 11 = PWM1CH1_INV (Invert output of PWM1 channel1).
[23:22]	**PB11MFP**	**Alternate Function Setting For PB11MFP** 00 = GPIO. 01 = I2S_SDO. 10 = CMP11.

[21:20]	**PB10MFP**	**Alternate Function Setting For PB10MFP** 00 = GPIO. 10 = CMP10.
[19:18]	**PB9MFP**	**Alternate Function Setting For PB9MFP** 00 = GPIO. 01 = I2S_BCLK(master). 10 = CMP9.
[17:16]	**PB8MFP**	**Alternate Function Setting For PB8MFP** 00 = GPIO. 01 = I2S_FS(master). 10 = CMP8.
[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 = PWM0CH1_INV (Invert output of PWM0 channel1). 10 = CMP5. 11 = SPI_MISO1.
[9:8]	**PB4MFP**	**Alternate Function Setting For PB4MFP** 00 = GPIO. 01 = PWM0CH0_INV (Invert output of PWM0 channel0). 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			SPI_SSB1	O
GPIOA6	VDD33	I2S_SDI	I	UART_TX	O		
GPIOA7	VDD33	I2S_SDO	O				
GPIOA8	VCCD	UART_TX	O	CMP12	AIO	PWM0CH2	O
GPIOA9	VCCD	UART_RX	I	CMP13	AIO	PWM0CH3	O
GPIOA10	VCCD	I2C_SDA	IO	CMP14	AIO	UART_RTSn	O
GPIOA11	VCCD	I2C_SCL	IO	CMP15	AIO	UART_CTSn	I
GPIOA12	VCCD	PWM0CH0	O	SPKP	O	I2S_FS	IO
GPIOA13	VCCD	PWM0CH1	O	SPKM	O	I2S_BCLK	IO
GPIOA14	VCCD	TM0	I	SDCLK	O	PWM1CH0	O
GPIOA15	VCCD	TM1	I	SDIN	I	PWM1CH1	O

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	PWM0CH0_INV	O	CMP4	AIO	SPI_MOSI0	O
GPIOB5	VCCD	PWM0CH1_INV	O	CMP5	AIO	SPI_MISO1	I

GPIOB6	VCCD	I2S_SDI	I	CMP6	AIO	SPI_MOSI1	O
GPIOB7	VCCD	I2S_SDO	O	CMP7	AIO		
GPIOB8	VCCD	I2S_FS(master)	O	CMP8	AIO		
GPIOB9	VCCD	I2S_BCLK (master)	O	CMP9	AIO		
GPIOB10	VCCD			CMP10	AIO		
GPIOB11	VCCD	I2S_SDO	O	CMP11	AIO		
GPIOB12	VDD33	SPI_MISO1	IO	PWM1CH0	O	PWM1CH1_INV	O
GPIOB13	VDD33	SPI_MOSI1	IO	PWM1CH1	O	PWM1CH0_INV	O
GPIOB14	VCCD	PWM0CH2	O	UART_TX	O	PWM0CH3_INV	O
GPIOB15	VCCD	PWM0CH3	O			PWM0CH2_INV	

Wakeup Pin Control Register (SYS_WKCTL)

The WAKEUP pin of the ISD9300 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

31	30	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	**Reserved**	**Reserved**	**Reserved**	**WKDIN**	**WKPUEN**	**WKOENB**	**WKDOUT**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**WKDOUT**	**Wakeup Output State** Default set 0
[2]	**WKOENB**	**Wakeup Pin Output Enable Bar** 0=drive WKDOUT to pin. 1= tri-state (default).
[1]	**WKPUEN**	**Wakeup Pin Pull-Up Control** This signal is latched in deep power down and preserved. 0=pull-up enable. 1= tri-state (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 Control	0x0000_0000

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

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

Oscillator Trim Control Register (SYS_IRCTCTL)

The master oscillator of the ISD9300 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.

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

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

6.2.7 System Timer (SYST)

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

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

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

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

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

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

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

6.2.7.1 System Timer Control Register Map

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

Register	Offset	R/W	Description	Reset Value
SYSTICK Base Address: **SYSTICK_BA = 0xE000_E000**				
SYST_CSR	SYSTICK_BA+ 0x10	R/W	SYST Control and Status Register	0x0000_0004
SYST_RVR	SYSTICK_BA + 0x14	R/W	SYST Reload Value Register	0xXXXX_XXXX
SYST_CVR	SYSTICK_BA + 0x18	R/W	SYST Current Value Register	0xXXXX_XXXX

6.2.7.2 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	SYST Control and Status Register	0x0000_0004

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**	**Returns 1 If Timer Counted To 0 Since Last Time This Register Was Read** COUNTFLAG is set by a count transition from 1 to 0. COUNTFLAG is cleared on read or by a write to the Current Value register.
[15:3]	**Reserved**	Reserved.
[2]	**CLKSRC**	**System Tick Clock Source Selection** 0= Clock selected from CLK_CLKSEL0.STCLKSEL is used as clock source. 1 = Core clock used for SYST.
[1]	**TICKINT**	**System Tick Interrupt Enabled** 0 = Counting down to 0 does not cause the SYST exception to be pended. Software can use COUNTFLAG to determine if a count to zero has occurred. 1 = Counting down to 0 will cause SYST exception to be pended. Clearing the SYST Current Value register by a register write in software will not cause SYST to be pended.
[0]	**ENABLE**	**System Tick Counter Enabled** 0 = Counter Disabled. 1 = Counter will operate in a multi-shot manner.

SYST Reload Value Register（SYST_RVR）

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

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

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

SYST Current Value Register（SYST_CVR）

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

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

Bits	Description	
[31:24]	Reserved	Reserved.
[23:0]	CURRENT	System Tick Current Value 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.

6.2.8 Nested Vectored Interrupt Controller (NVIC)

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

- Nested and Vectored interrupt support
- Automatic processor state saving and restoration
- 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 an interrupt is accepted, the starting address of the interrupt service routine (ISR) is fetched from a vector table in memory. There is no need to determine which interrupt is accepted and branch to the starting address of the correlated ISR by software. While the starting address is fetched, NVIC will also automatically save processor state including the registers “PC, PSR, LR, R0~R3, R12” to the stack. At the end of the ISR, the NVIC will restore the mentioned registers from stack and resume the normal execution. Thus it will take less and deterministic time to process the interrupt request.

The NVIC supports “Tail Chaining” which handles back-to-back interrupts efficiently without the overhead of states saving and restoration and therefore reduces delay time in switching to pending ISR at the end of current ISR. The NVIC also supports “Late Arrival” which improves the efficiency of concurrent ISRs. When a higher priority interrupt request occurs before the current ISR starts to execute (at the stage of state saving and starting address fetching), the NVIC will give priority to the higher one without delay penalty. Thus it advances the real-time capability.

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

6.2.8.1 Exception Modeland System Interrupt Map

The following table lists the exception model supported by ISD9300 series. Software can set four levels of priority on some of these 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”.

Exception Name	Vector Number	Priority
Reset	1	-3
NMI	2	-2
Hard Fault	3	-1
Reserved	4 ~ 10	Reserved
SVCall	11	Configurable
Reserved	12 ~ 13	Reserved
PendSV	14	Configurable
SYST	15	Configurable
Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	Configurable

Table 6-2 Exception Model

Vector Number	Interrupt Number (Bit In Interrupt Registers)	Interrupt Name	Source Module	Interrupt Description
1 ~ 15	-	-	-	System exceptions
16	0	**BOD_IRQn**	Brown-Out	Brownout low voltage detector interrupt
17	1	**WDT_IRQn**	WDT	Watch Dog Timer interrupt
18	2	**EINT0_IRQn**	GPIO	External signal interrupt from PB.0 pin
19	3	**EINT1_IRQn**	GPIO	External signal interrupt from PB.1 pin
20	4	**GPAB_IRQn**	GPIO	External signal interrupt from PA[15:0] / PB[7:2]
21	5	**ALC_IRQn**	ALC	Automatic Level Control Interrupt
22	6	**PWM0_IRQn**	PWM0	PWM0interrupt
23	7	**PWM1_IRQn**	PWM1	PWM1 interrupt
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	UART0interrupt
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	CapSense Relaxation Oscillator Interrupt
45	29	**ADC_INT**	SDADC	Audio ADC interrupt
46	30	**-**	-	Reserved
47	31	**RTC_INT**	RTC	Real time clock interrupt

Table 6-3 System Interrupt Map

6.2.8.2 Vector Table

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

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

Table 6-4 Vector Table Format

6.2.8.3 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.

6.2.8.4 NVIC Control Register Map

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

Register	Offset	R/W	Description	Reset Value
NVIC 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

6.2.8.5 NVIC Control Register Description

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

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

Bits	Description	
[31:0]	**SETENA**	**Interrupt Enable Register** Enable one or more interrupts. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Write Operation: 0 = No effect. 1 = Write 1 to enable associated interrupt. Read Operation: 0 = Associated interrupt status is Disabled. 1 = Associated interrupt status is Enabled. Read value indicates the current enable status.

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

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

Bits	Description	
[31:0]	**CLRENA**	**Interrupt Disable Bits** Disable one or more interrupts. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Write Operation: 0 = No effect. 1 = Write 1 to disable associated interrupt. Read Operation: 0 = Associated interrupt status is Disabled. 1 = Associated interrupt status is Enabled. Read value indicates the current enable status.

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

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

Bits	Description	
[31:0]	**SETPEND**	**Set Interrupt Pending Register** Write Operation: 0 = No effect. 1 = Write 1 to set pending state. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Read Operation: 0 = Associated interrupt in not in pending status. 1 = Associated interrupt is in pending status. Read value indicates the current pending status.

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

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

Bits	Description	
[31:0]	**CLRPEND**	**Clear Interrupt Pending Register** Write Operation: 0 = No effect. 1 = Write 1 to clear pending state. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Read Operation: 0 = Associated interrupt in not in pending status. 1 = Associated interrupt is in pending status. Read value indicates the current pending status.

IRQ0 ~ IRQ3 PriorityRegister (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[1:0]		Reserved					
23	22	21	20	19	18	17	16
PRI_2[1:0]		Reserved					
15	14	13	12	11	10	9	8
PRI_1[1:0]		Reserved					
7	6	5	4	3	2	1	0
PRI_0[1:0]		Reserved					

Bits	Description	
[31:30]	**PRI_3**	**Priority Of IRQ3** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_2**	**Priority Of IRQ2** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_1**	**Priority Of IRQ1** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_0**	**Priority Of IRQ0** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ4 ~ IRQ7 PriorityRegister (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[1:0]		Reserved					
23	22	21	20	19	18	17	16
PRI_6[1:0]		Reserved					
15	14	13	12	11	10	9	8
PRI_5[1:0]		Reserved					
7	6	5	4	3	2	1	0
PRI_4[1:0]		Reserved					

Bits	Description	
[31:30]	**PRI_7**	**Priority Of IRQ7** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_6**	**Priority Of IRQ6** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_5**	**Priority Of IRQ5** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_4**	**Priority Of IRQ4** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ8 ~ IRQ11 PriorityRegister (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[1:0]		Reserved					
23	22	21	20	19	18	17	16
PRI_10[1:0]		Reserved					
15	14	13	12	11	10	9	8
PRI_9[1:0]		Reserved					
7	6	5	4	3	2	1	0
PRI_8[1:0]		Reserved					

Bits	Description	
[31:30]	**PRI_11[1:0]**	**Priority Of IRQ11** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_10[1:0]**	**Priority Of IRQ10** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_9[1:0]**	**Priority Of IRQ9** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_8[1:0]**	**Priority Of IRQ8** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ12 ~ IRQ15 PriorityRegister (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[1:0]		Reserved					
23	22	21	20	19	18	17	16
PRI_14[1:0]		Reserved					
15	14	13	12	11	10	9	8
PRI_13[1:0]		Reserved					
7	6	5	4	3	2	1	0
PRI_12[1:0]		Reserved					

Bits	Description	
[31:30]	**PRI_15**	**Priority Of IRQ15** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_14**	**Priority Of IRQ14** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_13**	**Priority Of IRQ13** "0" denotes the highest priority and "3" denotes the lowest priority
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_12**	**Priority Of IRQ12** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ16 ~ IRQ19 PriorityRegister (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[1:0]		Reserved					
23	22	21	20	19	18	17	16
PRI_18[1:0]		Reserved					
15	14	13	12	11	10	9	8
PRI_17[1:0]		Reserved					
7	6	5	4	3	2	1	0
PRI_16[1:0]		Reserved					

Bits	Description	
[31:30]	**PRI_19**	**Priority Of IRQ19** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_18**	**Priority Of IRQ18** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_17**	**Priority Of IRQ17** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_16**	**Priority Of IRQ16** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ20 ~ IRQ23 PriorityRegister (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[1:0]		Reserved					
23	22	21	20	19	18	17	16
PRI_22[1:0]		Reserved					
15	14	13	12	11	10	9	8
PRI_21[1:0]		Reserved					
7	6	5	4	3	2	1	0
PRI_20[1:0]		Reserved					

Bits	Description	
[31:30]	**PRI_23**	**Priority Of IRQ23** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_22**	**Priority Of IRQ22** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_21**	**Priority Of IRQ21** "0" denotes the highest priority and "3" denotes the lowest priority
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_20**	**Priority Of IRQ20** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ24 ~ IRQ27 PriorityRegister (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[1:0]		Reserved					
23	22	21	20	19	18	17	16
PRI_26[1:0]		Reserved					
15	14	13	12	11	10	9	8
PRI_25[1:0]		Reserved					
7	6	5	4	3	2	1	0
PRI_24[1:0]		Reserved					

Bits	Description	
[31:30]	**PRI_27**	**Priority Of IRQ27** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_26**	**Priority Of IRQ26** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_25**	**Priority Of IRQ25** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_24**	**Priority Of IRQ24** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ28 ~ IRQ31 PriorityRegister (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[1:0]		Reserved					
23	22	21	20	19	18	17	16
PRI_30[1:0]		Reserved					
15	14	13	12	11	10	9	8
PRI_29[1:0]		Reserved					
7	6	5	4	3	2	1	0
PRI_28[1:0]		Reserved					

Bits	Description	
[31:30]	**PRI_31**	**Priority Of IRQ31** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_30**	**Priority Of IRQ30** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_29**	**Priority Of IRQ29** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_28**	**Priority Of IRQ28** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

6.2.8.6 Interrupt Source Register Map

Besides the interrupt control registers associated with the NVIC, the ISD9300 series also implement some specific control registers to facilitate the interrupt functions, including “interrupt source identification”, ”NMI source selection” and “interrupt test mode”, which are described below.

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

Register	Offset	R/W	Description	Reset Value
INT Base Address: **INT_BA = 0x5000_0300**				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/B) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (ALC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWM0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (PWM1) 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

6.2.8.7 Interrupt Source Register Description

Interrupt Source IdentityRegister (IRQn_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/B) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (ALC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWM0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (PWM1) 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

31	30	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[3:0]]**			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**INT_SRC**	**Interrupt Source** Define the interrupt sources for interrupt event.

Bits	Address	INT-Num	Description
[2:0]	INT_BA+0x00	0	Bit2: 0 Bit1: 0 Bit0: BOD_INT
[2:0]	INT_BA+0x04	1	Bit2: 0 Bit1: 0 Bit0: WDT_INT
[2:0]	INT_BA+0x08	2	Bit2: 0 Bit1: 0 Bit0: EINT0_INT– external interrupt 0
[2:0]	INT_BA+0x0C	3	Bit2: 0 Bit1: 0 Bit0: EINT1_INT – external interrupt 1
[2:0]	INT_BA+0x10	4	Bit2: 0 Bit1: GPB_INT Bit0: GPA_INT
[2:0]	INT_BA+0x14	5	Bit2: 0

			Bit1: 0 Bit0: ALC_INT
[2:0]	INT_BA+0x18	6	Bit2: 0 Bit1: 0 Bit0: PWM0_INT
[2:0]	INT_BA+0x1C	7	Bit2: 0 Bit1: 0 Bit0: PWM1_INT
[2:0]	INT_BA+0x20	8	Bit2: 0 Bit1: 0 Bit0: TMR0_INT
[2:0]	INT_BA+0x24	9	Bit2: 0 Bit1: 0 Bit0: TMR1_INT
[2:0]	INT_BA+0x28	10	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x2C	11	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x30	12	Bit2: 0 Bit1: 0 Bit0: UART0_INT
[2:0]	INT_BA+0x34	13	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x38	14	Bit2: 0 Bit1: 0 Bit0: SPI0_INT
[2:0]	INT_BA+0x3C	15	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x40	16	Bit2: 0 Bit1: 0 Bit0: SPI2_INT
[2:0]	INT_BA+0x44	17	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x48	18	Bit2: 0 Bit1: 0 Bit0: I2C0_INT
[2:0]	INT_BA+0x4C	19	Bit2: 0 Bit1: 0 Bit0: 0

[2:0]	INT_BA+0x54	20	Bit2: 0 Bit1: 0 Bit0: TALARM_INT
[2:0]	INT_BA+0x58	22	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x5C	23	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x60	24	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x64	25	Bit2: 0 Bit1: 0 Bit0: ACMP_INT
[2:0]	INT_BA+0x68	26	Bit2: 0 Bit1: 0 Bit0: PDMA_INT
[2:0]	INT_BA+0x6C	27	Bit2: 0 Bit1: 0 Bit0: I2S_INT
[2:0]	INT_BA+0x70	28	Bit2: 0 Bit1: 0 Bit0: CAPS_INT
[2:0]	INT_BA+0x74	29	Bit2: 0 Bit1: 0 Bit0: ADC_INT
[2:0]	INT_BA+0x78	30	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x7C	31	Bit2: 0 Bit1: 0 Bit0: RTC_INT

NMI Source InterruptSelect 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**	**Reserved**	**NMI_SEL[4:0]**				

Bits	Description	
[31:8]	**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))
[6:5]	**Reserved**	Reserved.
[4:0]	**NMI_SEL**	**NMI Interrupt Source Selection** The NMI interrupt to Cortex™-M0 can be selected from one of the peripheral interrupt by setting NMI_SEL.

MCU InterruptRequest Source 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 Register** The MCU_IRQ collects all the interrupts from the peripherals and generates the synchronous interrupt to Cortex™-M0. There are two modes to generate interrupt to Cortex™-M0, the normal mode and test mode. The MCU_IRQ collects all interrupts from each peripheral and synchronizes them and interrupts the Cortex™-M0. When the MCU_IRQ[n] is 0: Set MCU_IRQ[n] 1 will generate an interrupt to Cortex™-M0 NVIC[n]. When the MCU_IRQ[n] is 1 (mean an interrupt is assert), setting 1 to the MCU_IRQ[n] 1 will clear the interrupt and setting MCU_IRQ[n] 0: has no effect

6.2.9 System Control

The Cortex™-M0 status and operating mode control are managed by System Control Registers. Including CPUID, Cortex™-M0 interrupt priority and Cortex™-M0 power management can be controlled through these system control registers.

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

6.2.9.1 System Control Register Map

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

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

6.2.9.2 System Control Register Description

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[11:4]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PARTNO[3:0]				**REVISION[3:0]**			

Bits	Description	
[31:24]	**IMPCODE**	**Implementer Code Assigned By ARM** Implementer code assigned by ARM. (ARM = 0x41).
[23:20]	**Reserved**	Reserved.
[19:16]	**PART**	**Architecture Of The Processor** Read as 0xC for ARMv6-M parts
[15:4]	**PARTNO**	**Part Number Of The Processor** Read as 0xC20.
[3:0]	**REVISION**	**Revision Number** Read 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	VTPNDING[8:4]				
15	14	13	12	11	10	9	8
VTPEND[3:0]				Reserved	Reserved	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 APending PendSVInterrupt** This is normally used to request a context switch. Reads back with current state (1 if Pending, 0 if not).
[27]	PPSVICLR	**Clear A Pending PendSV Interrupt** Write 1 to clear a pending PendSV interrupt.
[26]	PSTKISET	**Set A Pending SYST** Reads back with current state (1 if Pending, 0 if not).
[25]	PSTKICLR	**ClearA Pending SYST** Write 1 to clear a pending SYST.
[23]	ISRPREEM	**ISR Preemptive** If set, a pending exception will be serviced on exit from the debug halt state.
[22]	ISRPEND	**ISR Pending** Indicates if an external configurable (NVIC generated) interrupt is pending.
[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 [15:8]							
23	22	21	20	19	18	17	16
VTKEY [7:0]							
15	14	13	12	11	10	9	8
ENDIANES	Reserved						
7	6	5	4	3	2	1	0
Reserved					SRSTREQ	CLRACTVT	Reserved

Bits	Description	
[31:16]	**VTKEY**	**Vector Key** The value 0x05FA must be written to this register, otherwise a write to register is UNPREDICTABLE.
[15]	**ENDIANES**	**Endianness** Read Only. Reads 0 indicating little endian machine.
[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 configurableexceptions. **0=**do not clear state information. **1=**clear state information. The effect of writing a 1 to this bit if the processor is not halted in Debug, is UNPREDICTABLE.
[0]	**Reserved**	Reserved.

System Control Register (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			SEVONPEND	Reserved	SLPDEEP	SLPONEXC	Reserved

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**SEVNONPN**	**Send Event On Pending Bit** 0 = only enabled interrupts or events can wake-up the processor, disabled interrupts are excluded. 1 = enabled events and all interrupts, including disabled interrupts, can wake-up the processor. When enabled, interrupt transitions from Inactive to Pending are included in the list of wakeup events for the WFE instruction. When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects the next WFE. The processor also wakes up on execution of an SEV instruction.
[3]	**Reserved**	Reserved.
[2]	**SLPDEEP**	**Controls Whether The Processor Uses Sleep Or Deep Sleep As Its Low Power Mode** 0 = sleep. 1 = deep sleep. The SLPDEEP flag is also used in conjunction with CLK_PWRCTL register to enter deeper power-down states than purely core sleep states.
[1]	**SLPONEXC**	**Sleep On Exception** When set to 1, the core can enter a sleep state on an exception return to Thread mode. This is the mode and exception level entered at reset, the base level of execution. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.
[0]	**Reserved**	Reserved.

System Handler Priority Register2 (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[1:0]		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:30]	**PRI11**	**Priority Of System Handler 11 – SVCall** "0" denotes the highest priority and "3" denotes lowest priority
[29:0]	**Reserved**	Reserved.

System Handler Priority Register3 (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[1:0]		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI14[1:0]		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:30]	**PRI15**	**Priority Of System Handler 15 –SYST** "0" denotes the highest priority and "3" denotes the lowest priority
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI14**	**Priority Of System Handler 14 –PendSV** "0" denotes the highest priority and "3" denotes the lowest priority
[21:0]	**Reserved**	Reserved.

6.3 Clock Controller and Power Management Unit (PMU)

6.3.1 Overview

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.

The clock generator consists of 3 clock sources as listed below:

- An internal programmable high frequency oscillator factory trimmed to provide frequencies of 49.152MHz, 32.768MHz and 36.864MHz to 1% accuracy. On H and M speed grade devices a doubled clock frequency is available on the CLK2X output.
- An external 32kHz crystal
- An internal low power16kHz oscillator.

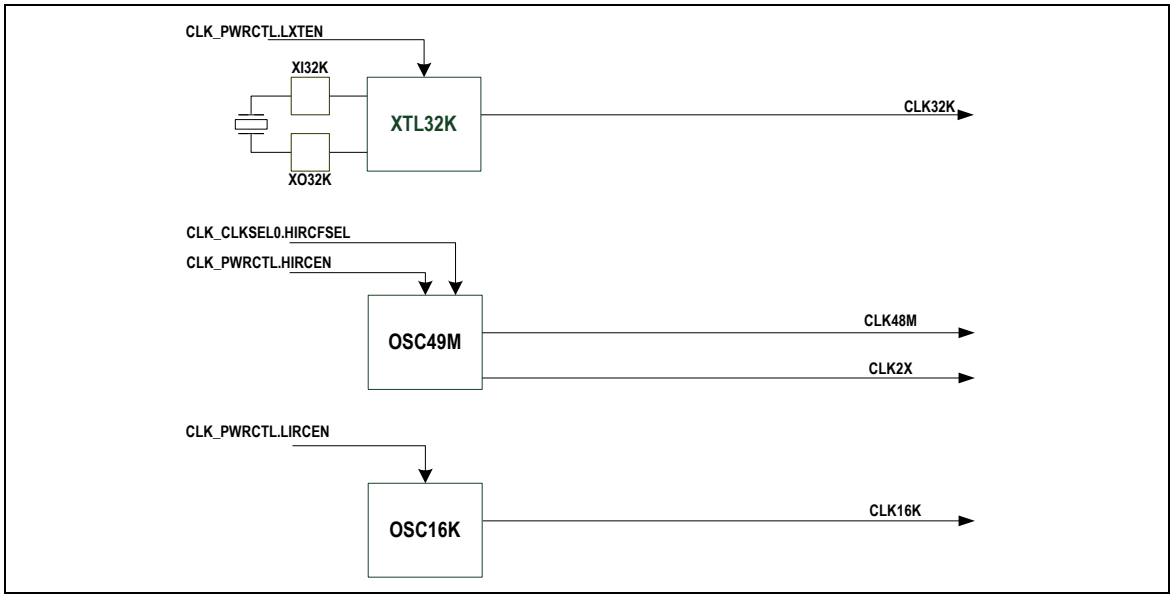


Figure 6-3 Clock Generator Block Diagram

6.3.2 System Clock &SYST Clock

The system clock has 4 clock sources from clock generator block. The clock source switch depends on the register HCLKSEL(CLK_CLKSEL0[2:0]). The clock is then divided by HCLK_N+1 to produce the master clock for the device. Note that CLK2X source is only available on M and H speed grade devices of the series. Care must be taken to set correct FMC wait state control for frequencies greater than 49.152MHz for these devices.

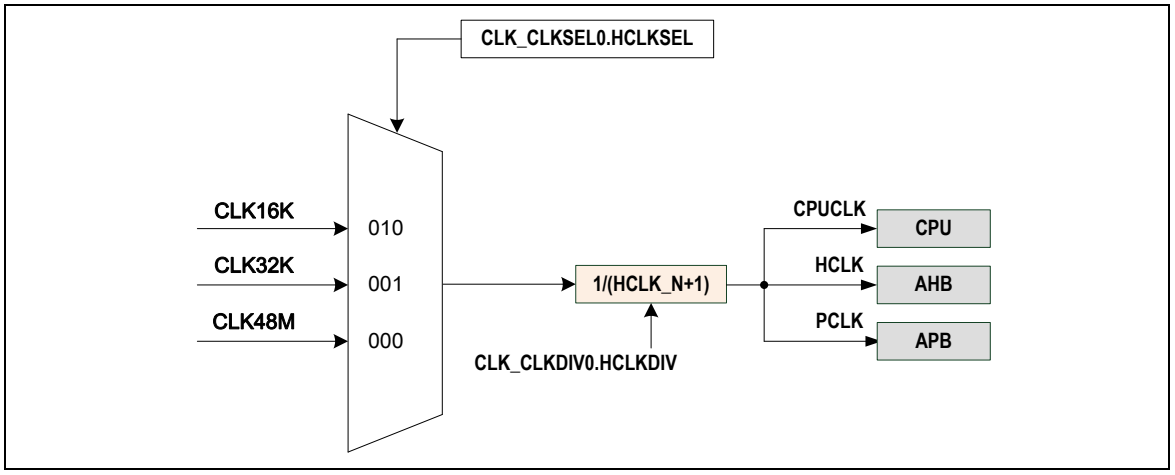


Figure 6-4 System Clock Block Diagram

The SYST clock (STCLK) has five clock sources. The clock source switch depends on the setting of the register STCLKSEL (CLK_CLKSEL0[5:3]). The block diagram is shown in Figure 6-5.

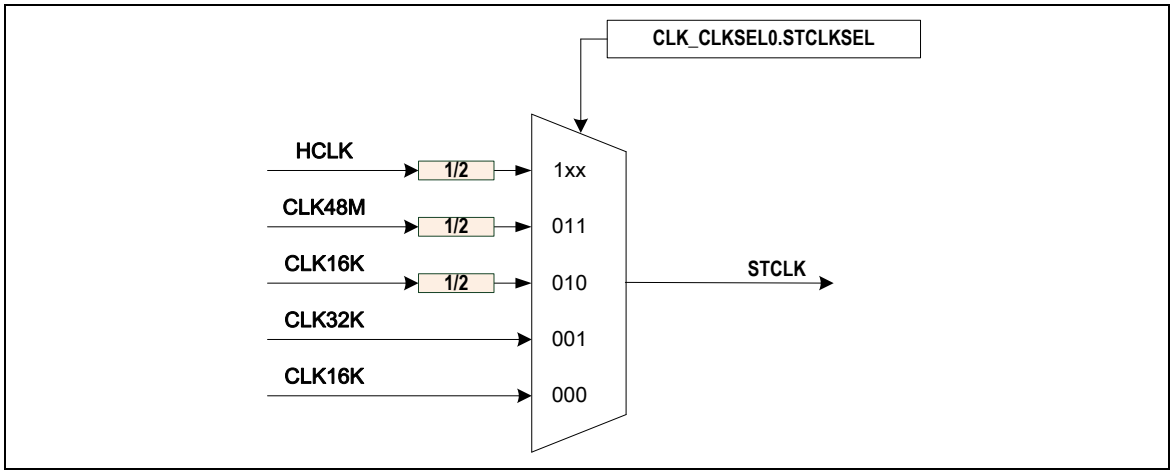


Figure 6-5 SysTick Clock Control Block Diagram

6.3.3 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.

6.3.4 Power Management

The ISD9300 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.

6.3.4.1 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:

- A high to low transition on the WAKEUP pin.
- A timed wakeup where the 16kHz oscillator is configured active and reaches a certain count.
- 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:

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

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

6.3.4.2 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:

- An interrupt from the GPIO block, for instance a pin transition.
- An interrupt from the RTC module, for instance an alarm or timer event.
- 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.

6.3.4.3 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 shut down 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: SYSCTL_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.

6.3.4.4 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.

6.3.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
CLKBase 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

6.3.6 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

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

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**	**POR Wakeup Flag** Read Only. This flag indicates that wakeup of device was requested with a power-on reset. Flag is cleared when DPD mode is entered.
[25]	**TMRWKF**	**Timer Wakeup Flag** Read Only. This flag indicates that wakeup of device was requested with TIMER count of the 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]=1: WAKEUP after 128 OSC16K clocks (12.8 ms) SELWKTMR[1]=1: WAKEUP after 256 OSC16K clocks (25.6 ms) SELWKTMR[2]=1: WAKEUP after 512 OSC16K clocks (51.2 ms) SELWKTMR[3]=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
Reserved					ISPCKEN	PDMACKEN	HCLKEN

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

APB Device Clock Enable Control Register （CLK_APBCLK0）

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

Register	Offset	R/W	Description	Reset Value
CLK_APBCLK0	CLK_BA + 0x08	R/W	APB Device Clock Enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PWM1CH01CKEN	ANACKEN	I2S0CKEN	ADCCKEN	Reserved	SBRAMCKEN	Reserved	
23	22	21	20	19	18	17	16
Reserved	ACMPCKEN	PWM0CH23CKEN	PWM0CH01CKEN	CRCCKEN	BQALCKEN	Reserved	UARTCKEN
15	14	13	12	11	10	9	8
Reserved		DPWMCKEN	SPI0CKEN	Reserved			I2C0CKEN
7	6	5	4	3	2	1	0
TMR1CKEN	TMR0CKEN	RTCCKEN	WDTCKEN	Reserved			

Bits	Description	
[31]	PWM1CH01CKEN	**PWM1CH0 And PWM1CH1 Clock Enable Control** 0=Disable. 1=Enable.
[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.
[21]	PWM0CH23CKEN	**PWM0CH2 And PWM0CH3 Block Clock Enable Control** 0=Disable. 1=Enable.

[20]	**PWM0CH01CKEN**	**PWM0CH0 And PWM0CH1 Block Clock Enable Control** 0=Disable. 1=Enable.
[19]	**CRCCKEN**	**Cyclic Redundancy Check Block Clock Enable Control** 0=Disable. 1=Enable.
[18]	**BFALCKEN**	**BiquadFilter And Automatic Level Control Block Clock Enable Control** 0=Disable. 1=Enable.
[16]	**UARTCKEN**	**UART 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							

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:6]	**HIRCFSEL**	**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 2: 36.864MHz
[5:3]	**STCLKSEL**	**MCU Cortex_M0 SYST 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 : clock source from 16kHz internal clock 001 : clock source from external 32kHz crystal clock 010 : clock source from 16kHz internal oscillator divided by 2 011 : clock source from OSC49M internal oscillator divided by 2 1xx : clock source from HCLK÷2(Default) Note that to use STCLKSEL as source of SysTic timer the CLKSRC bit of SYST_CSR must be set to 0.
[2:0]	**HCLKSEL**	**HCLK Clock Source Select** Ensure that related clock sources (pre-select and new-select) are enabled before updating register. These bits are protected, to write to bits first perform the unlock sequence (see Protected Register Lock Key Register (SYS_REGLCTL)) 000 : clock source from internal OSC48M oscillator. 001 : clock source from external 32kHz crystal clock 010 : clock source from internal 16kHz oscillator clock 100: CLK2X a frequency doubled output of OSC48M, only available on M and H speed grade devices. Others : RESERVED

Clock Source Select Control Register 1（CLK_CLKSEL1）

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

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL1	CLK_BA + 0x14	R/W	Clock Source Select Control Register 1	0x3300_771F

31	30	29	28	27	26	25	24
PWM0CH23CKSEL		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	Reserved		WDTSEL	

Bits	Description	
[31:30]	**PWM0CH23CKSEL**	**PWM0CH2 And PWM0CH3 Clock Source Select** PWM0CH2 and PWM0CH3 uses the same clock source, and pre-scaler 00 = clock source from internal 16kHz oscillator. 01 = clock source from external 32kHz crystal clock. 10 = clock source from HCLK. 11 = clock source from internal OSC48M oscillator clock.
[29:28]	**PWM0CH01CKSEL**	**PWM0CH0 And PWM0CH1 Clock Source Select** PWM0CH0 and PWM0CH1 uses the same clock source, and pre-scaler 00 = clock source from internal 16kHz 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 = clock source from internal 16kHz oscillator 001 = clock source from external 32kHz crystal clock 010 = clock source from HCLK 011 = clock source from external pin (GPIOA[15]) 1xx = clock source from internal OSC48M oscillator clock
[10:8]	**TMR0SEL**	**TIMER0 Clock Source Select** 000 = clock source from internal 16kHz oscillator 001 = clock source from external 32kHz crystal clock 010 = clock source from HCLK 011 = clock source from external pin (GPIOA[14])

		1xx = 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 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 16kHz 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**			

Bits	Description	
[23:16]	**ADCDIV**	**ADC Clock Divide Number From ADC Clock Source** The ADC clock frequency = (ADC clock source frequency) / (ADC_N + 1)
[11:8]	**UARTDIV**	**UART Clock Divide Number From UART Clock Source** The UART clock frequency = (UART clock source frequency) / (UART_N + 1)
[3:0]	**HCLKDIV**	**HCLK Clock Divide Number From HCLK Clock Source** The HCLK clock frequency = (HCLK clock source frequency) / (HCLK_N + 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		PWM1CH01CKSEL		Reserved		I2S0SEL	

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:4]	**PWM1CH01CKSEL**	**PWM1CH0 And PWM1CH1 Clock Source Select** PWM1CH0 and PWM1CH1 uses the same clock source, and pre-scaler 00 = clock source from internal 16kHz oscillator. 01 = clock source from external 32kHz crystal clock. 10 = clock source from HCLK. 11 = clock source from internal OSC48M oscillator clock.
[3:2]	**Reserved**	Reserved.
[1:0]	**I2S0SEL**	**I2S0 Clock Source Select** 00 = clock source from internal 16kHz 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

31	30	29	28	27	26	25	24
PWM1CH01CKEN	ANACKEN	I2SCKEN	ADCCKEN	Reserved	SBRAMCKEN	Reserved	
23	22	21	20	19	18	17	16
Reserved	ACMPCKEN	PWM0CH23CKEN	PWM0CH01CKEN	CRCCKEN	BQALCKEN	Reserved	UARTCKEN
15	14	13	12	11	10	9	8
Reserved		DPWMCKEN	SPI0CKEN	Reserved			I2C0CKEN
7	6	5	4	3	2	1	0
TMR1CKEN	TMR0CKEN	RTCCKEN	WDTCKEN	Reserved	ISPCKEN	PDMACKEN	HCLKCKEN

Bits	Description	
[31]	PWM1CH01CKEN	**PWM1CH0 And PWM1CH1 Block Sleep Clock Enable Control** 0=Disable. 1=Enable.
[30]	ANACKEN	**Analog Block Sleep Clock Enable Control** 0=Disable. 1=Enable.
[29]	I2SCKEN	**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.
[21]	PWM0CH23CKEN	**PWM0CH2 And PWM0CH3 Block Sleep Clock Enable Control**

		0=Disable. 1=Enable.
[20]	**PWM0CH01CKEN**	**PWM0CH0 And PWM0CH1 Block Sleep Clock Enable Control** 0=Disable. 1=Enable.
[19]	**CRCCKEN**	**Cyclic Redundancy Check Sleep Block Clock Enable Control** 0=Disable. 1=Enable.
[18]	**BQALCKEN**	**Biquad Filter/ALC Block Sleep Clock Enable Control** 0=Disable. 1=Enable.
[16]	**UARTCKEN**	**UART 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]	**HCLKCKEN**	**CPU Clock Sleep Enable (HCLK)** Must be left as ‘1’ for normal operation. 0=Disable.

		1=Enable.

Power State Flag Register (CLK_PWRSTSF)

Register	Offset	R/W	Description	Reset Value
CLK_PWRSTSF	CLK_BA+ 0x24	R/W	Power State Flag Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**SPDF**	**Powered Down Flag** This flag is set if core logic was powered down to Standby (SPD). Write ‘1’ to clearflag.
[1]	**STOPF**	**Stop Flag** This flag is set if core logic was stopped but not powered down. Write ‘1’ to clear flag.
[0]	**DSF**	**Deep Sleep Flag** This flag is set if core logic was placed in Deep Sleep mode. Write ‘1’ to clear flag.

Debug Power Down Register (CLK_DBGPD)

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

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

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

6.4 General Purpose I/O (GPIO)

6.4.1 Overview

The ISD9300 series has up to 32 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 84 pins are arranged in 6 ports named as GPIOA and GPIOB. Each of the 32 pins is independent and has the corresponding register bits to control the pin mode function and data.

The I/O type of each of I/O pins can be configured by software individually as input, output, open-drain or 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. All other GPIO generate and exception to interrupt number IRQ4.

6.4.2 Features

- Four I/O modes:
  - Quasi-bidirectional
  - Push-Pull output
  - Open-Drain output
  - Input only with high impendence
- I/O pin configured as interrupt source with edge/level setting
- I/O pin internal pull-up resistor enabled only in Quasi-bidirectional I/O mode

6.4.3 Basic Configuration

The GPIO pin functions are configured in SYS_GPA_MFP andSYS_GPB_MFPregisters.

6.4.4 Functional Description

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

6.4.4.1 Input Mode Explanation

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

6.4.4.2 Push-pull Output Mode Explanation

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

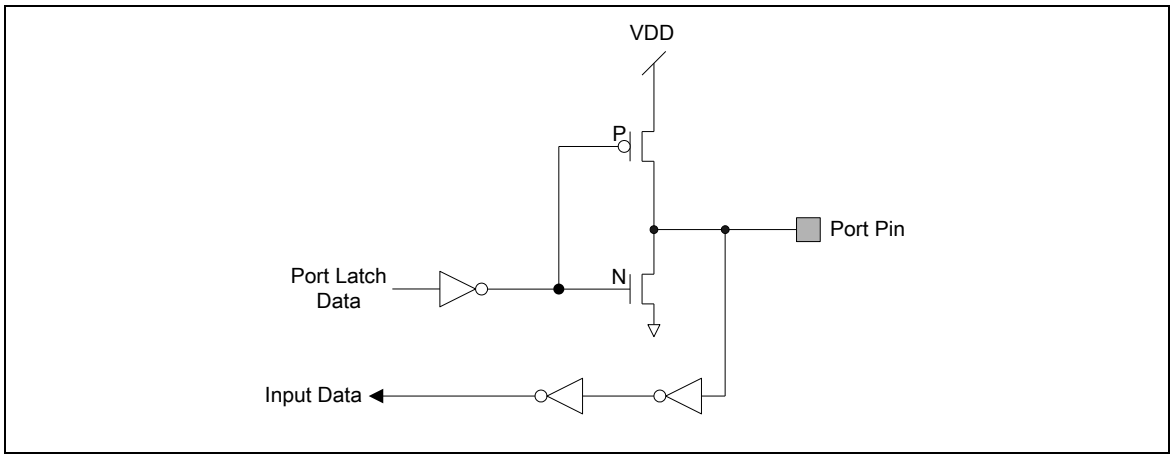


Figure 6-6 Push-Pull Output

6.4.4.3 Open-drain Output Mode Explanation

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

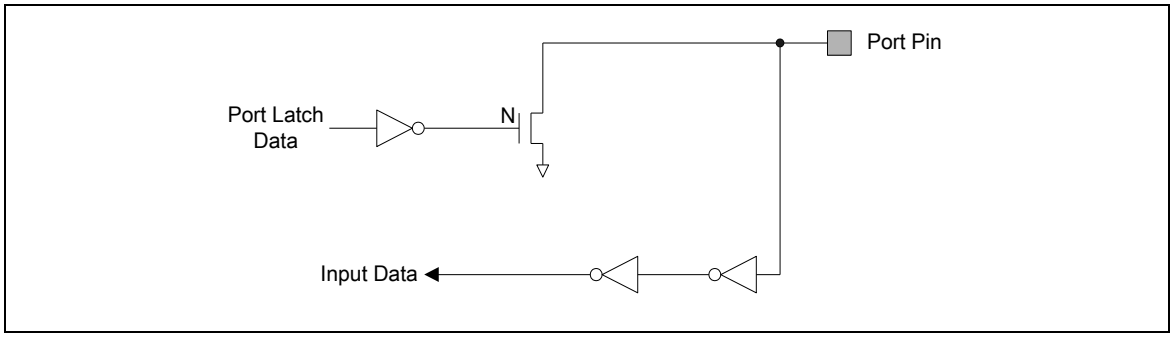


Figure 6-7 Open-Drain Output

6.4.4.4 Quasi-bidirectional Mode Explanation

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

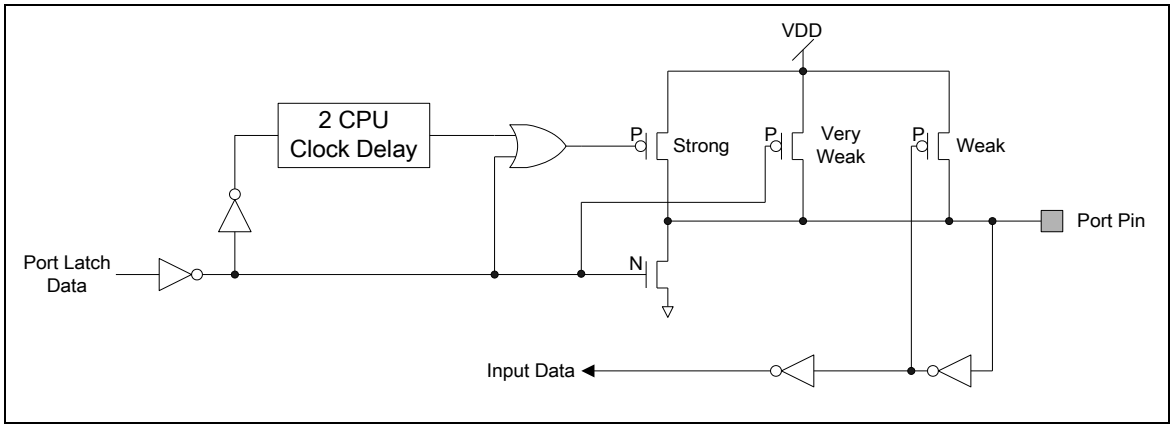


Figure 6-8 Quasi-bidirectional I/O Mode

6.4.4.5 GPIO Interrupt and Wake-up Function

Each GPIO pin can be set as chip interrupt source by setting correlative Px_INTEN bit and Px_INTTYPE. There are four types of interrupt condition can be selected: low level trigger, high level trigger, falling edge trigger and rising edge trigger. For edge trigger condition, user can enable input signal de-bounce function to prevent unexpected interrupt happened which caused by noise. The de-bounce clock source and sampling cycle can be set through GPIO_DBCTL register.

The GPIO can also be the chip wake-up source when chip enters Idle mode or Power-down mode. The setting of wake-up trigger condition is the same as GPIO interrupt trigger, but there isone thing need to be noticed if using GPIO as chip wake-up source

- **To ensure the I/O status before enter into Power-down mode**

When using toggle GPIO to wake-up system, user must make sure the I/O status before entering Idle mode or Power-down mode according to the relative wake-up settings.

For example, if configuring the wake-up event occurred by I/O rising edge/high level trigger, user must make sure the I/O status of specified pin is at low level before entering to Idle/Power-down mode; and if configure I/O falling edge/low level trigger to trigger a wake-up event, user must make sure the I/O status of specified pin is at high level before entering to Power-down mode.

6.4.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
GPIOBase Address: **GPIO_BA = 0x5000_4000**				
PA_MODE	GPIO_BA+0x000	R/W	GPIO Port A Pin I/O Mode Control	0xFFFF_FFFF
PA_DINOFF	GPIO_BA+0x004	R/W	GPIO Port A Pin Input Disable	0x0000_0000
PA_DOUT	GPIO_BA+0x008	R/W	GPIO Port A Data Output Value	0x0000_FFFF
PA_DATMSK	GPIO_BA+0x00C	R/W	GPIO Port A Data Output Write Mask	0xXXXX_0000
PA_PIN	GPIO_BA+0x010	R	GPIO Port A Pin Value	0x0000_XXXX
PA_DBEN	GPIO_BA+0x014	R/W	GPIO Port A De-bounce Enable	0xXXXX_0000
PA_INTTYPE	GPIO_BA+0x018	R/W	GPIO Port A Interrupt Trigger Type	0xXXXX_0000
PA_INTEN	GPIO_BA+0x01C	R/W	GPIO Port A Interrupt Enable	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	GPIO Port A Interrupt Source Flag	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	GPIO Port B Pin I/O Mode Control	0xXXXX_FFFF
PB_DINOFF	GPIO_BA+0x044	R/W	GPIO Port B Pin Input Disable	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	GPIO Port B Data Output Value	0x0000_FFFF
PB_DATMSK	GPIO_BA+0x04C	R/W	GPIO Port B Data Output Write Mask	0xXXXX_0000
PB_PIN	GPIO_BA+0x050	R	GPIO Port B Pin Value	0x0000_XXXX
PB_DBEN	GPIO_BA+0x054	R/W	GPIO Port B De-bounce Enable	0xXXXX_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	GPIOPortBInterrupt Trigger Type	0xXXXX_0000
PB_INTEN	GPIO_BA+0x05C	R/W	GPIO Port B Interrupt Enable	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	GPIO Port B Interrupt Source Flag	0x0000_0000
GPIO_DBCTL	GPIO_BA+0x180	R/W	Interrupt De-bounce Control	0x0000_0020

6.4.6 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	0xXXXX_FFFF

31	30	29	28	27	26	25	24
PMD15		PMD14		PMD13		PMD12	
23	22	21	20	19	18	17	16
PMD11		PMD10		PMD9		PMD8	
15	14	13	12	11	10	9	8
PMD7		PMD6		PMD5		PMD4	
7	6	5	4	3	2	1	0
PMD3		PMD2		PMD1		PMD0	

Bits	Description	
[2n+1 :2n] n=0,1..15	**MODEn**	**Px I/O Pin[N] Mode Control** Determine each I/O type of GPIOx pins 00 = GPIO port [n] pin is in INPUT mode. 01 = GPIO port [n] pin is in OUTPUT mode. 10 = GPIO port [n] pin is in Open-Drain mode. 11 = GPIO port [n] pin is in Quasi-bidirectional mode.

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

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

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

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

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

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	GPIO Port A Data Output Value	0x0000_FFFF
PB_DOUT	GPIO_BA+0x048	R/W	GPIO Port B Data Output Value	0x0000_FFFF

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

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

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

Register	Offset	R/W	Description	Reset Value
PA_DATMSK	GPIO_BA+0x00C	R/W	GPIO Port A Data Output Write Mask	0xXXXX_0000
PB_DATMSK	GPIO_BA+0x04C	R/W	GPIO Port B Data Output Write Mask	0xXXXX_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[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 DOUTn bit is writing protected. 0 = The corresponding Px_DOUT[n] bit can be updated. 1 = The corresponding Px_DOUT[n] bit is read only.

GPIO Port [A/B] Pin Value (Px _PIN)

Register	Offset	R/W	Description	Reset Value
PA_PIN	GPIO_BA+0x010	R	GPIO Port A Pin Value	0x0000_XXXX
PB_PIN	GPIO_BA+0x050	R	GPIO Port B Pin Value	0x0000_XXXX

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n = 0,1..15	**PIN[n]**	**Port [A/B] Pin Values** The value read from each of these bit reflects the actual status of the respective GPIO pin

GPIO Port [A/B] De-Bounce Enable (Px _DBEN)

Register	Offset	R/W	Description	Reset Value
PA_DBEN	GPIO_BA+0x014	R/W	GPIO Port A De-bounce Enable	0xXXXX_0000
PB_DBEN	GPIO_BA+0x054	R/W	GPIO Port B De-bounce Enable	0xXXXX_0000

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

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

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

Register	Offset	R/W	Description	Reset Value
PA_INTTYPE	GPIO_BA+0x018	R/W	GPIO Port A Interrupt Trigger Type	0xXXXX_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	GPIO Port B Interrupt Trigger Type	0xXXXX_0000

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

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

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

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

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

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

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

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

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

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

Interrupt De-Bounce Control (GPIO_DBCTL)

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

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

Bits	Description	
[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**

DBCLKSEL	Description
0	Sample interrupt input once per 1 clocks
1	Sample interrupt input once per 2 clocks
2	Sample interrupt input once per 4 clocks
3	Sample interrupt input once per 8 clocks
4	Sample interrupt input once per 16 clocks
5	Sample interrupt input once per 32 clocks
6	Sample interrupt input once per 64 clocks
7	Sample interrupt input once per 128 clocks
8	Sample interrupt input once per 256 clocks
9	Sample interrupt input once per 2*256 clocks
10	Sample interrupt input once per 4*256clocks
11	Sample interrupt input once per 8*256 clocks
12	Sample interrupt input once per 16*256 clocks

		13	Sample interrupt input once per 32*256 clocks	
		14	Sample interrupt input once per 64*256 clocks	
		15	Sample interrupt input once per 128*256 clocks	

6.5 Brownout Detection and Temperature Alarm

6.5.1 Overview

The ISD9300 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.

6.5.2 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

6.5.3 Register Description

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

31	30	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			**BODRANGE**	**BODHYS**	**BODVL**		

Bits	Description		
[31:5]	**Reserved**	Reserved.	
[4]	**BODRANGE**	**Range** Range setting for BODVL	
[3]	**BODHYS**	**BOD Hysteresis** 0=Hysteresis Disabled. 1= Enable Hysteresis of BOD detection.	
[2:0]	**BODVL**	**RANGE=0**	**RANGE=1**
		111b =4.6V 110b =3.0V 101b =2.8V 100b=2.65V 011b=2.5V 010b =2.4V 001b =2.2V 000b =2.1V	111b =4.2V 110b =3.9V 101b =3.8V 100b =3.7V 011b =3.6V 010b =3.4V 001b =3.2V 000b =3.1V

Brown-Out Detector Enable Register (BODTALM_BODCTL)

This register is initialized by user flash configuration bit config0[23]. If config0[23]=1, then reset value of BODTALM_BODCTL 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

31	30	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			**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. BODOUT=1 implies that VCC is less than BODVL.
[3]	**BODIF**	**Current Status Of Interrupt** Latched whenever a BOD event occurs and IE=1. Write '1' to clear.
[2]	**BODINTEN**	**BOD Interrupt Enable** 0= Disable BOD Interrupt. 1= Enable BOD Interrupt.
[1:0]	**BODEN**	**BOD Enable** 1xb =Enable continuous BOD detection. 01b =Enable time multiplexed BOD detection. See BODTALM_BODDTMR register. 00b =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

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

31	30	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				**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

31	30	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				**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=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 =DisableTALARM Detection. 1 = Enable TALARM Detection.

6.6 I²C Serial Interface Controller (I²C)

6.6.1 Overview

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.

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 6-9 I²C Bus Timingfor more detail I2C BUS Timing.

6.6.2 Features

The I²C bus uses two wires (I2Cn_SDA and I2Cn_SCL) to transfer information between devices connected to the bus. The main features of the I²C bus include:

- 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 allow devices with different bit rates to communicate via one serial bus
- Built-in a 14-bit time-out counter requesting the I²C interrupt if the I²C bus hangs up and timer-out counter overflows.
- Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer
- External pull-up are needed for high output
- Programmable clocks allow versatile rate control
- Supports 7-bit addressing mode
- Supports multiple address recognition (four slave address with mask option)

6.6.3 Functional Description

On I²C bus, data is transferred between a Master and a Slave. Data bits transfer on the I2Cn_SCL and I2Cn_SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit long. There is one I2Cn_SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of I2Cn_SCL; therefore, the I2Cn_SDA line may be changed only during the low period of I2Cn_SCL and must be held stable during the high period of I2Cn_SCL. A transition on the

I2Cn_SDA line while I2Cn_SCL is high is interpreted as a command (START or STOP). Please refer to the following figure for more detailed I^{2}C bus timing.

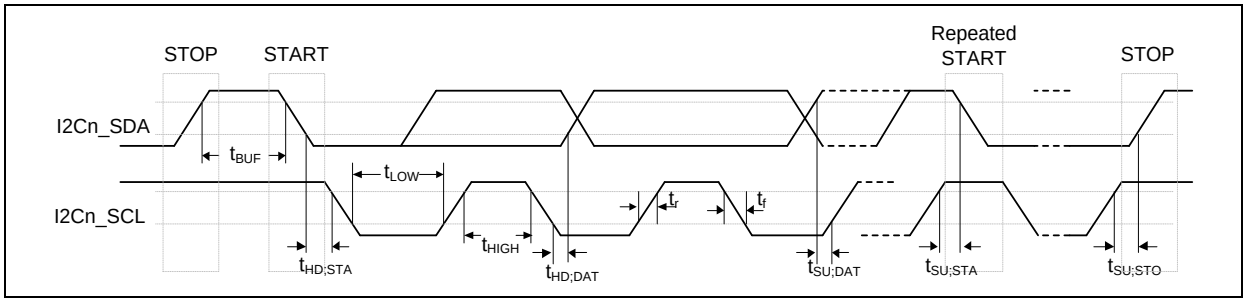


Figure 6-9 I^{2}C Bus Timing

The device's on-chip I^{2}C provides the serial interface that meets the I^{2}C bus standard mode specification. The I^{2}C port handles byte transfers autonomously. The I^{2}C hardware interfaces to the I^{2}C bus via two pins: I2Cn_SDA and I2Cn_SCL. When I/O pins are used as I^{2}C ports, user must set the pins function to I^{2}C in advance.

Note: Pull-up resistor is needed for I^{2}C operation as the I2Cn_SDA and I2Cn_SCL are open-drain pins.

6.6.3.1 I^{2}C Protocol

The following figure shows the typical I^{2}C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

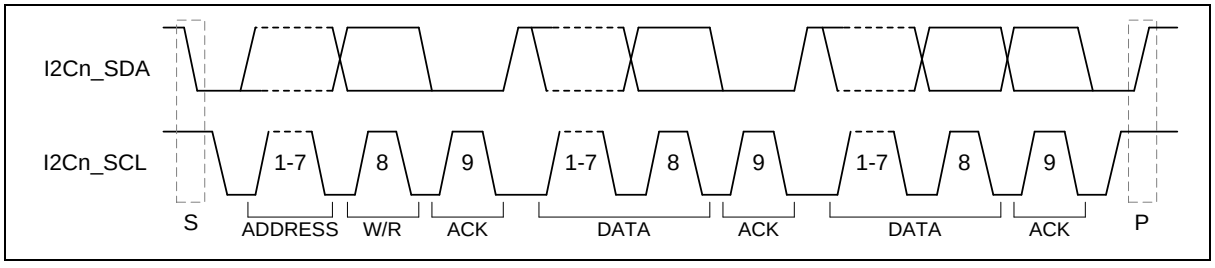


Figure 6-10 I^{2}C Protocol

6.6.3.1.1 *START or Repeated START signal*

When the bus is free or idle, meaning no master device is engaging the bus (both I2Cn_SCL and I2Cn_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 I2Cn_SDA line while I2Cn_SCL is HIGH. The START signal denotes the beginning of a new data transmission.

After having sent the address byte (address and read/write bit) the master may send any number

of bytes followed by a stop condition. Instead of sending the stop condition it is also allowed to send another start condition again followed by an address (and of course including a read/write bit) and more data. The start condition is called as Repeat START (Sr). This is defined recursively allowing any number of start conditions to be sent. The purpose of this is to allow combined write/read operations to one or more devices without releasing the bus and thus with the guarantee that the operation is not interrupted. The controller 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.

6.6.3.1.2 *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 I2Cn_SDA line while I2Cn_SCL is HIGH.

The following figure shows the waveform of START, Repeat START and STOP.

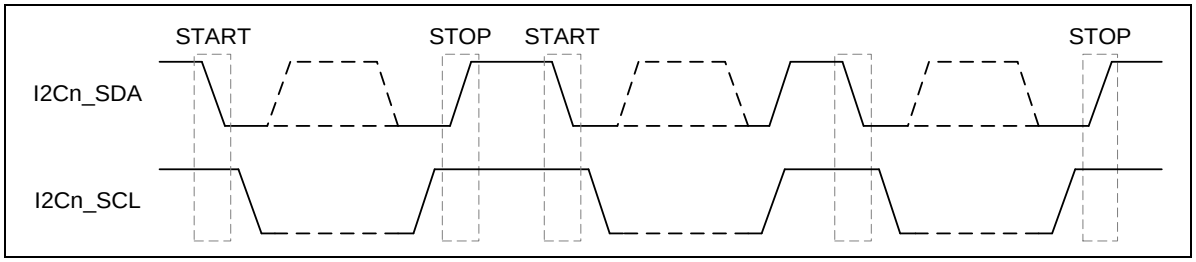


Figure 6-11 START and STOP Conditions

6.6.3.1.3 *Slave Address Transfer*

The first byte of data transferred by the master immediately after the START signal is the Slave address (SLA). This is a 7-bit calling address followed by a Read/Write (R/W) bit. The R/W bit signals of the slave indicate 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 I2Cn_SDA low at the 9th I2Cn_SCL clock cycle.

6.6.3.1.4 *Data Transfer*

When a slave receives a correct address with an R/W bit, the data will follow R/W bit specified to transfer. Each transferred byte is followed by an acknowledge bit on the 9th I2Cn_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 a receiving device, does Not Acknowledge (NACK) the slave, the slave releases the I2Cn_SDA line for the master to generate a STOP or Repeated START signal.

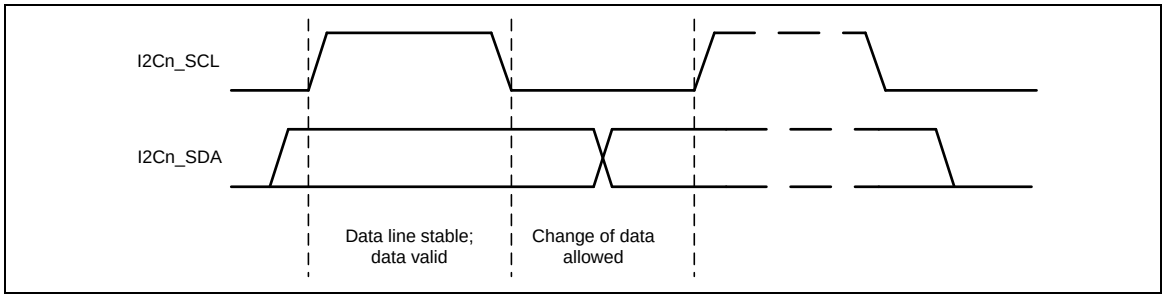


Figure 6-12 Bit Transfer on the I^2C Bus

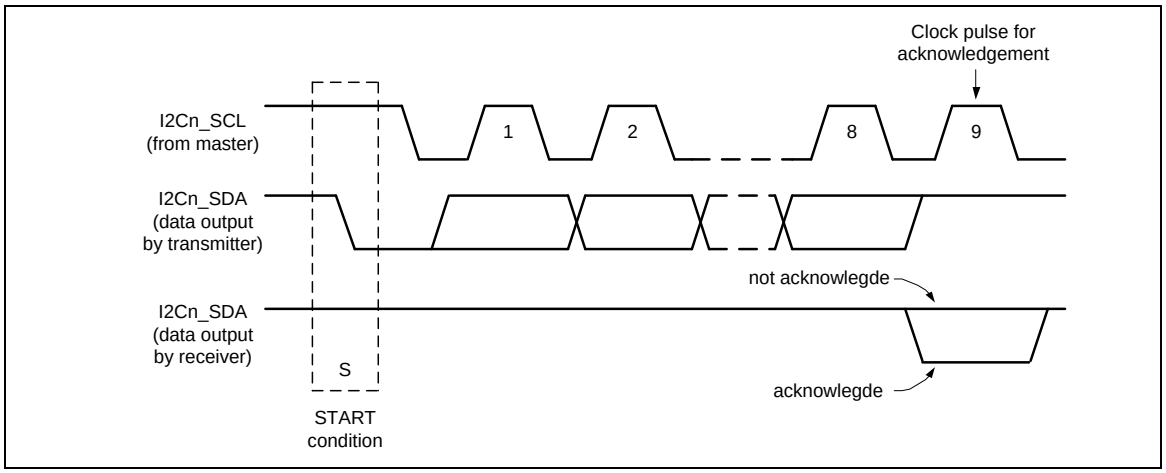


Figure 6-13 Acknowledge on the I^{2}C Bus

6.6.3.1.5 *Data transfer on the I^{2}C bus*

The following figure shows a master transmits data to slave. A master addresses a slave with a 7-bit address and 1-bit write index to denote that the master wants to transmit data to the slave. The master keeps transmitting data after the slave returns acknowledge to the master.

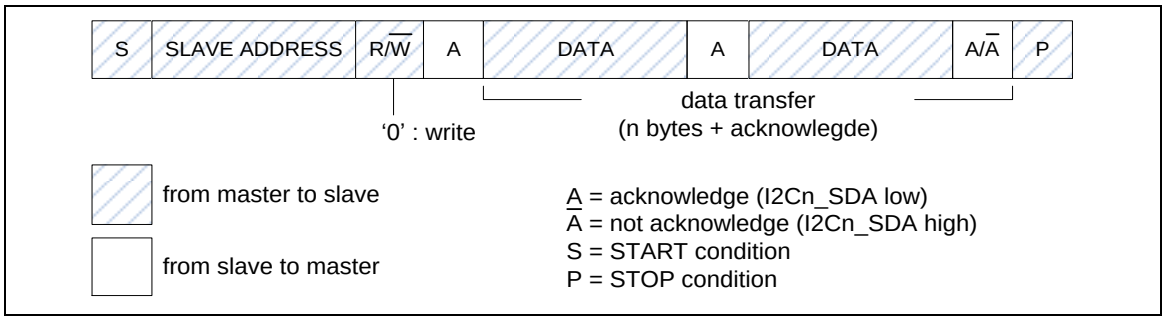


Figure 6-14 Master Transmits Data to Slave

The following figure shows a master read data from slave. A master addresses a slave with a 7-bit address and 1-bit read index to denote that the master wants to read data from the slave. The slave will start transmitting data after the slave returns acknowledge to the master.

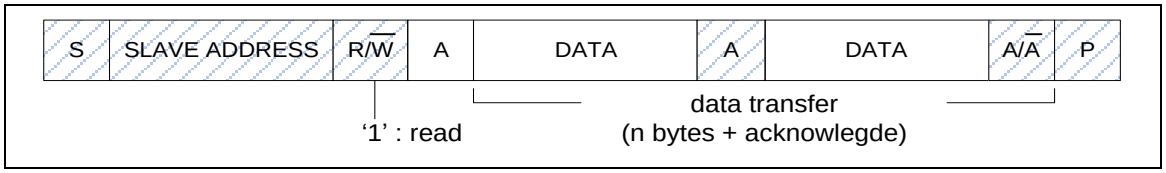


Figure 6-15 Master Reads Data from Slave

6.6.3.2 Operation Modes

The on-chip I^{2}C ports support three operation modes, Master, Slave, and General Call Mode.

In a given application, I²C port may operate as a master or as a slave. In 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), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer.

To control the I²C bus transfer in each mode, user needs to set I2C_CTL, I2C_DAT registers according to current status code of I2C_STATUS register. In other words, for each I²C bus action, user needs to check current status by I2C_STATUS register, and then set I2C_CTL, I2C_DAT registers to take bus action. Finally, check the response status by I2C_STATUS.

The bits, STA(I2C_CTL[5]), STO(I2C_CTL[4]) and AA(I2C_CTL[2]) are used to control the next state of the I²C hardware after SI (I2C_CTL[3])flag is cleared. Upon completion of the new action, a new status code will be updated in I2C_STATUS register 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.

The following figure shows the current I²C status code is 0x08, and then set I2C_DATA=SLA+W and (STA,STO,SI,AA) = (0,0,1,x) to send the address to I²C bus. If a slave on the bus matches the address and response ACK, the I2C_STATUS will be updated by status code 0x18.

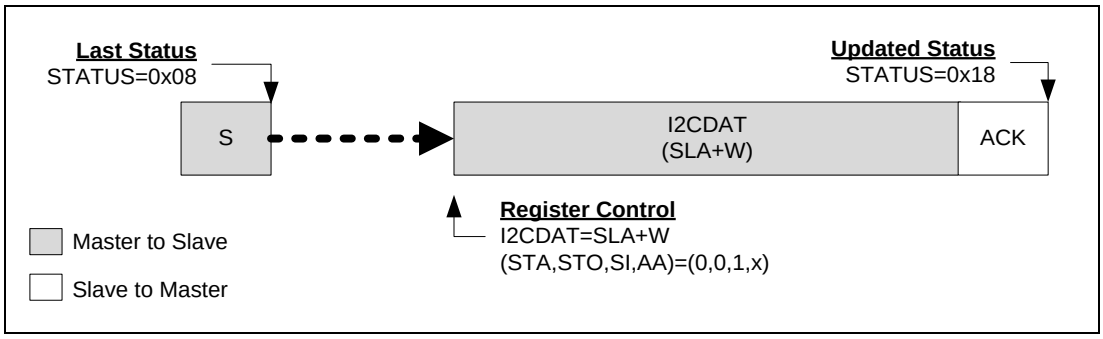


Figure 6-16 Control I²C Bus according to Current I²C Status

6.6.3.2.1 *Master Mode*

In below figures, all possible protocols for I^{2}C master are shown. User needs to follow proper path of the flow to implement required I^{2}C protocol.

In other words, user can send a START signal to bus and I^{2}C will be in Master Transmitter mode (Figure 6-17) or Master receiver mode (Figure 6-19) after START signal has been sent successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

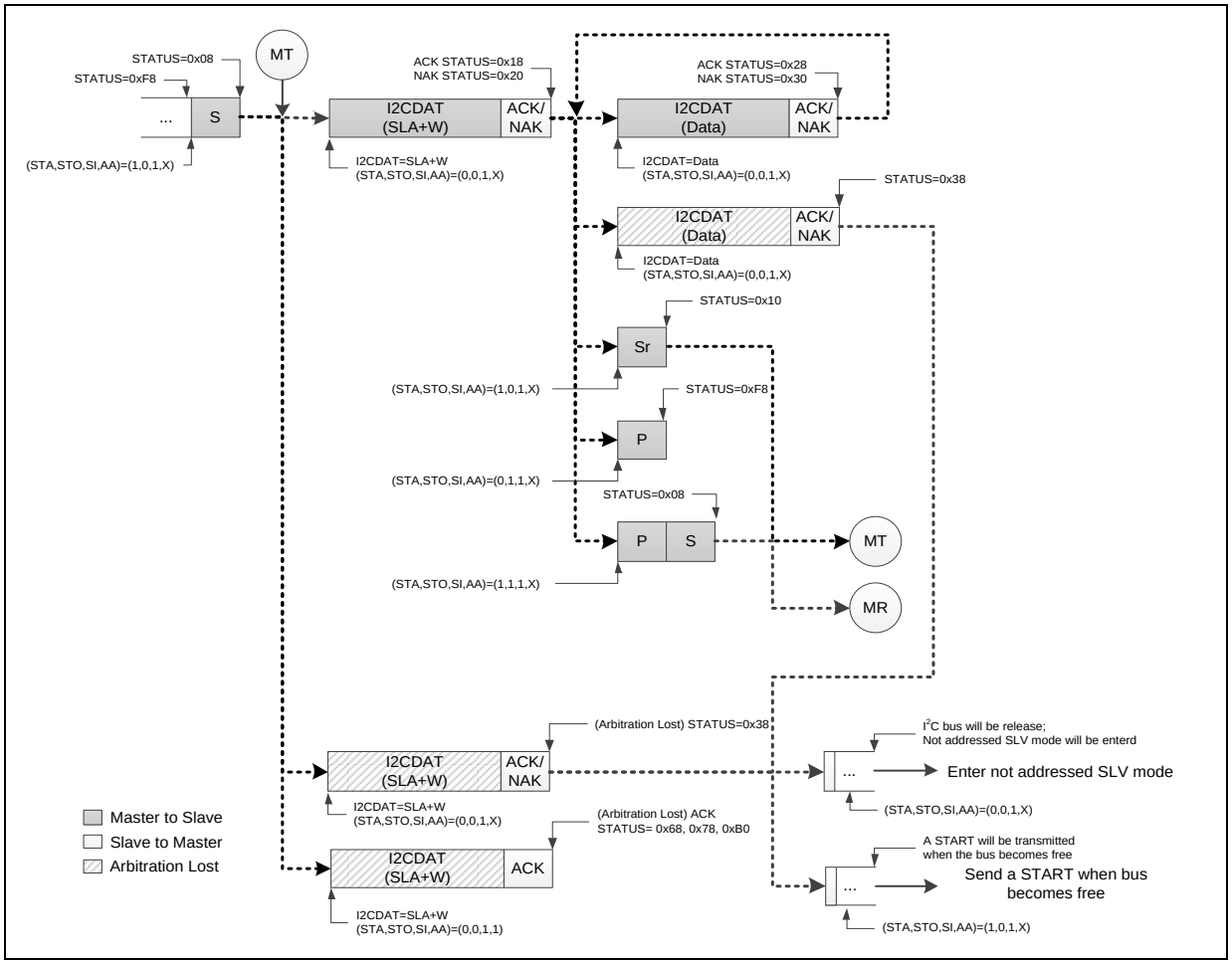


Figure 6-17 Master Transmitter Mode Control Flow

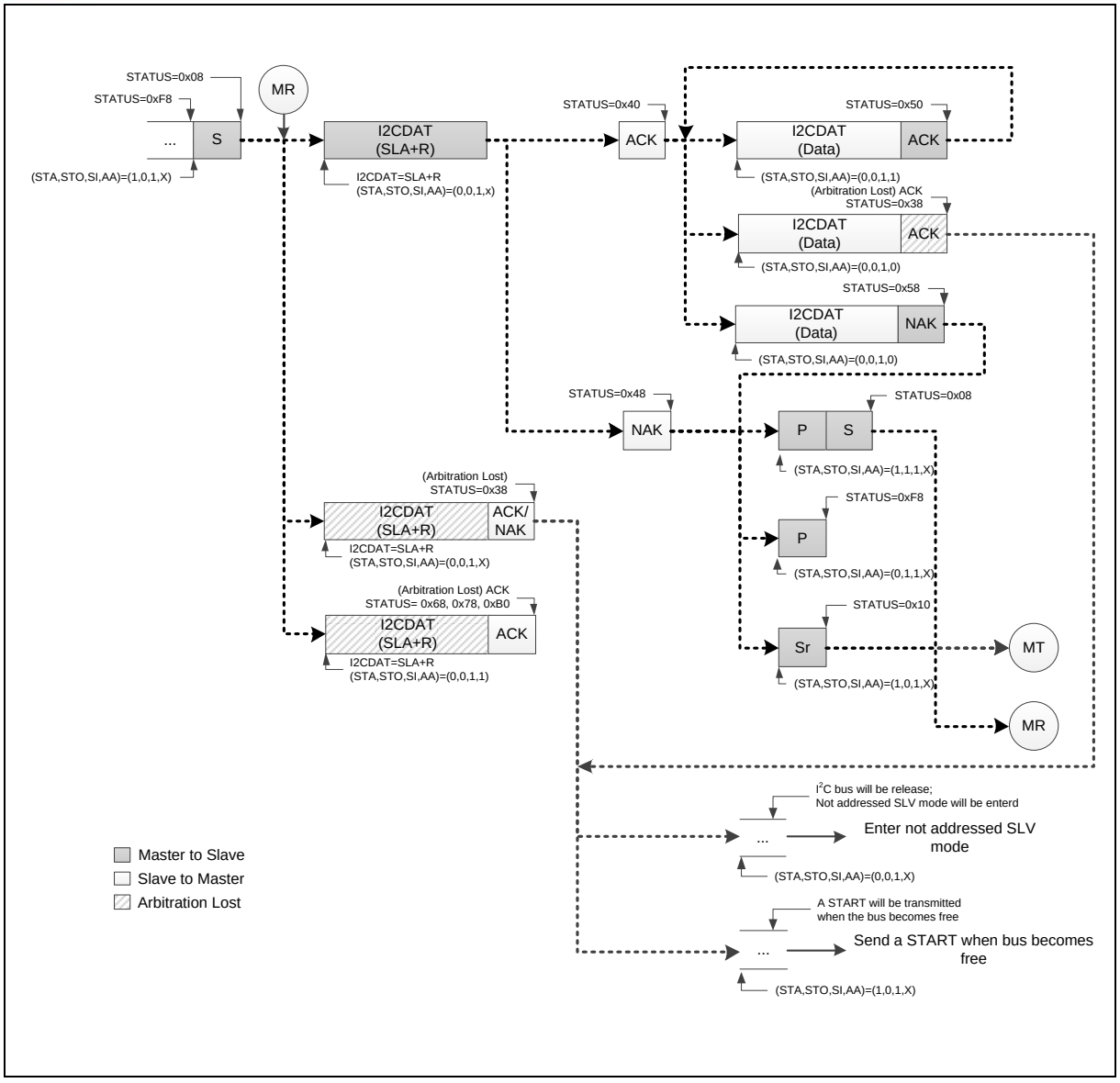


Figure 6-18 Master Receiver Mode Control Flow

If the I²C is in Master mode and gets arbitration lost, the status codewill be 0x38. In status 0x38, user may set (STA, STO, SI, AA) = (1, 0, 1, X) to send START to re-start Master operation when bus become free. Otherwise, user may set (STA, STO, SI, AA) = (0, 0, 1, X) to release I²C bus and enter not addressed Slave mode.

6.6.3.2.2 *Slave Mode*

When reset default, I²C is not addressed and will not recognize the address on I²C bus. User can set slave address by I2C_ADDRx and set (STA, STO, SI, AA) = (0, 0, 1, 1) to let I²C recognize the address sent by master. Figure 6-19 shows all the possible flow for I²C in Slave mode. Users need to follow a proper flow (as shown in Figure 6-19 to implement their own I²C protocol.

If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can

detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) after arbitration lost, the status code is 0x68. If the detected address is SLA+R (Master want to read data from Slave) after arbitration lost, the status code is 0xB0.

Note: During I^{2}C communication, the I2Cn_SCL clock will be released when writing '1' to clear SI flag in Slave mode.

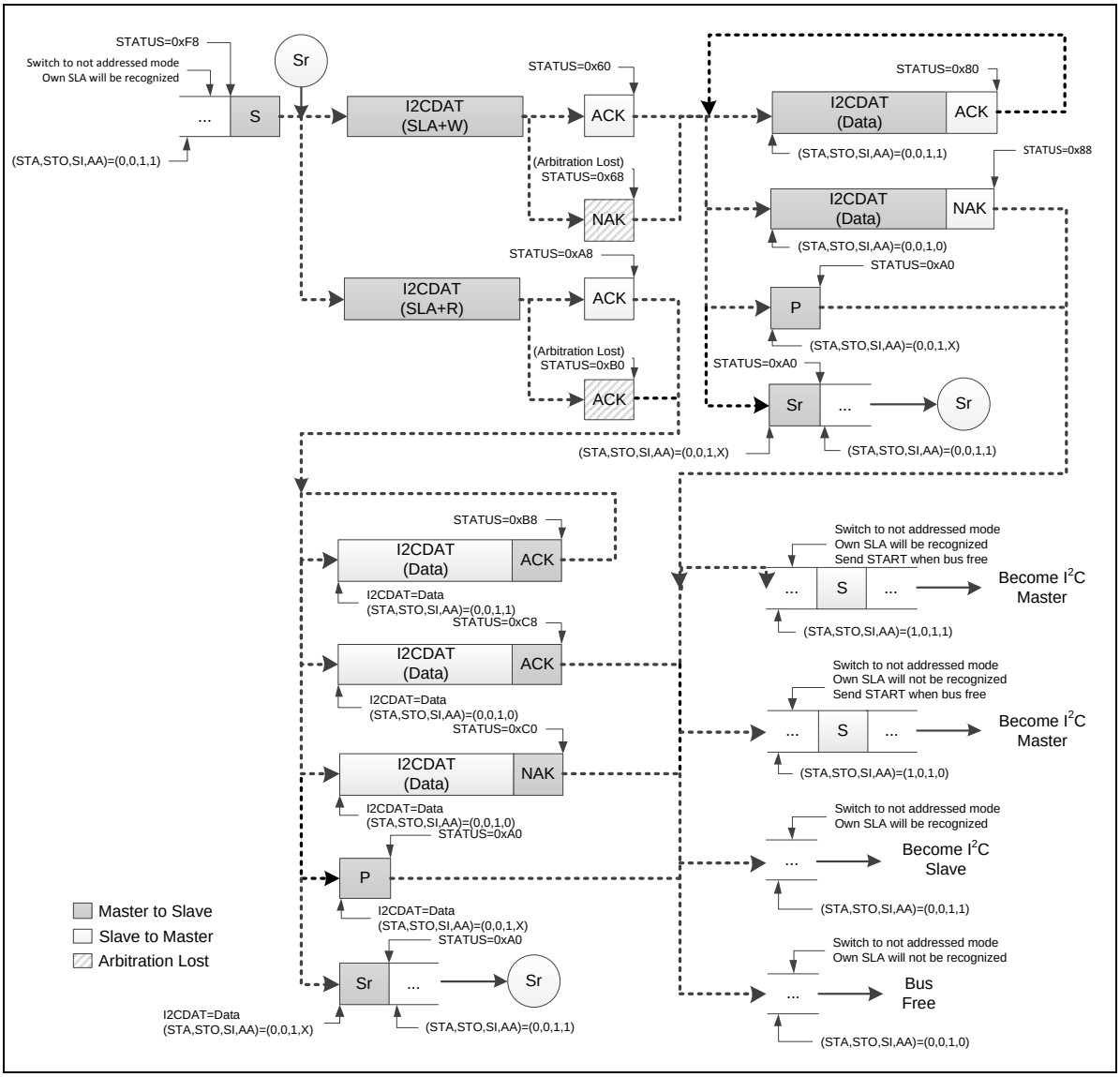


Figure 6-19 Save Mode Control Flow

If I^{2}C is still receiving data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x88 as shown in the above figure when getting 0xA0 status.

If I^{2}C is still transmitting data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0xC8 as shown in the above figure when getting 0xA0 status.

Note: After slave gets status of 0x88, 0xC8, 0xC0 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I²C signal or address from master. At this status, I²C should be reset to leave this status.

6.6.3.2.3 *General Call (GC) Mode*

If the GC(I2C_ADDRn [0]) bit is set, the I²C port hardware will respond to General Call address (0x00). User can clear GC bit to disable general call function. When the GC bit is set and the I²C in Slave mode, it can receive the general call address by 0x00 after master send general call address to I²C bus, then it will follow status of GC mode.

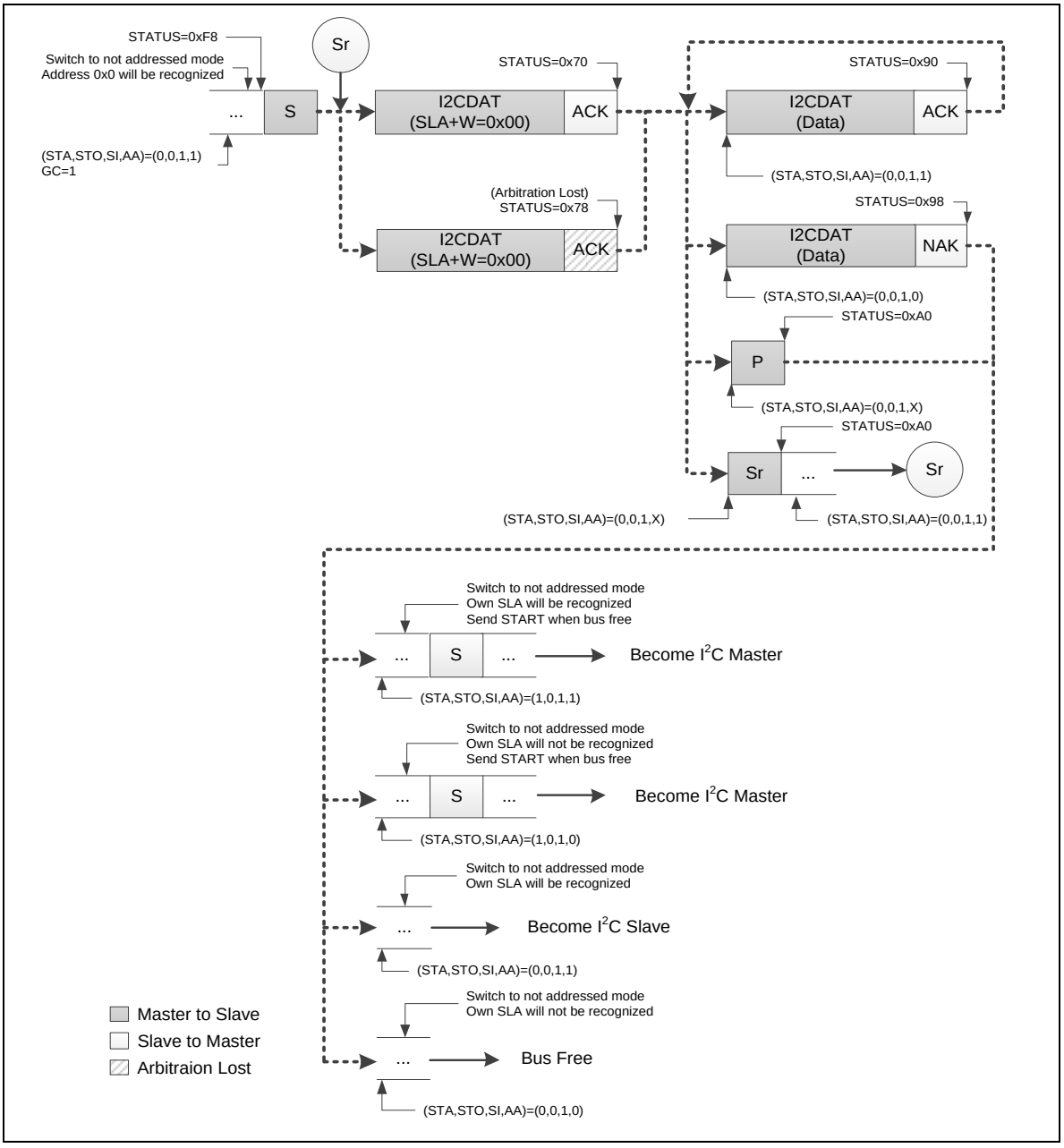


Figure 6-20 GC Mode

If I²C is still receiving data in GC mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x98 in above figure when getting 0xA0 status.

Note: After slave gets status of 0x98 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I²C signal or address from master. At this time, I²C controller should be reset to leave this status.

6.6.3.2.4 *Multi-Master*

In some applications, there are two or more masters on the same I²C bus to access slaves, and the masters may transmit data simultaneously. The I²C supports multi-master by including collision detection and arbitration to prevent data corruption.

If for some reason two masters initiate command at the same time, the arbitration procedure determines which master wins and can continue with the command. Arbitration is performed on the I2Cn_SDA signal while the I2Cn_SCL signal is high. Each master checks if the I2Cn_SDA signal on the bus corresponds to the generated I2Cn_SDA signal. If the I2Cn_SDA signal on the bus is low but it should be high, then this master has lost arbitration. The device that has lost arbitration can generate I2Cn_SCL pulses until the byte ends. The arbitration procedure can continue until all the data is transferred. This means that in multi-master system each master must monitor the bus for collisions and act accordingly.

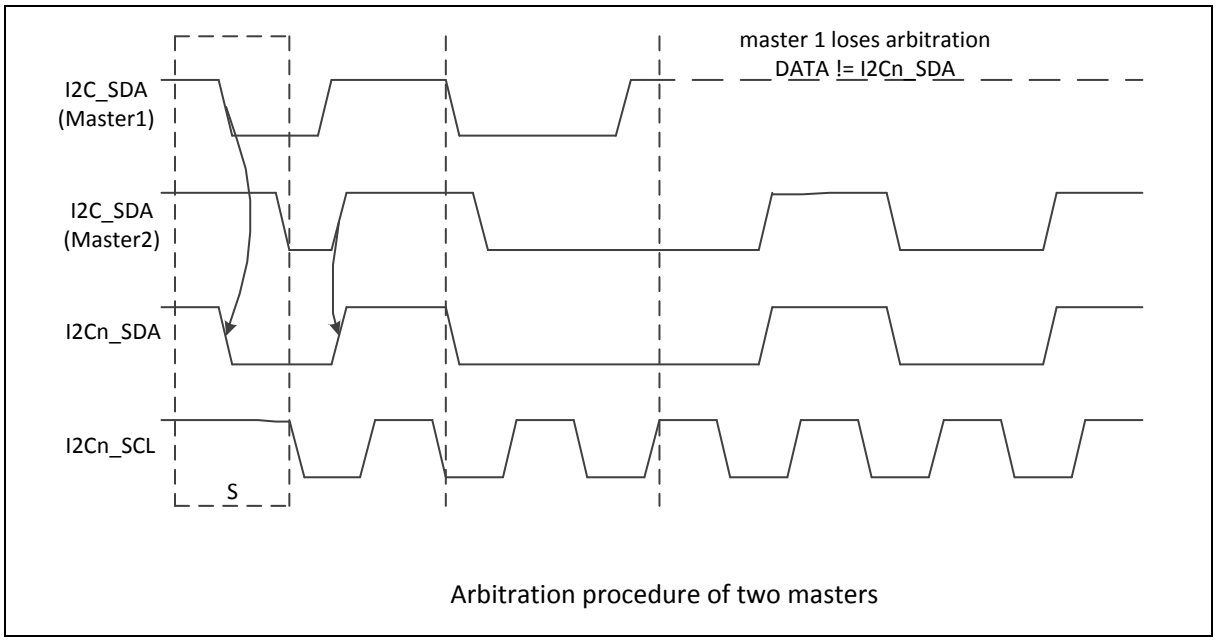


Figure 6-21 Arbitration Lost

- When I2C_STATUS = 0x38, an “Arbitration Lost” is received. Arbitration lost event maybe occur during the send START bit, data bits or STOP bit. User could set (STA, STO, SI, AA) = (1, 0, 1, X) to send START again when bus free, or set (STA, STO, SI, AA) = (0, 0, 1, X) back to not addressed Slave mode.
- When I2C_STATUS = 0x00, a “Bus Error” is received. To recover I²C bus from a bus error, STO(I2C_CTL[4]) should be set and SI(I2C_CTL[3]) should be cleared, and then STO(I2C_CTL[4]) is cleared to release bus.
  - Set (STA, STO, SI, AA) = (0, 1, 1, X) to stop current transfer
  - Set (STA, STO, SI, AA) = (0, 0, 1, X) to release bus

6.6.3.3 I²C Protocol Registers

The CPU interfaces to the SIO port through the following thirteen special function registers: I2C_CTL (control register), I2C_STATUS (status register), I2C_DAT (data register), I2C_ADDRn (address registers, n=0~3), I2C_ADDRMSKn (address mask registers, n=0~3), I2C_CLKDIV (clock rate register) and I2C_TOCTL (Time-out counter register). Bits 31~ bit 8 of these I2C special function registers are reserved. These bits do not have any functions and are all zero if read back.

When I²C port is enabled by setting ENSI (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 theISD9300 is 0x4002_0000.

6.6.3.3.1 *AddressRegisters (I2C_ADDR)*

I²C port is equipped with four slave address registers I2C_ADDRn (n=0~3). The contents of the register are irrelevant when I²C is in master mode. In the slave mode, the bit field I2C_ADDRn[7:1] must be loaded with the MCU's own slave address. The I²C hardware will react if the contents of I2C_ADDR are matched with the received slave address.

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.

6.6.3.3.2 *Data Register (I2C_DAT)*

This register contains a byte of serial data to be transmitted or a byte which just has been received. The CPU can be read from or written to the 8-bit (I2C_DAT [7:0]) directly while it is not in the process of shifting a byte. When I²C is in a defined state and the SI (I2C_CTL[3]) is set, data in I2C_DAT [7:0] remains stable. 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 presented on the bus.

The acknowledge bit is controlled by the I²C hardware and cannot be accessed by the CPU. Serial data is shifted into I2C_DAT [7:0] on the rising edges of serial clock pulses on the I2Cn_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. In order to monitor bus status while sending data, the bus date will be shifted to I2C_DAT[7:0] when sending I2C_DAT[7:0] to bus. In the case of sending data, serial data bits are shifted out from I2C_DAT [7:0] on the falling edge of I2Cn_SCL clocks, and is shifted to I2C_DAT [7:0] on the rising edge of I2Cn_SCL clocks.

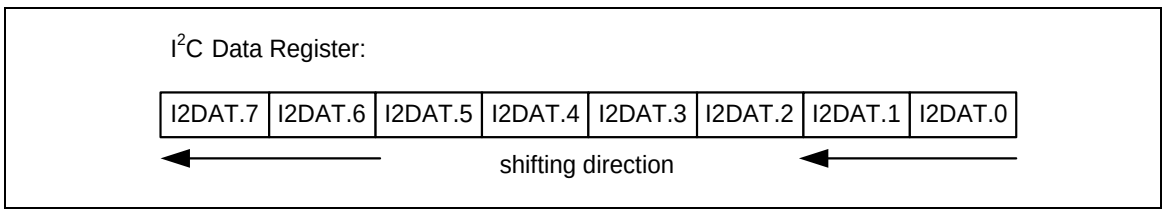


Figure 6-22 I²C Data Shifting Direction

6.6.3.3.3 *Control Register (I2C_CTL)*

The CPU can be read from and written to I2C_CTLregister directly. When the I^2C port is enabled by setting ENS1 (I2C_CTL [6]) to high, the internal states will be controlled by I2C_CTL and I^2C logic hardware.

There are two bits are affected by hardware: the SI(I2C_CTL[3]) bit is set when the I^2C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO(I2C_CTL[4]) bit is also cleared when ENS1(I2C_CTL[6]) = 0.

Once a new status code is generated and stored in I2C_STATUS, the I^2C Interrupt Flag bit SI will be set automatically. If the Enable Interrupt bit EI (I2C_CTL [7]) is set at this time, the I^2C interrupt will be generated. These bit fields I2C_STATUS[7:0] stores the internal state code, the content keeps stable until SI(I2C_CTL [3]) is cleared by software.

6.6.3.3.4 *StatusRegister (I2C_STATUS)*

I2C_STATUS [7:0] is an 8-bit read-only register. The bit fields I2C_STATUS [7:0] contains the status code and there are 26 possible status codes. All states are listed in 0 when I2C_STATUS [7:0] is 0xF8, no serial interrupt is requested. All other I2C_STATUS [7:0] values correspond to the defined I^2C states. When each of these states is entered, a status interrupt is requested (SI (I2C_CTL [3]) = 1). A valid status code is present in I2C_STATUS[7:0] one cycle after SI set by hardware and is still present one cycle after SI reset by software.

In addition, the state 0x00 stands for a Bus Error, which occurs when a START or STOP condition is present at an incorrect position in the I^2C format frame. A Bus Error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I^2C from bus error, STO (I2C_CTL [4]) should be set and SI(I2C_CTL [3]) should be cleared to enter Not Addressed Slave mode. Then STO(I2C_CTL [4]) is cleared to release bus and to wait for a new communication. The I^2C bus cannot recognize stop condition during this action when a bus error occurs.

Master Mode		**Slave Mode**	
STATUS	**Description**	**STATUS**	**Description**
0x08	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10	Master Repeat Start	0xA8	Slave Transmit Address ACK
0x18	Master Transmit Address ACK	0xB0	Slave Transmit Arbitration Lost
0x20	Master Transmit Address NACK	0xB8	Slave Transmit Data ACK
0x28	Master Transmit Data ACK	0xC0	Slave Transmit Data NACK
0x30	Master Transmit Data NACK	0xC8	Slave Transmit Last Data ACK
0x38	Master Arbitration Lost	0x60	Slave Receive Address ACK
0x40	Master Receive Address ACK	0x68	Slave Receive Arbitration Lost
0x48	Master Receive Address NACK	0x80	Slave Receive Data ACK
0x50	Master Receive Data ACK	0x88	Slave Receive Data NACK
0x58	Master Receive Data NACK		
0x00	Bus error		

0xF8	Bus Released **Note:** Status “0xF8” exists in both master/slave modes, and it won’t raise interrupt.

Table 6-5 I^2C Status Code Description

6.6.3.3.5 *Clock Baud Rate Bits (I2C_CLKDIV)*

The data baud rate of I^2C is determines by I2C_CLKDIV(I2C_CLKDIV[7:0]) when I^2C is in Master Mode, and it is not necessary in a Slave mode. In the Slave mode, I^2C will automatically synchronize it with any clock frequency from master I^2C device.

The data baud rate of I^2C setting is Data Baud Rate of I^2C = (system clock) / (4x (I2CLK [7:0] +1)). If system clock = 16 MHz, the I2CLK [7:0] = 40 (0x28), the data baud rate of I^2C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

6.6.3.3.6 *Time-out CounterRegister (I2C_TOCTL)*

There is a 14-bit time-out counter which can be used to deal with the I^2C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TOIF (I2C_TOCTL[0]) =1) and generates I^2C interrupt to CPU or stops counting by clearing TOCEN(I2C_TOCTL[2]) to 0. When time-out counter is enabled, writing 1 to the SI (I2C_CTL[3]) flag will reset counter and re-start up counting after SI is cleared. If I^2C bus hangs up, it causes the I2C_STATUS and flag SI (I2C_CTL[3]) are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the I^2C interrupt. Refer to the following figure for the 14-bit time-out counter. User may write 1 to clear TOIF(I2C_TOCTL[0]) to 0.

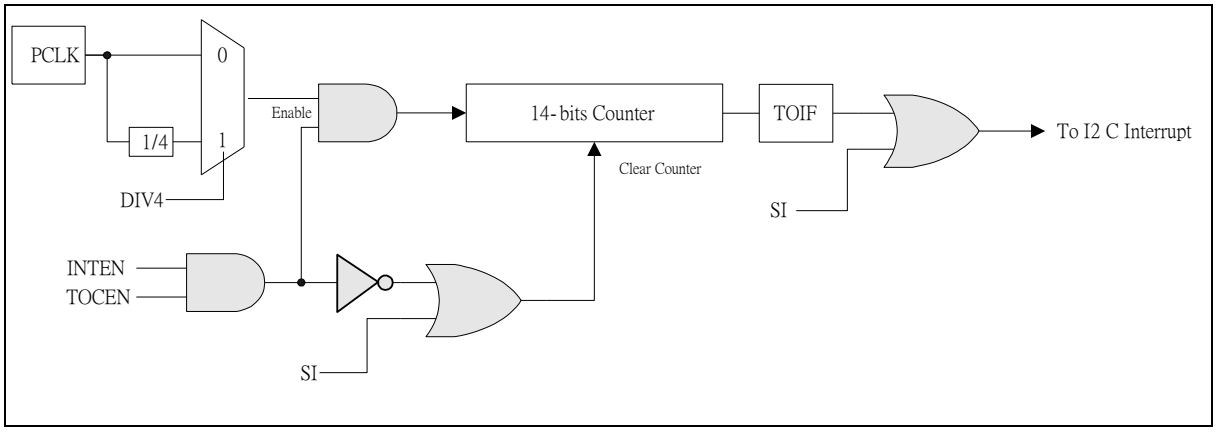


Figure 6-23 I^2C Time-out Count Block Diagram

6.6.4 Register Map

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

Register	Offset	R/W	Description	Reset Value
I²C Base Address: **I2C_BA = 0x4002_0000**				
I2C_CTL	I2C_BA+0x00	R/W	I2C Control Register	0x0000_0000
I2C_ADDR0	I2C_BA+0x04	R/W	I2C Slave address Register0	0x0000_0000
I2C_DAT	I2C_BA+0x08	R/W	I2C DATA Register	0x0000_0000
I2C_STATUS	I2C_BA+0x0C	R	I2C Status Register	0x0000_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

6.6.5 Register Description

I²C Control Register (I2C_CTL)

Register	Offset	R/W	Description	Reset Value
I2C_CTL	I2C_BA+0x00	R/W	I2C Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN	**I2CEN**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**INTEN**	**Enable Interrupt** 0 = Disable interrupt. 1 = Enable interrupt CPU.
[6]	**I2CEN**	**I²CController Enable Bit** 0 = Disable. 1 = Enable. Set to enableI²C serial function block.
[5]	**STA**	**I²C START Control** Setting STA to logic 1 to 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** In Master mode, setting STO to transmit a STOP condition to bus then I²C hardware will check the bus condition if a STOP condition is detected this bit will be cleared by hardware automatically. 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.
[3]	**SI**	**I²C Interrupt Flag** When a new I²C 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 by writing 1 to this bit.
[2]	**AA**	**Assert Acknowledge Control** When AA =1 prior to address or data received, an acknowledged (low level to I2Cn_SDA) will be returned during the acknowledge clock pulse on the I2Cn_SCL line when 1.) A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to I2Cn_SDA) will be returned during the acknowledge clock pulse on the I2Cn_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

31	30	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[7:0]							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DAT**	**I²C Data Register** During master or slave transmit mode, data to be transmitted is written to this register. During master or slave receive mode, data that has been received may be read from this register.

I²C Status Register (I2C_STATUS)

Register	Offset	R/W	Description	Reset Value
I2C_STATUS	I2C_BA+0x0C	R	I2C Status Register	0x0000_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**
STATUS[7:0]							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**STATUS**	**I²C Status Register** The status register of I2C: The three least significant bits are always 0. The five most significant bits contain the status code. There are 26 possible status codes. When I2C_STATUS contains F8H, no serial interrupt is requested. All other I2C_STATUS values correspond to defined I2C states. When each of these states is entered, a status interrupt is requested (SI= 1). A valid status code is present in I2C_STATUS one PCLK cycle after SI is set by hardware and is still present one PCLK cycle after SI has been reset by software. In addition, states 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit.

I²C Clock Divided Register (I2C_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2C_CLKDIV	I2C_BA+0x10	R/W	I2C clock divided Register	0x0000_0000

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

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

I²C Time-out Counter Register (I2C_TOCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TOCTL	I2C_BA+0x14	R/W	I2C Time out control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TOCEN	TOCDIV4	TOIF

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**TOCEN**	**Time-Out Counter Control Bit** 0 = Disable. 1 = Enable. When enabled, the 14 bit time-out counter will start counting when SI is clear. Setting flag SI to high will reset counter and re-start up counting after SI is cleared.
[1]	**TOCDIV4**	**Time-Out Counter Input Clock Divide By 4** 0 = Disable. 1 = Enable. When enabled, the time-out clock is PCLK/4.
[0]	**TOIF**	**Time-Out Flag** 0 = No time-out. 1 = Time-out flag is set by H/W. It can interrupt CPU. Write 1 to clear..

I²C Slave Address Register (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

31	30	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**
ADDR[7:1]							**GC**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**ADDR**	**I²C Address Register** The content of this register is irrelevant when I²C is in Master mode. In the slave mode, the seven most significant bits must be loaded with the chip's own address. The I²C hardware will react if either of the address is matched.
[0]	**GC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**ADDRMSK**	**I²C Address Mask Register** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). I²C bus controllers support multiple address recognition with four address mask register. 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.
[0]	**Reserved**	Reserved.

6.7 PWM Generator and Capture Timer (PWM)

6.7.1 Overview

The ISD9300 series has three PWM generators which can be configured as 6 independent PWM outputs,PWM0CH0, PWM0CH1, PWM0CH2,PWM0CH3,PWM1CH0 and PWM1CH1, or as a complementary PWM pairs with programmable dead-zone generator. Each PWM Generator has an 8-bit pre-scaler, a clock divider providing 5 divided frequencies (1, 1/2, 1/4, 1/8, 1/16), two PWM Timers including two clock selectors, two 16-bit PWM down-counters for PWM period control, two 16-bit comparators for PWM duty control and one dead-zone generator.

The PWM Generator provides PWM interrupt flags which are set by hardware when the corresponding PWM period down counter reaches zero. Each PWM interrupt source, with its corresponding enable bit, can generate a PWM interrupt request to the CPU. The PWM generator can be configured in one-shot mode to produce only one PWM cycle signal or continuous mode to output a periodic PWM waveform.

When PWM_CTL.DTEN01 is set, PWM0CH0 and PWM0CH1 perform complementary paired PWM function; the paired PWM timing, period, duty and dead-time are determined by PWM0 timer and Dead-zone generator0.

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 PWMtimer is configured in continuous mode, when the down counter reaches zero, it is reloaded with PWM Counter Register (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 thePWM-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 whichis 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 (RCAPDAT) when input channel has a rising transition and latched to Capture Falling Latch Register(PWM0_FCAPDAT) when input channel has a falling transition. Capture channel 0 interrupt is programmable by setting PWM_CAPCTL01.CRLIEN0[1] (Rising latch Interrupt enable) and PWM_CAPCTL01.CFLIEN0[2]] (Falling latch Interrupt enable) to determine the condition of interrupt occurrence. Capture channel 1 has the same feature by setting PWM_CAPCTL01.CRLIEN1[17] and PWM_CAPCTL01.CFLIEN1[18]. Whenever Capture issues interrupt, the PWM counter will also be reloaded.

6.7.2 Features

6.7.2.1 PWM Function:

- 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 6 PWM channels or three paired PWM channel.
- Up to 16-bit resolution
- PWM Interrupt request synchronized with PWM period
- Single-shot or Continuous mode PWM.
- Dead-Zone generator.

6.7.2.2 Capture Function:

- Timing control logic shared with PWM Generators
- 6 Capture input channels shared with 6 PWM output channels.
- Each channel supports a rising latch register (RCAPDAT), a falling latch register (PWM0_FCAPDAT) and Capture interrupt flag (CAPIFx)

6.7.3 Block Diagram

Figure 6-24 illustrate the architecture of PWM0CH01 in pair.

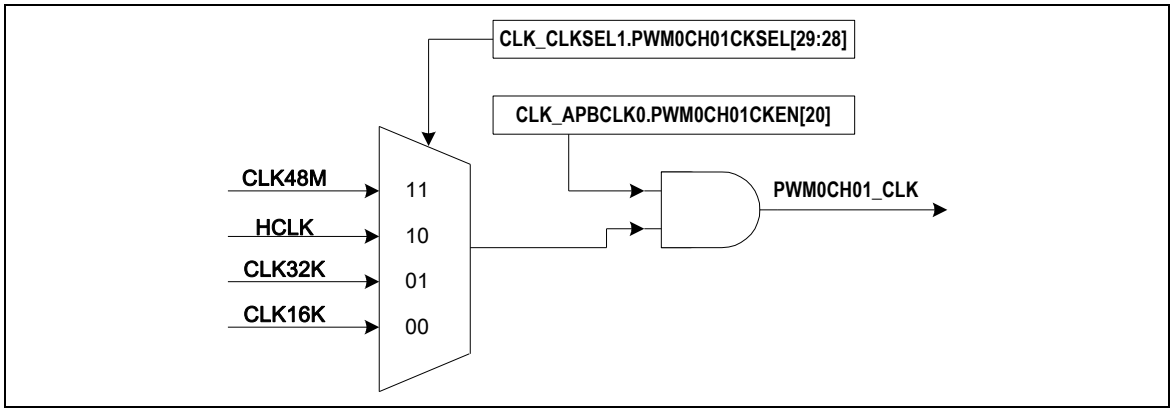


Figure 6-24 PWM Generator Clock Source Control

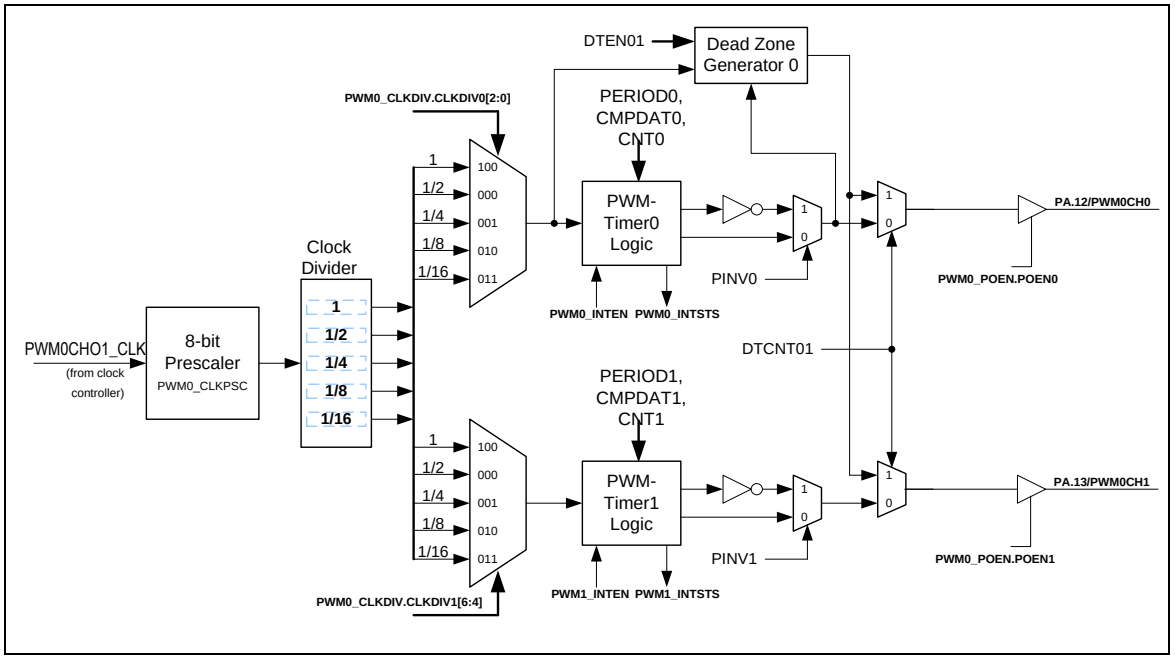


Figure 6-25 PWM Generator Architecture Diagram

6.7.4 Functional Description

6.7.4.1 PWM-Timer Operation

The PWM controller supports Edge-aligned operation type.

6.7.4.2 Edge-aligned PWM (down-counter)

In Edge-aligned PWM Output mode, the 16 bits PWM counter will starts down-counting from PWMx_PERIODx to match with the value of the duty cycle PWMx_CMPDATx (old), when this happen it will toggle the PWMn generator output to low. The counter will continue down-counting to 0, at this moment, it toggles the PWMn generator output to high and PWMx_CMPDATx(new) and PWMx_PERIODx(new) are updated.

The PWM period and duty control are configured by PWM down-counter register (PWMx_PERIODx) and PWM comparator register (PWMx_CMPDATx). The pulse width modulation follows the formula below. Note that the corresponding GPIO pins must be configured as PWM function (enable POE and disable CAPENR) for the corresponding PWM channel.

- PWM frequency = PWMxy_CLK/[(prescale+1)*(clock divider)*(PERIOD+1)]; where xy, could be 01, 23, 45, depends on selected PWM channel.
- Duty ratio = (CMPDAT+1)/(PERIOD+1)
- CMPDAT>= PERIOD: PWM output is always high
- CMPDAT<PERIOD: PWM low width= (PERIOD-CMPDAT) unit[1]; PWM high width = (CMPDAT+1) unit
- CMPDAT = 0: PWM low width = (PERIOD) unit; PWM high width = 1 unit

Note [1]: unit = one PWM clock cycle.

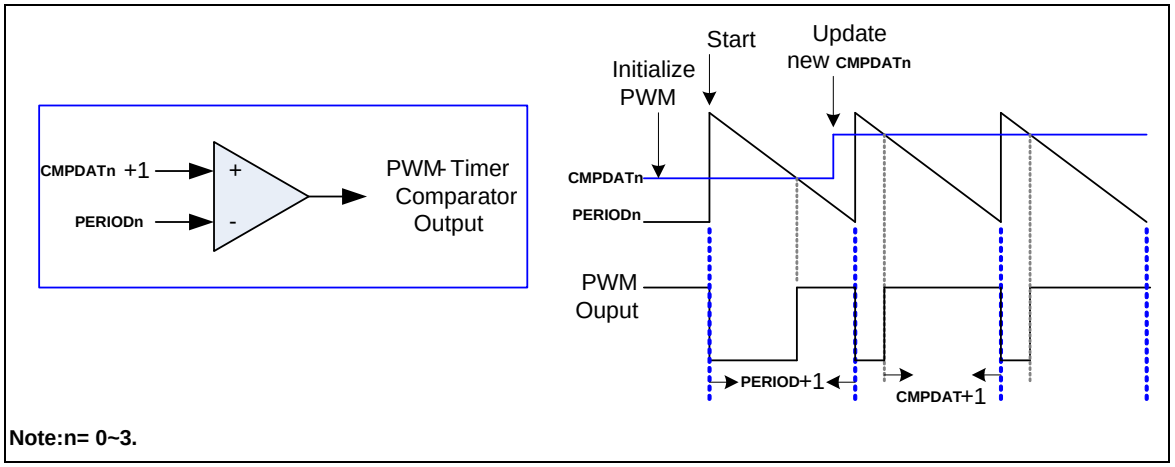


Figure 6-26 Legend of Internal Comparator Output of PWM-Timer

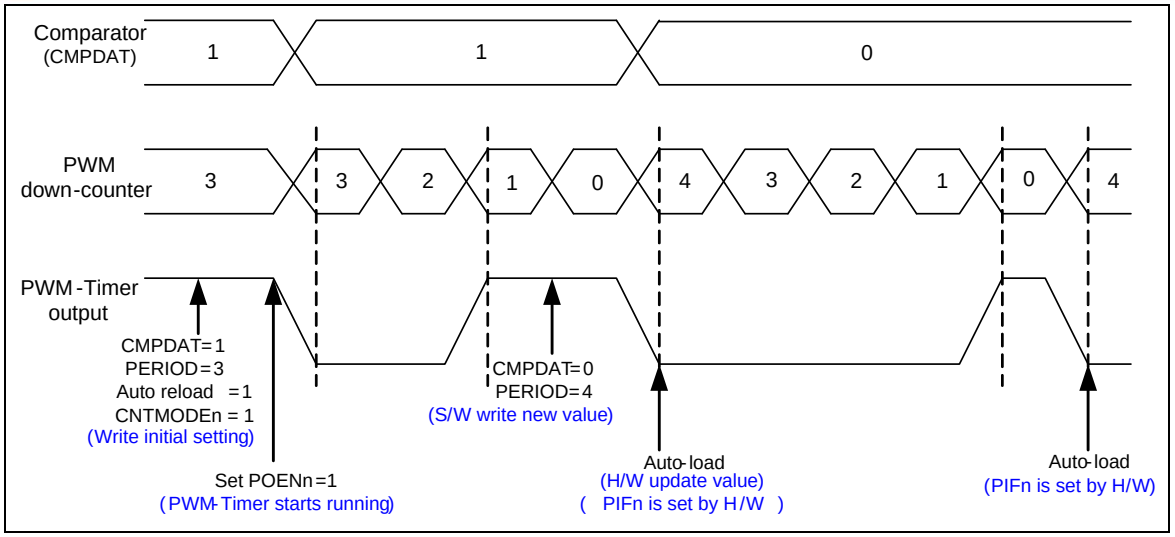


Figure 6-27 PWM-Timer Operation Timing

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

The ISD9300 series PWM Timers are double buffered, the reload value is updated at the start of next period without affecting current timer operation. The PWM counter reset value can be written into PWM_PERIOD0~1 and current PWM counter value can be read from PWM_CNT0~1.

The bit CNTMODEx in PWM Control Register (PWM_CTL) determines whether PWMx operates in auto-reload or one-shot mode. If CNTMODEx is set to one, the auto-reload operation loadsPERIODx 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.

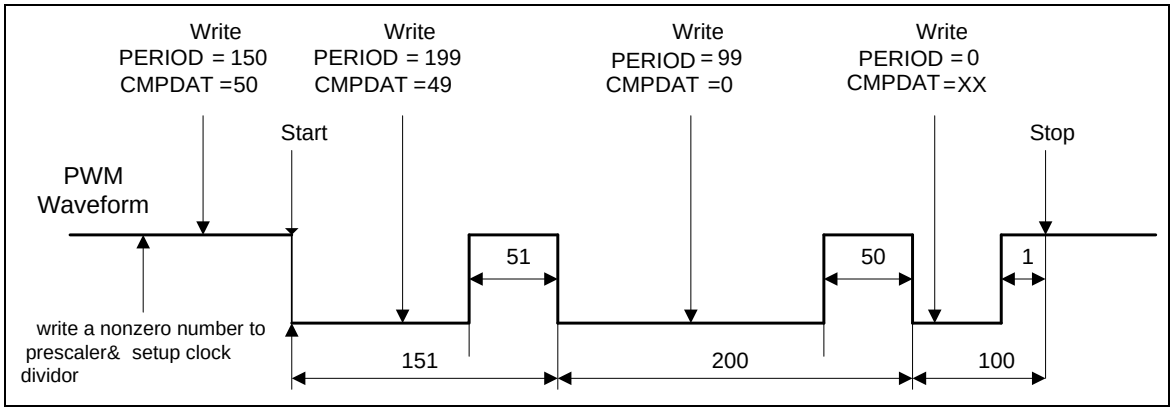


Figure 6-28 PWM Double Buffering Illustration

6.7.4.4 Modulate Duty Ratio

The double buffering allows CMPDAT to be written at any point in current cycle. The loaded value will take effect from next cycle.

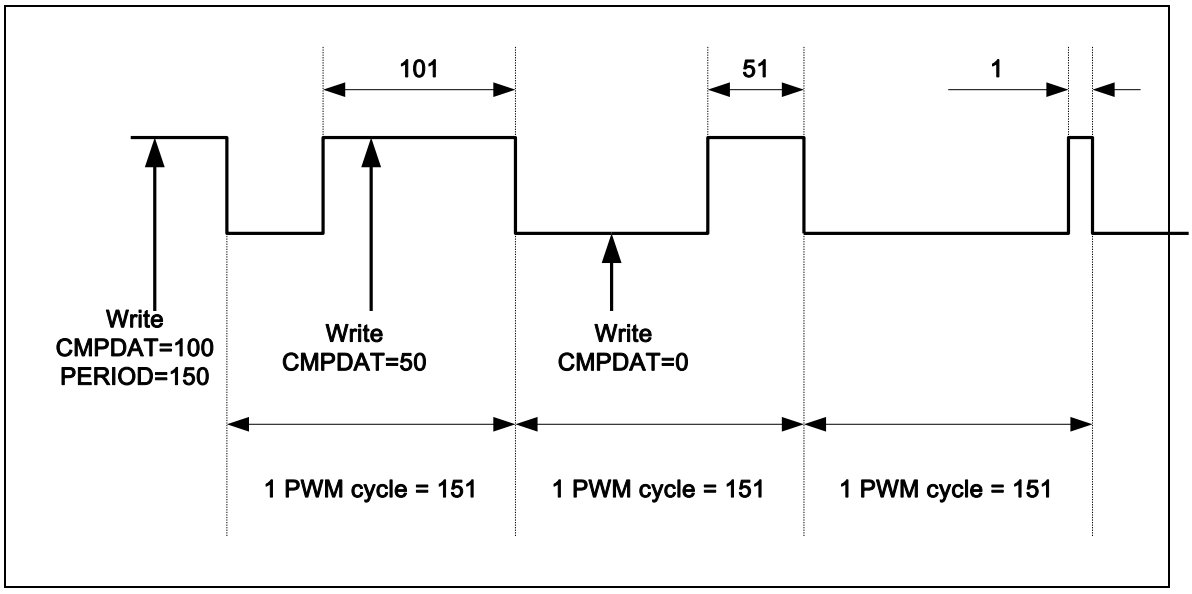


Figure 6-29 PWM Controller Output Duty Ratio

6.7.4.5 Dead-Zone Generator

The ISD9300 PWM generator includes a Dead Zone generator. This is used to ensure neither PWM output is active simultaneously for power device protection. The function generates a programmable time gap between rising PWM outputs. The user can program PPRx.DZI to determine the Dead Zone interval.

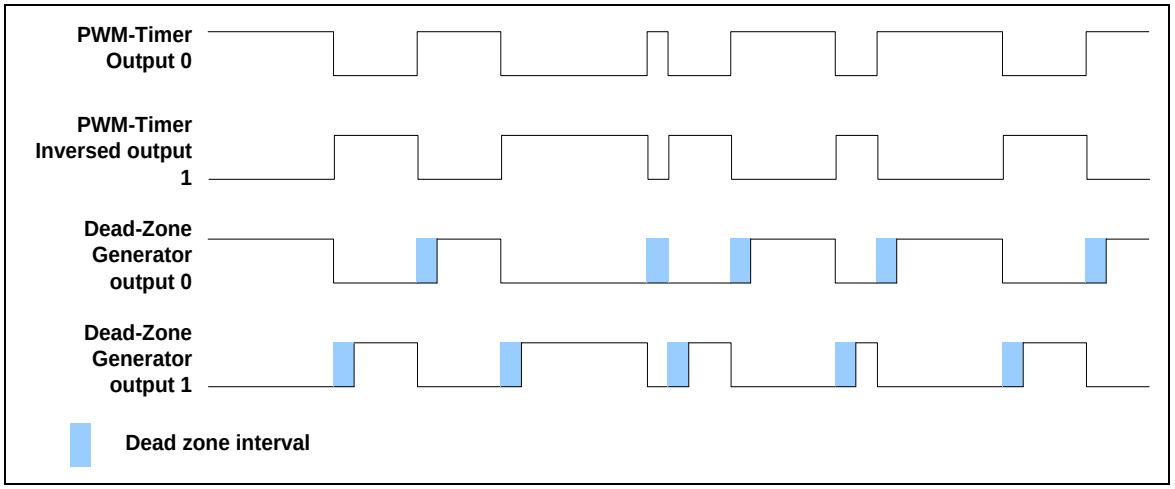


Figure 6-30 Paired-PWM Output with Dead-zone Generation Operation

6.7.4.6 Capture 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 PWM0_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.

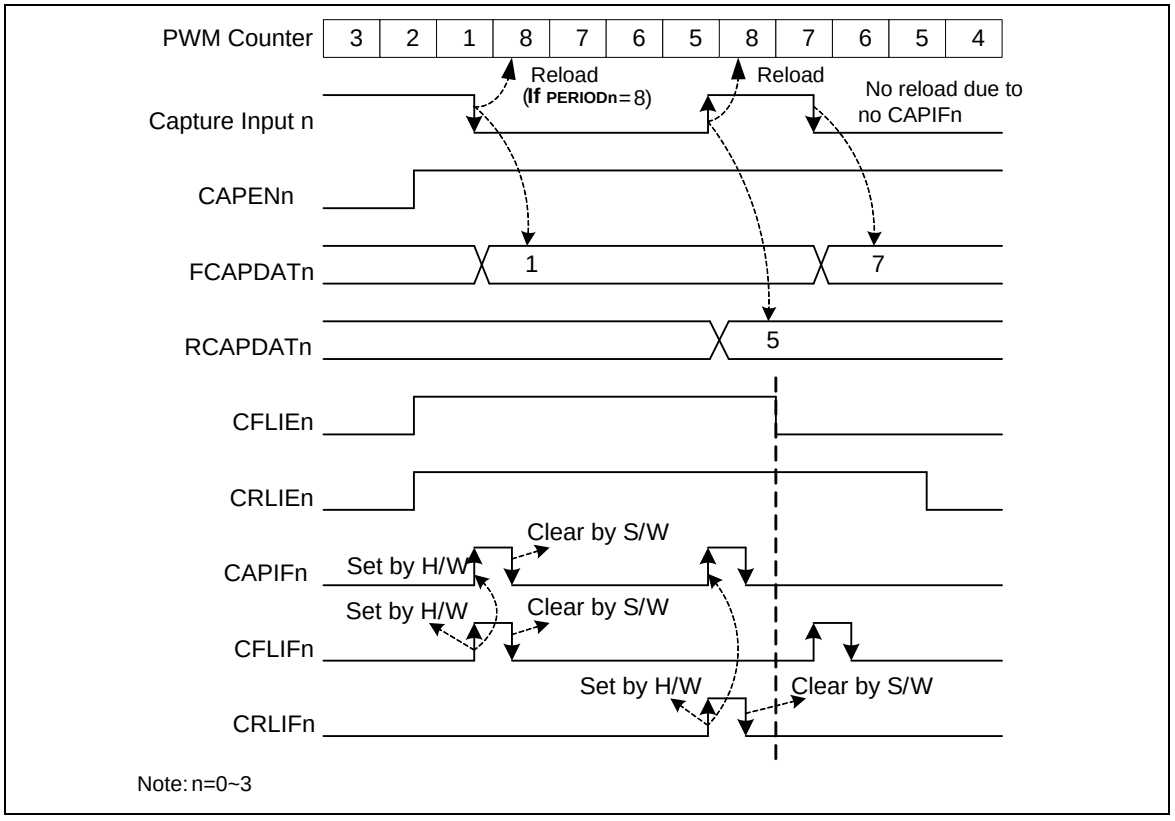


Figure 6-31 Capture Operation Timing

In this case, the PERIOD is 8:

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

6.7.4.7 PWM-Timer Interrupt Architecture

There are fourPWM interrupts, PWM0_INTEN.PIENn (n=0~3), which are multiplexed into PWM0_IRQ. PWM 0 and Capture 0 share one interrupt, PWM1 and Capture 1 share the same interrupt. Figure 6-32 demonstrates the architecture of PWM-Timer interrupts.

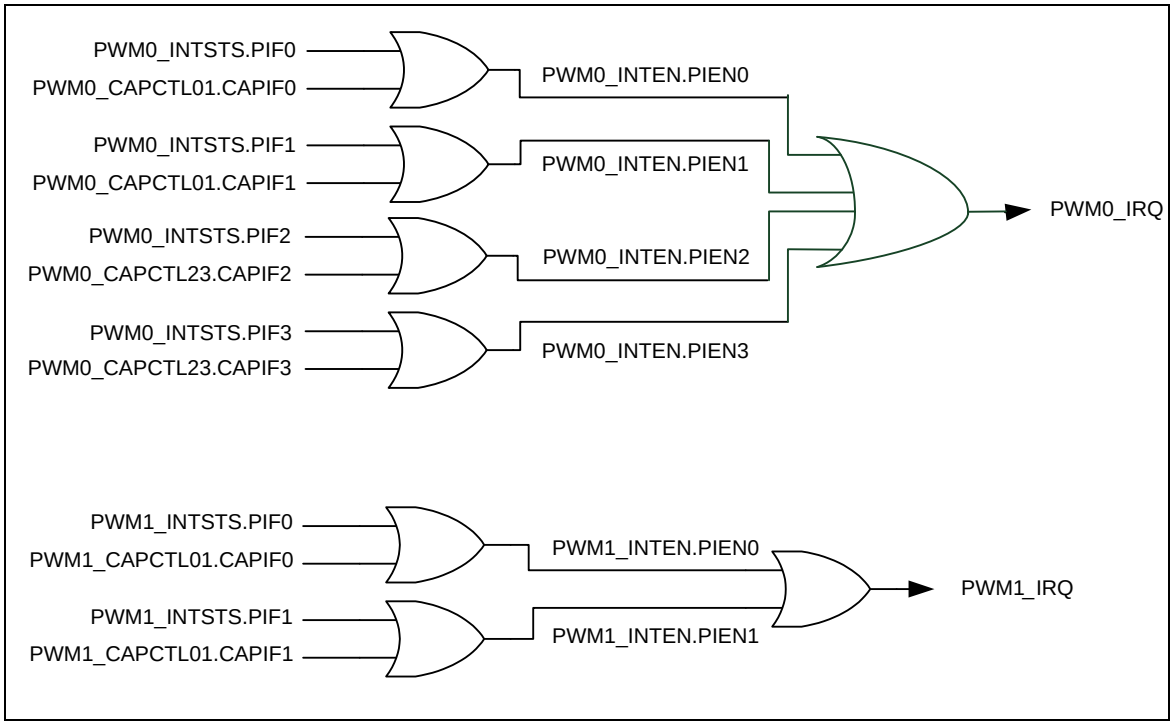


Figure 6-32 PWM-Timer Interrupt Architecture Diagram

6.7.4.8 PWM-Timer Start Procedure

The following procedure is recommended for starting a PWM drive.

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

6.7.4.9 PWM-Timer Stop Procedure

Method 1:

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

Method 2:

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

Method 3:

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

The reason why method 3 is not recommended is that disable CNTENx will immediately stop PWM output signal and lead to change the duty of the PWM output, this may cause damage to the control circuit of motor

6.7.4.10 Capture Start Procedure

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

6.7.5 PWM0 Register Map

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

Register	Offset	R/W	Description	Reset Value
PWM0 Base Address: **PWM_BA = 0x4004_0000**				
PWM0_CLKPSC	PWM_BA+0x000	R/W	PWM Prescaler Register	0x0000_0000
PWM0_CLKDIV	PWM_BA+0x004	R/W	PWM Clock Select Register	0x0000_0000
PWM0_CTL	PWM_BA+0x008	R/W	PWM Control Register	0x0000_0000
PWM0_PERIOD0	PWM_BA+0x00C	R/W	PWM Counter Register 0	0x0000_0000
PWM0_CMPDAT0	PWM_BA+0x010	R/W	PWM Comparator Register 0	0x0000_0000
PWM0_CNT0	PWM_BA+0x014	R	PWM Data Register 0	0x0000_0000
PWM0_PERIOD1	PWM_BA+0x018	R/W	PWM Counter Register 1	0x0000_0000
PWM0_CMPDAT1	PWM_BA+0x01C	R/W	PWM Comparator Register 1	0x0000_0000
PWM0_CNT1	PWM_BA+0x020	R	PWM Data Register 1	0x0000_0000
PWM0_PERIOD2	PWM_BA+0x024	R/W	PWM Counter Register 2	0x0000_0000
PWM0_CMPDAT2	PWM_BA+0x028	R/W	PWM Comparator Register 2	0x0000_0000
PWM0_CNT2	PWM_BA+0x02C	R	PWM Data Register 2	0x0000_0000
PWM0_PERIOD3	PWM_BA+0x030	R/W	PWM Counter Register 3	0x0000_0000
PWM0_CMPDAT3	PWM_BA+0x034	R/W	PWM Comparator Register 3	0x0000_0000
PWM0_CNT3	PWM_BA+0x038	R	PWM Data Register 3	0x0000_0000
PWM0_INTEN	PWM_BA+0x040	R/W	PWM Interrupt Enable Register	0x0000_0000
PWM0_INTSTS	PWM_BA+0x044	R/W	PWM Interrupt Flag Register	0x0000_0000
PWM0_CAPCTL01	PWM_BA+0x050	R/W	Capture Control RegisterFor Pair Of PWM0CH0And PWM0CH1	0x0000_0000
PWM0_CAPCTL23	PWM_BA+0x054	R/W	Capture Control RegisterFor Pair Of PWM0CH2And PWM0CH3	0x0000_0000
PWM0_RCAPDAT0	PWM_BA+0x058	R	Capture Rising Latch Register (Channel 0)	0x0000_0000
PWM0_FCAPDAT0	PWM_BA+0x05C	R	Capture Falling Latch Register (Channel 0)	0x0000_0000
PWM0_RCAPDAT1	PWM_BA+0x060	R	Capture Rising Latch Register (Channel 1)	0x0000_0000
PWM0_FCAPDAT1	PWM_BA+0x064	R	Capture Falling Latch Register (Channel 1)	0x0000_0000
PWM0_RCAPDAT2	PWM_BA+0x068	R	Capture Rising Latch Register (Channel 2)	0x0000_0000
PWM0_FCAPDAT2	PWM_BA+0x06C	R	Capture Falling Latch Register (Channel 2)	0x0000_0000
PWM0_RCAPDAT3	PWM_BA+0x070	R	Capture Rising Latch Register (Channel 3)	0x0000_0000

PWM0_FCAPDAT3	PWM_BA+0x074	R	Capture Falling Latch Register (Channel 3)	0x0000_0000
PWM0_CAPINEN	PWM_BA+0x078	R/W	Capture Input Enable Register	0x0000_0000
PWM0_POEN	PWM_BA+0x07C	R/W	PWM0 Output Enable Register for CH0~CH3	0x0000_0000

6.7.6 Register Description

PWM Pre-Scale Register (PWM0_CLKPSC)

Register	Offset	R/W	Description	Reset Value
PWM0_CLKPSC	PWM_BA+0x000	R/W	PWM Prescaler Register	0x0000_0000

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

Bits	Description	
[31:24]	**DTCNT23**	**Dead Zone Interval Register For Pair Of PWM0CH2And PWM0CH3** These 8 bits determine dead zone length. The unit time of dead zone length is that from clock selector 0.
[23:16]	**DTCNT01**	**Dead Zone Interval Register For Pair Of PWM0CH0And PWM0CH1** These 8 bits determine dead zone length. The unit time of dead zone length is that from clock selector 0.
[15:8]	**CLKPSC23**	**Clock Pre-Scaler For Pair Of PWM0CH2And PWM0CH3** Clock input is divided by (CLKPSC23 + 1) If CLKPSC23=0, then the pre-scaler output clock will be stopped. This implies PWM counter 2 and 3 will also be stopped.
[7:0]	**CLKPSC01**	**Clock Pre-ScalerPair Of PWM0CH0And PWM0CH1** Clock input is divided by (CLKPSC01 + 1) If CLKPSC01=0, then the pre-scaler output clock will be stopped. This implies PWM counter 0 and 1 will also be stopped.

PWM Clock Select Register (PWM0_CLKDIV)

Register	Offset	R/W	Description	Reset Value
PWM0_CLKDIV	PWM_BA+0x004	R/W	PWM Clock Select Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**CLKDIV3**			**Reserved**	**CLKDIV2**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CLKDIV1**			**Reserved**	**CLKDIV0**		

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**CLKDIV3**	**Timer 3 Clock Source Selection** Select clock source divider for PWM timer 3. (Table is as CLKDIV0)
[11]	**Reserved**	Reserved.
[10:8]	**CLKDIV2**	**Timer 2 Clock Source Selection** Select clock source divider for PWM timer 2. (Table is as CLKDIV0)
[7]	**Reserved**	Reserved.
[6:4]	**CLKDIV1**	**Timer 1 Clock Source Selection** Select clock source divider for PWM timer 1. (Table is as CLKDIV0)
[3]	**Reserved**	Reserved.
[2:0]	**CLKDIV0**	**Timer 0 Clock Source Selection** Select clock source divider for PWM timer 0. 000 = 2. 001 = 4. 010 = 8. 011 = 16. 100 = 1.

PWM Control Register (PWM0_CTL)

Register	Offset	R/W	Description	Reset Value
PWM0_CTL	PWM_BA+0x008	R/W	PWM Control Register	0x0000_0000

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

Bits	Description	
[27]	**CNTMODE3**	**PWM-Timer 3 Auto-Reload/One-Shot Mode** 0 = One-Shot Mode. 1 = Auto-load Mode. Note: A rising transition of this bit will cause PWM_PERIOD3 and PWM_CMPDAT3 to be cleared.
[26]	**PINV3**	**PWM-Timer 3 Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.
[24]	**CNTEN3**	**PWM-Timer 3 Enable/Disable Start Run** 0 = Stop PWM-Timer 3. 1 = Enable PWM-Timer 3 Start/Run.
[19]	**CNTMODE2**	**PWM-Timer 2 Auto-Reload/One-Shot Mode** 0 = One-Shot Mode. 1 = Auto-load Mode. Note: A rising transition of this bit will cause PWM_PERIOD2 and PWM_CMPDAT2 to be cleared.
[18]	**PINV2**	**PWM-Timer 2 Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.
[16]	**CNTEN2**	**PWM-Timer 2 Enable/Disable Start Run** 0 = Stop PWM-Timer 2. 1 = Enable PWM-Timer 2 Start/Run.
[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.
[5]	**DTEN23**	**Dead-Zone 23 Generator Enable/DisablePair Of PWM0CH2And PWM0CH3** 0 = Disable. 1 = Enable. Note: When Dead-Zone Generator is enabled, the pair of PWM0CH2 and PWM0CH3become a complementary pair.
[4]	**DTEN01**	**Dead-Zone 01 Generator Enable/DisablePair Of PWM0CH0And PWM0CH1** 0 = Disable. 1 = Enable. Note: When Dead-Zone Generator is enabled, the pair of PWM0CH0 and PWM0CH1 become a complementary pair.
[3]	**CNTMODE0**	**PWM-Timer 0 Auto-Reload/One-Shot Mode** 0 = One-Shot Mode. 1 = Auto-reload Mode. Note: A rising transition of this bit will cause PWM_PERIOD0 and PWM_CMPDAT0 to be cleared.
[2]	**PINV0**	**PWM-Timer 0 Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.
[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 (PWM0_PERIOD1-0)

Register	Offset	R/W	Description	Reset Value
PWM0_PERIOD0	PWM_BA+0x00C	R/W	PWM Counter Register 0	0x0000_0000
PWM0_PERIOD1	PWM_BA+0x018	R/W	PWM Counter Register 1	0x0000_0000
PWM0_PERIOD2	PWM_BA+0x024	R/W	PWM Counter Register 2	0x0000_0000
PWM0_PERIOD3	PWM_BA+0x030	R/W	PWM Counter Register 3	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**PWM Timer Loaded Value** PERIODdetermines the PWM period. PWM frequency = PWM0CHx_CLK/[(prescale+1)*(clock divider)*(PERIOD+1)]; where x, depends on selected PWM channel. For Edge-aligned type: ● Duty ratio = (CMPDAT+1)/(PERIOD+1). ● CMPDAT>= PERIOD: PWM output is always high. ● CMPDAT<PERIOD: PWM low width = (PERIOD- CMPDAT) unit; PWM high width = (CMPDAT+1) unit. ● CMPDAT= 0: PWM low width = (PERIOD) unit; PWM high width = 1 unit. **Note:** Any write to PERIODwill take effect in next PWM cycle. **Note:**When PERIODvalue is set to 0, PWM output is always high.

PWM Comparator Register 1-0 (PWM0_CMPDAT3-0)

Register	Offset	R/W	Description	Reset Value
PWM0_CMPDAT0	PWM_BA+0x010	R/W	PWM Comparator Register 0	0x0000_0000
PWM0_CMPDAT1	PWM_BA+0x01C	R/W	PWM Comparator Register 1	0x0000_0000
PWM0_CMPDAT2	PWM_BA+0x028	R/W	PWM Comparator Register 2	0x0000_0000
PWM0_CMPDAT3	PWM_BA+0x034	R/W	PWM Comparator Register 3	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPDATx**	**PWM Comparator Register** CMPDATXdetermines the PWM duty. PWM frequency = PWM0CHx_CLK/[(prescale+1)*(clock divider)*(PERIOD+1)]; where x, depends on selected PWM channel. For Edge-aligned type: ● Duty ratio = (CMPDATX+1)/(PERIOD+1). ● CMPDATX>= PERIOD: PWM output is always high. ● CMPDATX<PERIOD: PWM low width = (PERIOD-CMPDATX) unit; PWM high width = (CMPDATX+1) unit. ● CMPDATX= 0: PWM low width = (PERIOD) unit; PWM high width = 1 unit. **Note:** Any write to PERIODwill take effect in next PWM cycle.

PWM Data Register 1-0 (PWM0_CNT3~PWM0_CNT0)

Register	Offset	R/W	Description	Reset Value
PWM0_CNT0	PWM_BA+0x014	R	PWM Data Register 0	0x0000_0000
PWM0_CNT1	PWM_BA+0x020	R	PWM Data Register 1	0x0000_0000
PWM0_CNT2	PWM_BA+0x02C	R	PWM Data Register 2	0x0000_0000
PWM0_CNT3	PWM_BA+0x038	R	PWM Data Register 3	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNT**	**PWM Data Register** User can monitor PDR to know the current value in 16-bit counter.

PWM Interrupt Enable Register (PWM0_INTEN)

Register	Offset	R/W	Description	Reset Value
PWM0_INTEN	PWM_BA+0x040	R/W	PWM Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PIEN3**	**PIEN2**	**PIEN1**	**PIEN0**

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

PWM Interrupt Flag Register (PWM0_INTSTS)

Register	Offset	R/W	Description	Reset Value
PWM0_INTSTS	PWM_BA+0x044	R/W	PWM Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PIF3**	**PIF2**	**PIF1**	**PIF0**

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

Note: User can clear each interrupt flag by writing 1 to corresponding bit in PWM_INTSTS.

Capture Control Register (PWM0_CAPCTL01)

Register	Offset	R/W	Description	Reset Value
PWM0_CAPCTL01	PWM_BA+0x050	R/W	Capture Control RegisterFor Pair Of PWM0CH0And PWM0CH1	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CFLIF1**	**PWM_FCAPDAT1 Latched Indicator Bit** When input channel 1 has a **falling** transition, PWM_FCAPDAT1 was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[22]	**CRLIF1**	**PWM_RCAPDAT1 Latched Indicator Bit** When input channel 1 has a **rising** transition, PWM_RCAPDAT1 was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[20]	**CAPIF1**	**Capture1 Interrupt Indication Flag** If channel 1 rising latch interrupt is enabled (CRLIEN1=1), a rising transition at input channel 1 will result in CAPIF1 to high; Similarly, a falling transition will cause CAPIF1 to be set high if channel 1 falling latch interrupt is enabled (CFLIEN1= 1). This flag is cleared by software writing a '1' to it.
[19]	**CAPEN1**	**Capture Channel 1 Transition Enable/Disable** 0 = Disable capture function on channel 1. 1 = Enable capture function on channel 1. When enabled, Capture function latches the PMW-counter to RCAPDAT (Rising latch) and FCAPDAT (Falling latch) registers on input edge transition. When disabled, Capture function is inactive as is interrupt.
[18]	**CFLIEN1**	**Channel 1 Falling Latch Interrupt Enable** 0 = Disable falling edge latch interrupt. 1 = Enable falling edge latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.
[17]	**CRLIEN1**	**Channel 1 Rising Latch Interrupt Enable** 0 = Disable rising edge latch interrupt. 1 = Enable rising edge latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.

[16]	**CAPINV1**	**Channel 1 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer
[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 zeroto 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 zeroto it.
[4]	**CAPIF0**	**Capture0 Interrupt Indication Flag** If channel 0 rising latch interrupt is enabled (CRLIEN0=1), a rising transition at input channel 0 will result in CAPIF0 to high; Similarly, a falling transition will cause CAPIF0 to be set high if channel 0 falling latch interrupt is enabled (CFLIEN0= 1). This flag is cleared by software writing a '1' to it.
[3]	**CAPEN0**	**Capture Channel 0 Transition Enable/Disable** 0 = Disable capture function on channel 0. 1 = Enable capture function on channel 0. When enabled, Capture function latches the PMW-counter to RCAPDAT (Rising latch) and FCAPDAT (Falling latch) registers on input edge transition. When disabled, Capture function is inactive as is interrupt.
[2]	**CFLIEN0**	**Channel 0 Falling Latch Interrupt Enable ON/OFF** 0 = Disable falling latch interrupt. 1 = Enable falling latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.
[1]	**CRLIEN0**	**Channel 0 Rising Latch Interrupt Enable ON/OFF** 0 = Disable rising latch interrupt. 1 = Enable rising latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.
[0]	**CAPINV0**	**Channel 0 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer

Capture Control Register (PWM0_CAPCTL23)

Register	Offset	R/W	Description	Reset Value
PWM0_CAPCTL23	PWM_BA+0x054	R/W	Capture Control RegisterFor Pair Of PWM0CH2And PWM0CH3	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CFLIF3**	**PWM_FCAPDAT3 Latched Indicator Bit** When input channel 1 has a **falling** transition, PWM_FCAPDAT3 was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[22]	**CRLIF3**	**PWM_RCAPDAT3 Latched Indicator Bit** When input channel 1 has a **rising** transition, PWM_RCAPDAT3 was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[20]	**CAPIF3**	**Capture3 Interrupt Indication Flag** If channel 1 rising latch interrupt is enabled (CRLIEN3=1), a rising transition at input channel 1 will result in CAPIF3 to high; Similarly, a falling transition will cause CAPIF3 to be set high if channel 1 falling latch interrupt is enabled (CFLIEN3= 1). This flag is cleared by software writing a '1' to it.
[19]	**CAPEN3**	**Capture Channel 3 Transition Enable/Disable** 0 = Disable capture function on channel 1. 1 = Enable capture function on channel 1. When enabled, Capture function latches the PMW-counter to RCAPDAT (Rising latch) and FCAPDAT (Falling latch) registers on input edge transition. When disabled, Capture function is inactive as is interrupt.
[18]	**CFLIEN3**	**Channel 3 Falling Latch Interrupt Enable** 0 = Disable falling edge latch interrupt. 1 = Enable falling edge latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.
[17]	**CRLIEN3**	**Channel 3 Rising Latch Interrupt Enable** 0 = Disable rising edge latch interrupt. 1 = Enable rising edge latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.

[16]	**CAPINV3**	**Channel 3 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer
[7]	**CFLIF2**	**PWM_FCAPDAT2 Latched Indicator Bit** When input channel 0 has a **falling** transition, PWM_FCAPDAT2 was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zeroto it.
[6]	**CRLIF2**	**PWM_RCAPDAT2 Latched Indicator Bit** When input channel 0 has a **rising** transition, PWM_RCAPDAT2 was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zeroto it.
[4]	**CAPIF2**	**Capture2 Interrupt Indication Flag** If channel 0 rising latch interrupt is enabled (CRLIEN2=1), a rising transition at input channel 0 will result in CAPIF2 to high; Similarly, a falling transition will cause CAPIF2 to be set high if channel 0 falling latch interrupt is enabled (CFLIEN2= 1). This flag is cleared by software writing a '1' to it.
[3]	**CAPEN2**	**Capture Channel 2 Transition Enable/Disable** 0 = Disable capture function on channel 0. 1 = Enable capture function on channel 0. When enabled, Capture function latches the PMW-counter to RCAPDAT (Rising latch) and FCAPDAT (Falling latch) registers on input edge transition. When disabled, Capture function is inactive as is interrupt.
[2]	**CFLIEN2**	**Channel 2 Falling Latch Interrupt Enable ON/OFF** 0 = Disable falling latch interrupt. 1 = Enable falling latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.
[1]	**CRLIEN2**	**Channel 2 Rising Latch Interrupt Enable ON/OFF** 0 = Disable rising latch interrupt. 1 = Enable rising latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.
[0]	**CAPINV2**	**Channel 2 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer

Capture Rising Latch Register n (PWM0_RCAPDATn)

Register	Offset	R/W	Description	Reset Value
PWM0_RCAPDAT0	PWM_BA+0x058	R	Capture Rising Latch Register (Channel 0)	0x0000_0000
PWM0_RCAPDAT1	PWM_BA+0x060	R	Capture Rising Latch Register (Channel 1)	0x0000_0000
PWM0_RCAPDAT2	PWM_BA+0x068	R	Capture Rising Latch Register (Channel 2)	0x0000_0000
PWM0_RCAPDAT3	PWM_BA+0x070	R	Capture Rising Latch Register (Channel 3)	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RCAPDAT**	**Capture Rising Latch Register** In Capture mode, this register is latched with the value of the PWM counter on a rising edge of the input signal.

Capture Falling Latch Register n(PWM0_FCAPDATn)

Register	Offset	R/W	Description	Reset Value
PWM0_FCAPDAT0	PWM_BA+0x05C	R	Capture Falling Latch Register (Channel 0)	0x0000_0000
PWM0_FCAPDAT1	PWM_BA+0x064	R	Capture Falling Latch Register (Channel 1)	0x0000_0000
PWM0_FCAPDAT2	PWM_BA+0x06C	R	Capture Falling Latch Register (Channel 2)	0x0000_0000
PWM0_FCAPDAT3	PWM_BA+0x074	R	Capture Falling Latch Register (Channel 3)	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FCAPDAT**	**Capture Falling Latch Register** In Capture mode, this register is latched with the value of the PWM counter on a falling edge of the input signal.

Capture Input Enable Register (PWM0_CAPINEN)

Register	Offset	R/W	Description	Reset Value
PWM0_CAPINEN	PWM_BA+0x078	R/W	Capture Input Enable Register	0x0000_0000

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

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

PWM Output Enable Register (PWM0_POEN) for CH0~CH3

Register	Offset	R/W	Description	Reset Value
PWM0_POEN	PWM_BA+0x07C	R/W	PWM0 Output Enable Register for CH0~CH3	0x0000_0000

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

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

6.7.7 PWM1 Register Map

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

Register	Offset	R/W	Description	Reset Value
PWM1 Base Address: **PWM_BA = 0x4004_0000**				
PWM1_CLKPSC	PWM_BA+0x080	R/W	PWM Prescaler Register	0x0000_0000
PWM1_CLKDIV	PWM_BA+0x084	R/W	PWM Clock Select Register	0x0000_0000
PWM1_CTL	PWM_BA+0x088	R/W	PWM Control Register	0x0000_0000
PWM1_PERIOD0	PWM_BA+0x08C	R/W	PWM Counter Register 0	0x0000_0000
PWM1_CMPDAT0	PWM_BA+0x090	R/W	PWM Comparator Register 0	0x0000_0000
PWM1_CNT0	PWM_BA+0x094	R	PWM Data Register 0	0x0000_0000
PWM1_PERIOD1	PWM_BA+0x098	R/W	PWM Counter Register 1	0x0000_0000
PWM1_CMPDAT1	PWM_BA+0x09C	R/W	PWM Comparator Register 1	0x0000_0000
PWM1_CNT1	PWM_BA+0x0A0	R	PWM Data Register 1	0x0000_0000
PWM1_INTEN	PWM_BA+0x0C0	R/W	PWM Interrupt Enable Register	0x0000_0000
PWM1_INTSTS	PWM_BA+0x0C4	R/W	PWM Interrupt Flag Register	0x0000_0000
PWM1_CAPCTL01	PWM_BA+0x0D0	R/W	Capture Control Register For Pair Of PWM1CH0And PWM1CH1	0x0000_0000
PWM1_RCAPDAT0	PWM_BA+0x0D8	R	Capture Rising Latch Register (Channel 0)	0x0000_0000
PWM1_FCAPDAT0	PWM_BA+0x0DC	R	Capture Falling Latch Register (Channel 0)	0x0000_0000
PWM1_RCAPDAT1	PWM_BA+0x0E0	R	Capture Rising Latch Register (Channel 1)	0x0000_0000
PWM1_FCAPDAT1	PWM_BA+0x0E4	R	Capture Falling Latch Register (Channel 1)	0x0000_0000
PWM1_CAPINEN	PWM_BA+0x0F8	R/W	Capture Input Enable Register	0x0000_0000
PWM1_POEN	PWM_BA+0x0FC	R/W	PWM1 Output Enable Register for CH0~CH1	0x0000_0000

6.7.8 Register Description

PWM Pre-Scale Register (PWM1_CLKPSC)

Register	Offset	R/W	Description	Reset Value
PWM1_CLKPSC	PWM_BA+0x080	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							

Bits	Description	
[31:24]	**Reserved**	Reserved
[23:16]	**DTCNT01**	**Dead Zone Interval Register For Pair Of PWM1CH0And PWM1CH1** 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-ScalerPair Of PWM1CH0And PWM1CH1** Clock input is divided by (CLKPSC01 + 1) If CLKPSC01=0, then the pre-scaler output clock will be stopped. This implies PWM counter 0 and 1 will also be stopped.

PWM Clock Select Register (PWM1_CLKDIV)

Register	Offset	R/W	Description	Reset Value
PWM1_CLKDIV	PWM_BA+0x084	R/W	PWM Clock Select Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:4]	**CLKDIV1**	**Timer 1 Clock Source Selection** Select clock source divider for PWM timer 1. (Table is as CLKDIV0)
[3]	**Reserved**	Reserved.
[2:0]	**CLKDIV0**	**Timer 0 Clock Source Selection** Select clock source divider for PWM timer 0. 000 = 2. 001 = 4. 010 = 8. 011 = 16. 100 = 1.

PWM Control Register (PWM1_CTL)

Register	Offset	R/W	Description	Reset Value
PWM1_CTL	PWM_BA+0x088	R/W	PWM Control Register	0x0000_0000

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**CNTMODE1**	**PWM-Timer 1 Auto-Reload/One-Shot Mode** 0 = One-Shot Mode. 1 = Auto-load Mode. Note: A rising transition of this bit will cause PWM_PERIOD1 and PWM_CMPDAT1 to be cleared.
[10]	**PINV1**	**PWM-Timer 1 Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.
[9]	**Reserved**	Reserved.
[8]	**CNTEN1**	**PWM-Timer 1 Enable/Disable Start Run** 0 = Stop PWM-Timer 1. 1 = Enable PWM-Timer 1 Start/Run.
[7:5]	**Reserved**	Reserved.
[4]	**DTEN01**	**Dead-Zone 01 Generator Enable/DisableFor Pair Of PWM1CH0And PWM1CH1** 0 = Disable. 1 = Enable. Note: When Dead-Zone Generator is enabled, the pair of PWM1CH0 and PWM1CH1become a complementary pair.
[3]	**CNTMODE0**	**PWM-Timer 0 Auto-Reload/One-Shot Mode** 0 = One-Shot Mode. 1 = Auto-reload Mode. Note: A rising transition of this bit will cause PWM_PERIOD0 and PWM_CMPDAT0 to be cleared.
[2]	**PINV0**	**PWM-Timer 0 Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.

[1]	**Reserved**	Reserved.
[0]	**CNTEN0**	**PWM-Timer 0 Enable/Disable Start Run** 0 = Stop PWM-Timer 0 Running. 1 = Enable PWM-Timer 0 Start/Run.

PWM Counter Register 1-0 (PWM1_PERIOD1-0)

Register	Offset	R/W	Description	Reset Value
PWM1_PERIOD0	PWM_BA+0x08C	R/W	PWM Counter Register 0	0x0000_0000
PWM1_PERIOD1	PWM_BA+0x098	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]							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**PWM Timer Loaded Value** PERIOD determines the PWM period. PWM frequency = PWM1CHx_CLK/[(prescale+1)*(clock divider)*(PERIOD +1)]; where x, depends on selected PWM channel. For Edge-aligned type: ● Duty ratio = (CMPDAT +1)/(PERIOD +1). ● CMPDAT>= PERIOD: PWM output is always high. ● CMPDAT<PERIOD: PWM low width = (PERIOD - CMPDAT) unit; PWM high width = (CMPDAT +1) unit. ● CMPDAT = 0: PWM low width = (PERIOD) unit; PWM high width = 1 unit. **Note:** Any write to PERIOD will take effect in next PWM cycle. **Note:**When PERIODvalue is set to 0, PWM output is always high.

PWM Comparator Register 1-0 (PWM1_CMPDAT3-0)

Register	Offset	R/W	Description	Reset Value
PWM1_CMPDAT0	PWM_BA+0x090	R/W	PWM Comparator Register 0	0x0000_0000
PWM1_CMPDAT1	PWM_BA+0x09C	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**
CMPDATx [15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPDATx [7:0]							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPDATx**	**PWM Comparator Register** CMPDATX determines the PWM duty. PWM frequency = PWM1CHx_CLK/[(prescale+1)*(clock divider)*(PERIOD +1)]; where x, depends on selected PWM channel. For Edge-aligned type: ● Duty ratio = (CMPDATX+1)/(PERIOD +1). ● CMPDATX>= PERIOD: PWM output is always high. ● CMPDATX<PERIOD: PWM low width = (PERIOD -CMPDATX) unit; PWM high width = (CMPDATX +1) unit. ● CMPDATX = 0: PWM low width = (PERIOD) unit; PWM high width = 1 unit. **Note:** Any write to PERIODwill take effect in next PWM cycle.

PWM Data Register 1-0 (PWM1_CNT1~PWM1_CNT0)

Register	Offset	R/W	Description	Reset Value
PWM1_CNT0	PWM_BA+0x094	R	PWM Data Register 0	0x0000_0000
PWM1_CNT1	PWM_BA+0x0A0	R	PWM Data 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**
CNT[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[7:0]							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNT**	**PWM Data Register** User can monitor PDR to know the current value in 16-bit counter.

PWM Interrupt Enable Register (PWM1_INTEN)

Register	Offset	R/W	Description	Reset Value
PWM1_INTEN	PWM_BA+0x0C0	R/W	PWM Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**PIEN1**	**PIEN0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[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 (PWM1_INTSTS)

Register	Offset	R/W	Description	Reset Value
PWM1_INTSTS	PWM_BA+0x0C4	R/W	PWM Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**PIF1**	**PIF0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**PIF1**	**PWM Timer 1 Interrupt Flag** Flag is set by hardware when PWM1CH1 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 PWM1CH0 down counter reaches zero, software can clear this bit by writing '1' to it.

Note: User can clear each interrupt flag by writing 1 to corresponding bit in PWM_INTSTS.

Capture Control Register (PWM1_CAPCTL01)

Register	Offset	R/W	Description	Reset Value
PWM1_CAPCTL01	PWM_BA+0x0D0	R/W	Capture Control RegisterFor Pair Of PWM1CH0And PWM1CH1	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CFLIF1**	**PWM_FCAPDAT1 Latched Indicator Bit** When input channel 1 has a **falling** transition, PWM_FCAPDAT1 was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[22]	**CRLIF1**	**PWM_RCAPDAT1 Latched Indicator Bit** When input channel 1 has a **rising** transition, PWM_RCAPDAT1 was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[20]	**CAPIF1**	**Capture1 Interrupt Indication Flag** If channel 1 rising latch interrupt is enabled (CRLIEN1=1), a rising transition at input channel 1 will result in CAPIF1 to high; Similarly, a falling transition will cause CAPIF1 to be set high if channel 1 falling latch interrupt is enabled (CFLIEN1= 1). This flag is cleared by software writing a '1' to it.
[19]	**CAPEN1**	**Capture Channel 1 Transition Enable/Disable** 0 = Disable capture function on channel 1. 1 = Enable capture function on channel 1. When enabled, Capture function latches the PMW-counter to RCAPDAT (Rising latch) and FCAPDAT (Falling latch) registers on input edge transition. When disabled, Capture function is inactive as is interrupt.
[18]	**CFLIEN1**	**Channel 1 Falling Latch Interrupt Enable** 0 = Disable falling edge latch interrupt. 1 = Enable falling edge latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.
[17]	**CRLIEN1**	**Channel 1 Rising Latch Interrupt Enable** 0 = Disable rising edge latch interrupt. 1 = Enable rising edge latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.

[16]	**CAPINV1**	**Channel 1 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer
[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 zeroto 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 zeroto it.
[4]	**CAPIF0**	**Capture0 Interrupt Indication Flag** If channel 0 rising latch interrupt is enabled (CRLIEN0=1), a rising transition at input channel 0 will result in CAPIF0 to high; Similarly, a falling transition will cause CAPIF0 to be set high if channel 0 falling latch interrupt is enabled (CFLIEN0= 1). This flag is cleared by software writing a '1' to it.
[3]	**CAPEN0**	**Capture Channel 0 Transition Enable/Disable** 0 = Disable capture function on channel 0. 1 = Enable capture function on channel 0. When enabled, Capture function latches the PMW-counter to RCAPDAT (Rising latch) and FCAPDAT (Falling latch) registers on input edge transition. When disabled, Capture function is inactive as is interrupt.
[2]	**CFLIEN0**	**Channel 0 Falling Latch Interrupt Enable ON/OFF** 0 = Disable falling latch interrupt. 1 = Enable falling latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.
[1]	**CRLIEN0**	**Channel 0 Rising Latch Interrupt Enable ON/OFF** 0 = Disable rising latch interrupt. 1 = Enable rising latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.
[0]	**CAPINV0**	**Channel 0 Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before Capture timer

Capture Rising Latch Register n (PWM1_RCAPDATn)

Register	Offset	R/W	Description	Reset Value
PWM1_RCAPDAT0	PWM_BA+0x0D8	R	Capture Rising Latch Register (Channel 0)	0x0000_0000
PWM1_RCAPDAT1	PWM_BA+0x0E0	R	Capture Rising Latch Register (Channel 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**
RCAPDAT [15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RCAPDAT [7:0]							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RCAPDAT**	**Capture Rising Latch Register** In Capture mode, this register is latched with the value of the PWM counter on a rising edge of the input signal.

Capture Falling Latch Register n(PWM1_FCAPDATn)

Register	Offset	R/W	Description	Reset Value
PWM1_FCAPDAT0	PWM_BA+0x0DC	R	Capture Falling Latch Register (Channel 0)	0x0000_0000
PWM1_FCAPDAT1	PWM_BA+0x0E4	R	Capture Falling Latch Register (Channel 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**
FCAPDAT [15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FCAPDAT [7:0]							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FCAPDAT**	**Capture Falling Latch Register** In Capture mode, this register is latched with the value of the PWM counter on a falling edge of the input signal.

Capture Input Enable Register (PWM1_CAPINEN)

Register	Offset	R/W	Description	Reset Value
PWM1_CAPINEN	PWM_BA+0x0F8	R/W	Capture Input Enable Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**CAPINEN**	**Capture Input Enable Register** 0 : OFF (GPA[15:14] pin input disconnected from Capture block) 1 : ON (GPA[15:14] pin, if in PWM alternative function, will be configured as an input and fed to capture function) CAPINEN[1:0] Bit [1][0] Bit x1 ： Capture channel 0 is from GPA [14] Bit 1x ： Capture channel 1 is from GPA [15]

PWM Output Enable Register (PWM1_POEN) for CH0~CH1

Register	Offset	R/W	Description	Reset Value
PWM1_POEN	PWM_BA+0x0FC	R/W	PWM1 Output Enable Register for CH0~CH1	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						**POEN1**	**POEN0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**POEN1**	**PWM1CH1 Output Enable Register** 0 = Disable PWM1CH1 output to pin. 1 = Enable PWM1CH1 output to pin. Note: The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)
[0]	**POEN0**	**PWM1CH0 Output Enable Register** 0 = Disable PWM1CH0 output to pin. 1 = Enable PWM1CH 0 output to pin. Note: The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)

6.8 Real Time Clock (RTC)

6.8.1 Overview

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

The RTC unit supports periodic Time-Tick and Alarm-Match interrupts. The periodic interrupt has 8 period options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second which are selected by RTC_TICK.TTR. When RTC counter in RTC_TIME and RTC_CAL is equal to alarm setting registers RTC_TALM and RTC_CALM, the alarm interrupt flag (RTC_INTSTS.AIF) is set and the alarm interrupt is requested if the alarm interrupt is enabled (RTC_INTEN.AIER=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.TWKE=1).

6.8.2 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 of RTC clock source compensate by RTC Frequency Compensation Register (FCR) register
- All time and calendar message expressed in BCD format
- Supports periodic RTC Time Tick interrupt with 8 period interval options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second
- Supports RTC Time Tick and Alarm Match interrupt
- Support CPU wakeup from sleep or standby power-down mode.

6.8.3 Block Diagram

The block diagram of Real Time Clock is depicted as follows:

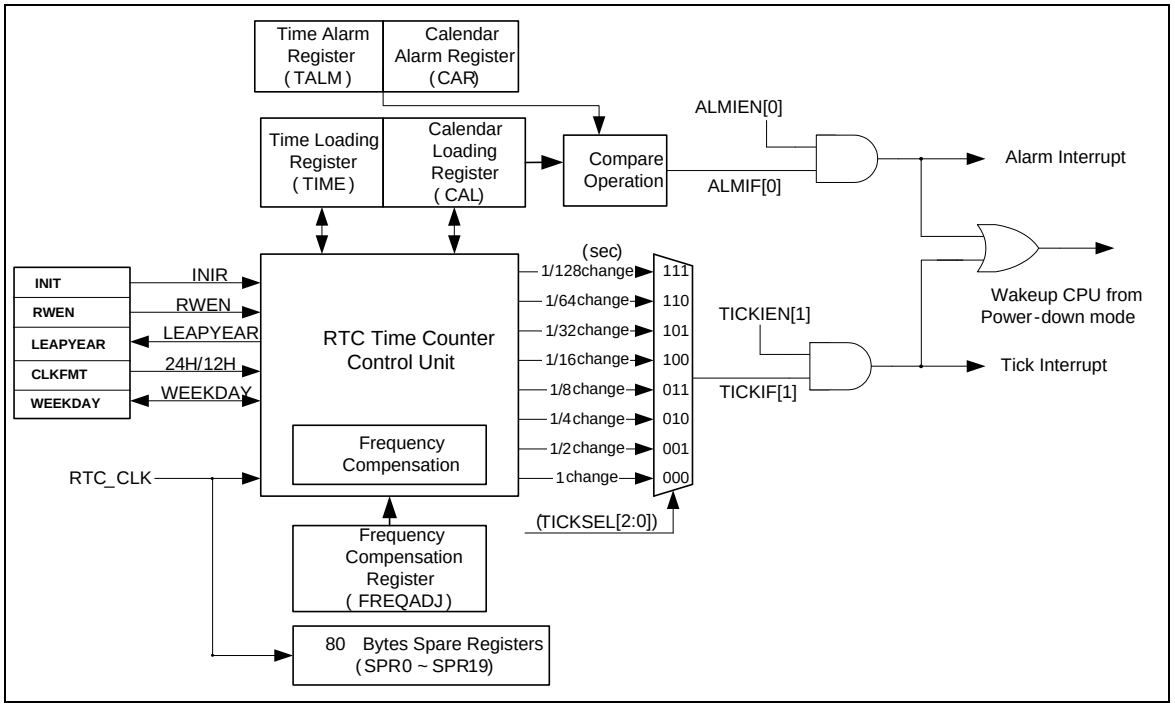


Figure 6-33 RTC Block Diagram

6.8.4 Functional Description

6.8.4.1 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.

6.8.4.2 RTC Read/Write Enable

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

6.8.4.3 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 to32776Hz. 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.

Integer Part Of Detected Value	INTEGER (FCR[11:8])	Integer Part Of Detected Value	INTEGER (FCR[11:8])
32776	1111	32768	0111
32775	1110	32767	0110
32774	1101	32766	0101
32773	1100	32765	0100
32772	1011	32764	0011
32771	1010	32763	0010
32770	1001	32762	0001
32769	1000	32761	0000

Following are the compensation examples for the real RTC source clock is higher or lower than 32768 Hz.

Example 1: (RTC Source Clock >32768 Hz) **RTC Source Clock Measured: 32773.65 Hz (> 32768 Hz)** **Integer Part: 32773 => 0x8005** **INTEGER (FCR [11:8] Integer Part)= 0x05 – 0x01 + 0x08 = 0x0c** **Fraction Part: 0.65** **FRACTION (FCR [5:0] Fraction Part) = 0.65 X 60 = 39 = 0x27** **RTC FCR Register Should Be As 0xC27.**
Example 2:(RTC source clock ≦ 32768 Hz)

RTC source clock measured: 32765.27Hz (≦ 32768 Hz)

Integer part: 32765 => 0x7FFD

INTEGER (FCR [11:8] Integer Part) = 0x0D – 0x01 – 0x08 = 0x04

Fraction part: 0.27

FRACTION (FCR [5:0] Fraction Part) = 0.27 x 60 = 16.2= 0x10

RTC FCR register should be as 0x410.

6.8.4.4 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.

6.8.4.5 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. The 12/24 hour time scale selection depends on RTC_CLKFMT bit 0.

24-Hour Time Scale (24H_12H = 1)	12-Hour Time Scale (24H_12H = 0)	24-Hour Time Scale (24H_12H = 1)	12-Hour Time Scale (PM Time + 20) (24H_12H = 0)
00	12 (AM12)	12	32 (PM12)
01	01 (AM01)	13	21 (PM01)
02	02 (AM02)	14	22 (PM02)
03	03 (AM03)	15	23 (PM03)
04	04 (AM04)	16	24 (PM04)
05	05 (AM05)	17	25 (PM05)
06	06 (AM06)	18	26 (PM06)
07	07 (AM07)	19	27 (PM07)
08	08 (AM08)	20	28 (PM08)
09	09 (AM09)	21	29 (PM09)
10	10 (AM10)	22	30 (PM10)
11	11 (AM11)	23	31 (PM11)

6.8.4.6 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.

6.8.4.7 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.TTR[2:0]. When periodic time tick interrupt is enabled by setting RTC_INTEN.TIER to 1, the Periodic Time Tick Interrupt is requested as selected by RTC_TICK register.

6.8.4.8 Alarm 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.AIER=1) the alarm interrupt is also requested.

6.8.4.9 *Application* Note

1. All data in RTC_TALM, RTC_CALM, RTC_TIME and RTC_CAL registers are all BCD counter.
2. User has to make sure that the loaded values are reasonable. For example, load RTC_CAL as 201a (year), 13 (month), 00 (day), or CLR does not match with DWR, etc.
3. Registers value after powered on or reset:

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-hour 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.

6.8.5 Register Map

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

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

6.8.6 Register Description

RTC Initiation Register (RTC_INIT)

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

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

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

RTC Access Enable Register (RTC_RWEN)

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

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**RWENF**	**RTC Register Access Enable Flag (Read Only)** 1 = RTC register read/write enable. 0 = RTC register read/write disable. This bit will be set after RWEN[15:0] register is set to 0xA965, it will clear automatically in 512 RTC clock cycles or RWEN[15:0] != 0xA965.
[15:0]	**RWEN**	**RTC Register Access Enable Password (Write Only)** 0xA965 = Enable RTC access Others = 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					

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**INTEGER**	**Integer Part** Register should contain the value (INT(F_{actual}) – 32761) Ex: Integer part of detected value = 32772, RTC_FREQADJ.INTEGER = 32772-32761 =11 (1011b) The range between 32761 and 32776
[5:0]	**FRACTION**	**Fraction Part** Formula = (fraction part of detected value) x 60. **Note:** Digit in FCR must be expressed as hexadecimal number..

Note: This register can be read back after the (RTC Register Access Enable Flag) is active.

RTC Time Load Register (RTC_TIME)

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

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

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

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

Note:

1. RTC_TIME is a BCD digit counter and RTC controller will not check the loaded datais reasonable or not.
2. The reasonable value range is listed in the parenthesis.
3. When RTC runs as 12-hour time scale mode, the high bit of TENHR field means AM/PM.

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**			

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

Note:

1. RTC_CAL is a BCD digit counter and RTC will not checkthe loaded datais reasonable or not.
2. The reasonable value range is listed in the parenthesis.

RTC Time Scale Selection Register (RTC_CLKFMT)

Register	Offset	R/W	Description	Reset Value
RTC_CLKFMT	RTC_BA+0x014	R/W	Time Scale Selection Register	0x0000_0001

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**24HEN**	**24-Hour / 12-Hour Time Scale Selection** It indicates that RTC_TIME and RTC_TALMcounter are in 24-hour time scale or 12-hour time scale. 0 = 24-hour time scale selected. 1 = 24-hour time scale selected.

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**WEEKDAY**	**Day Of The Week Register** 000 = Sunday. 001 = Monday. 010 = Tuesday. 011 = Wednesday. 100 = Thursday. 101 = Friday. 110 = Saturday. 111 = Reserved.

RTC Time Alarm Register (RTC_TALM)

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

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

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

Note:

1. This register can be read back after the (RTC Register Access Enable Flag) is active.
2. RTC_TALM is a BCD digit counter and RTC controller will not check the loaded datais reasonable or not.
3. The reasonable value range is listed in the parenthesis.
4. When RTC runs as 12-hour time scale mode, the high bit of 10HR field means AM/PM.

RTC Calendar Alarm Register (RTC_CALM)

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

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

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

Note:

1. This register can be read back after the (RTC Register Access Enable Flag) is active.
2. RTC_CALM is a BCD digit counter and RTC will not check the loaded datais reasonable or not.
3. The reasonable value range is listed in the parenthesis.

RTC Leap year Indication Register (RTC_LEAPYEAR)

Register	Offset	R/W	Description	Reset Value
RTC_LEAPYEAR	RTC_BA+0x024	R	Leap year Indicator Register	0x0000_0000

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

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

RTC Interrupt Enable Register (RTC_INTEN)

Register	Offset	R/W	Description	Reset Value
RTC_INTEN	RTC_BA+0x028	R/W	RTC Interrupt Enable Register	0x0000_0000

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

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

RTC Interrupt Indication Register (RTC_INTSTS)

Register	Offset	R/W	Description	Reset Value
RTC_INTSTS	RTC_BA+0x02C	R/W	RTC Interrupt Indicator Register	0x0000_0000

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

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

RTC Time-Tick Register (RTC_TICK)

Register	Offset	R/W	Description	Reset Value
RTC_TICK	RTC_BA+0x030	R/W	RTC TimeTick Register	0x0000_0000

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

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

6.9 Serial Peripheral Interface (SPI)

6.9.1 Overview

The Serial Peripheral Interface (SPI) is a synchronous serial data communication protocol that operates in full duplex mode. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The ISD9300 series contains up to four sets of SPI controllers performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. Each set of SPI controller can be configured as a master or a slave device.

The SPI controller can be set as a master with up to 2 slave select (SSB) address lines to access two slave devices;it also can be set as a slave controlled by an off-chip master device.In addition the SPI interface supports Dual and Quad IO as is common on serial flash memories, where data is transferred 2 or 4 bits per clock period.

6.9.2 Features

- SupportsMaster or Slave mode operation
- Supports one or two channels of serial data
- 8 word FIFO on transmit and receive
- Supports MSB first or LSB first transfer sequence
- Twoslave select lines in Master mode, single device/slave select line in slave mode
- SupportsByte or Word Suspend mode
- SupportsPDMA transfer

6.9.3 Block Diagram

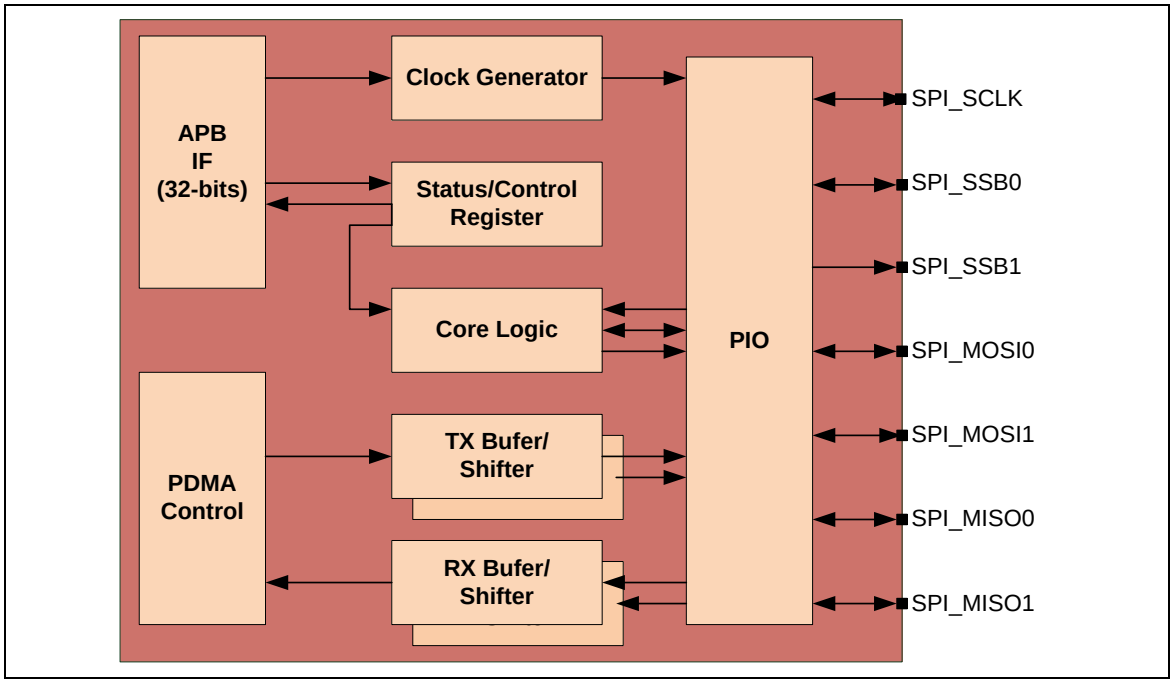


Figure 6-34 SPI Block Diagram

6.9.4 FunctionalDescription

6.9.4.1 Terminology

SPI Peripheral Clock and SPI Bus Clock

The SPI controller derives its clock source from the system HCLK as determined by the CLK_SEL1 register. The frequency of the SPI master clock is determined by the divisor ratio SPI_CLKDIV.

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

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 ISD9300 generates SCLK and SSB signals to access one or more slave devices. In slave mode the ISD9300 monitors SCLK and SSB signals to respond to data transactions from an off-chip master. The signal directions are summarized in the application block diagrams.

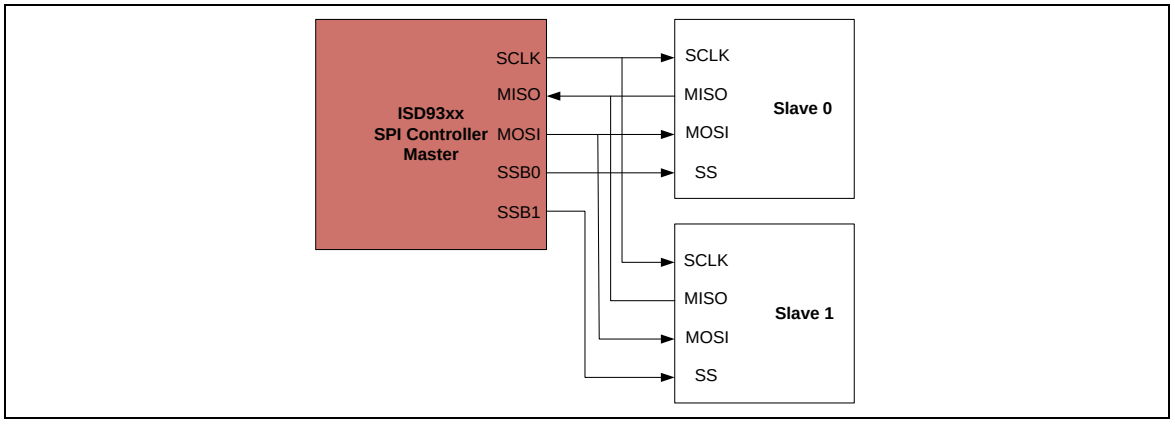


Figure 6-35 SPI Master Mode Application Block Diagram

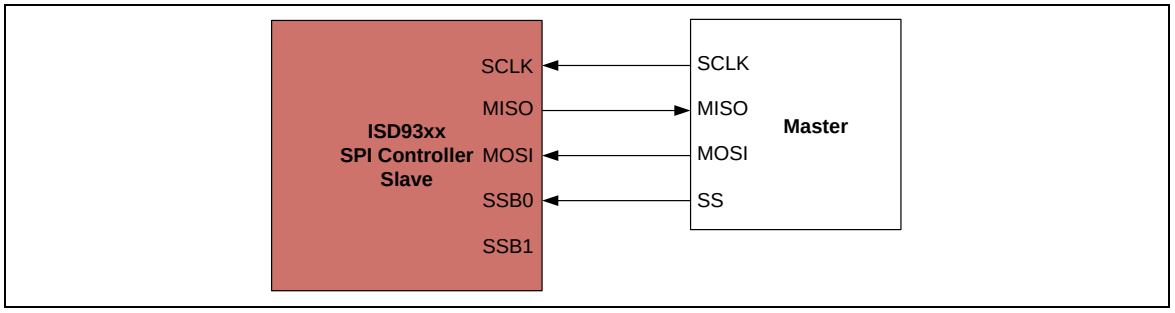


Figure 6-36 SPI Slave Mode Application Block Diagram

Slave Select

In master mode, the SPI controller canaddress up to twooff-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 selectsignal can be programmed to be active low or active high via the SPI_SSCTL.SSACTPOL bit.In addition the SPI_SSCTL.SS_LTRIG 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.ASS 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.ASS 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.

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

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

Serial Clock

In master mode, writing a divisor into the SPI_CLKDIV.DIVIDERregister 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.

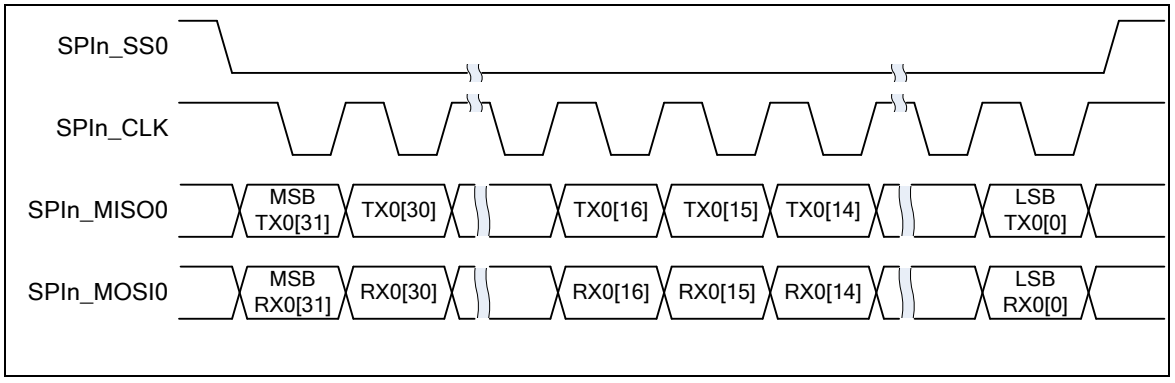


Figure 6-37 32-Bit in One Transaction

LSB/MSB First

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

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

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

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

Word Suspend

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

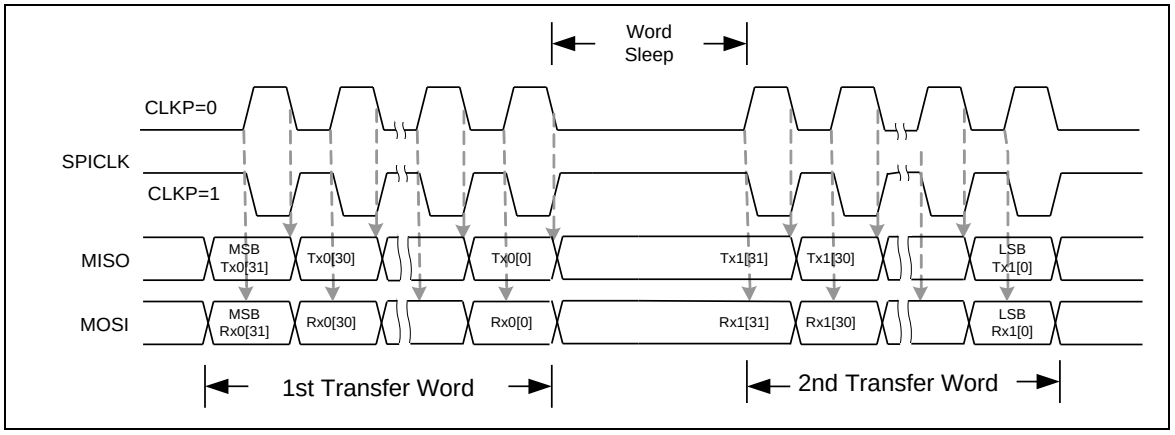


Figure 6-38 Word Sleep Suspend Mode

6.9.4.2 Byte Reorder Function

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 byteis processed first. For DWIDTH =0 (32 bits transfer), the sequence of transmitted/received data will be BYTE0, BYTE1, BYTE2, and then BYTE3. If DWIDTH is set to 24-bits, the sequence will be

BYTE0, BYTE1, and BYTE2.For Quad and Dual SPI transactions, REORDER is only valid for receive operation. For transmit in Dual/Quad modes, REORDER must be set to 0.

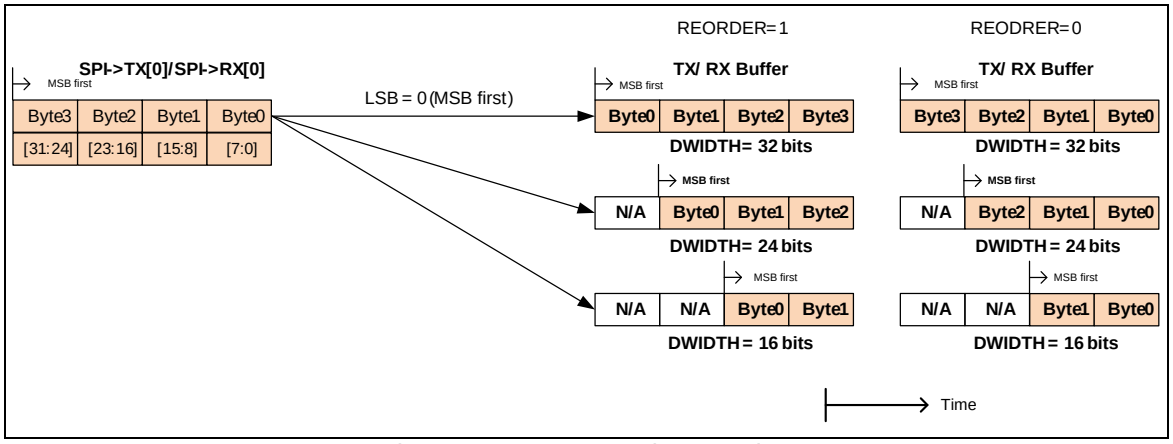


Figure 6-39 Byte Reorder Function

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 intuiSPI_DATA[]={0x01020304, 0x05060708};

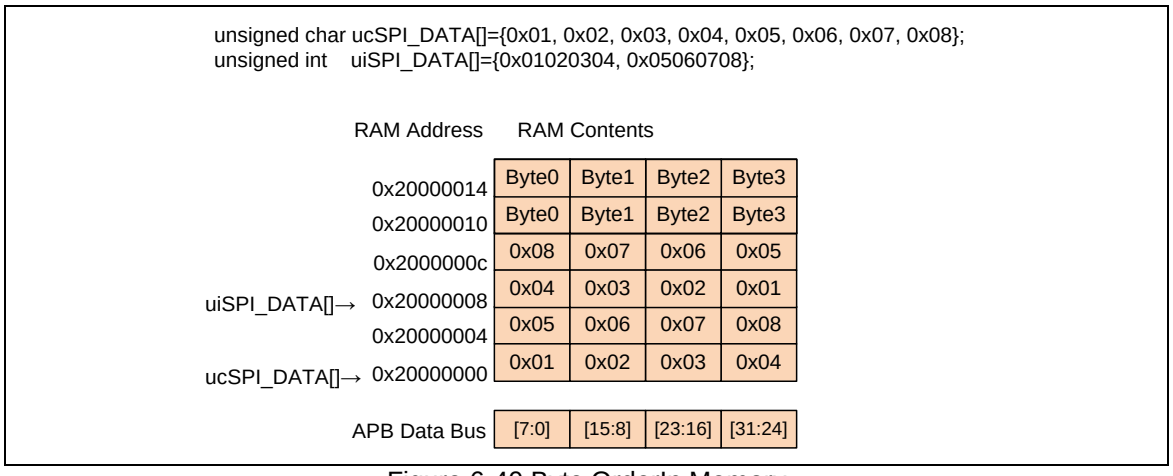


Figure 6-40 Byte OrderIn Memory

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

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

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

It would be common that a byte array of data is constructed but user, for efficiency, wishes to

transfer data to SPI via word transfers. Consider the situation of where a int pointer points to the byte data array.

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

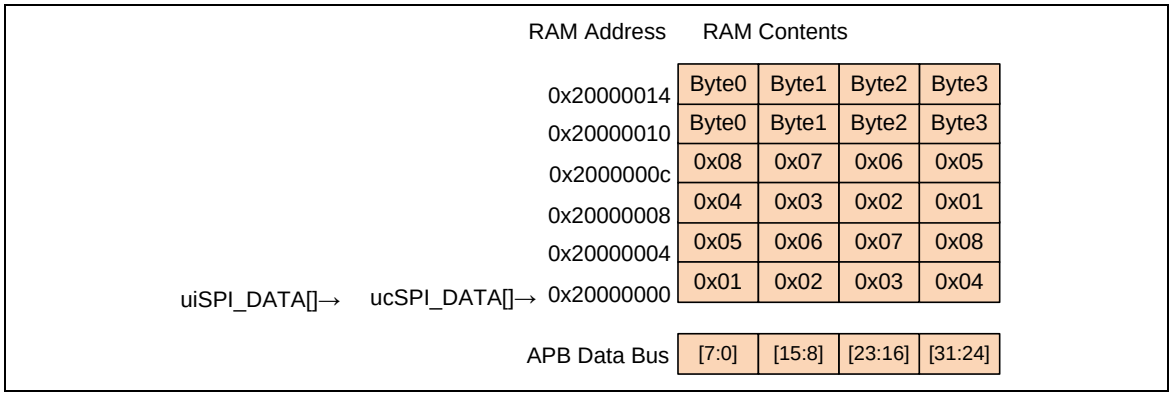


Figure 6-41 Byte Reorder In Memory

6.9.4.3 Interrupt

- SPI unit transfer interrupt

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

- SPI Slave 3-wire mode start interrupt

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

- Receive FIFO time-out interrupt

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

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

- Slave Error 0 interrupt

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

Note:

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

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

- Slave Under-run and Slave Error 1 interrupts

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

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

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

- Receive Over-run interrupt

In Slave mode, if the receive FIFO buffer contains 8 unread data, the RXFULL flag will be set to 1 and the RXOVIF will be set 1 if there is more serial data is received from SPIMOSI and the RXOVIF will be set to 1 and follow-up data will be dropped. The SPI controller will issue an interrupt if the SLVOVR_INTEN, SPI_FIFOCTL[5], is set to 1.

- Receive FIFO time-out interrupt

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

- Transmit FIFO interrupt

In FIFO mode, if the valid data countof the transmit FIFO bufferislessthan or equal to thesetting value ofTXTH, the transmit FIFO interrupt flag will be set to 1. The SPI controller will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, SPI_FIFOCTL[3], is set to 1.

- Receive FIFO interrupt

In FIFO mode, if the valid data countof the receive FIFO bufferislargerthan thesetting value ofRXTH, the receive FIFO interrupt flag will be set to 1. The SPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, SPI_FIFOCTL[2], is set to 1.

6.9.4.4 Slave 3-wire Mode

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

6.9.4.5 2-bit Transfer Mode

The SPI controller supports 2-bit Transfer mode when setting the TWOBIT bit (SPI_CTL[16]) to 1. In 2-bit mode, the SPI controllerperforms full duplex data transfer. In other words, the2-bit serial data can be transmitted and received simultaneously.

For example, in Master mode, the first data written to the TX FIFO will be transmitted through SPI_MOSI0and the second data writtento the TX FIFO will be transmitted through the SPI_MOSI1 pin. After transmission, the first read of RX FIFO will result in the data received from SPI_MISO0 pin and the second read the data received from SPI_MISO1 pin.

In Slave mode, the first two data stored in theTX FIFO will be transmitted through the SPI_MISO0 and SPI_MISO1 pin respectively. Concurrently, the RX FIFO will store the data received from the

SPI_MOSI0 and SPI_MOSI1 pin, same as Master mode.

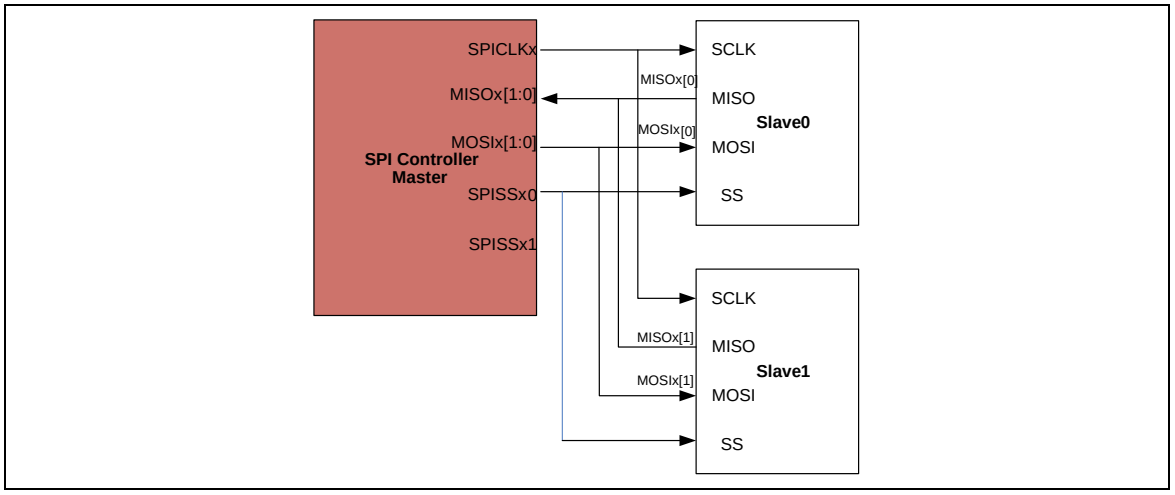


Figure 6-42 2-bit System Architecture

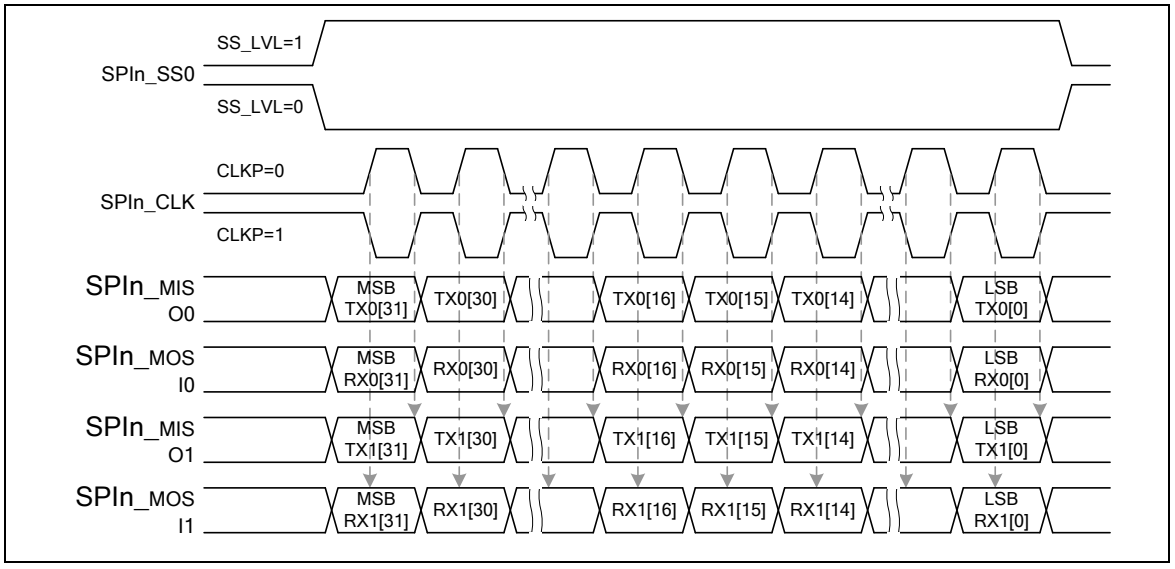


Figure 6-43 2-bit Transfer Mode (Slave Mode)

6.9.4.6 Dual I/O Mode

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

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

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

is set as 1 and QDIODIR is set as 0, both the SPI_MISO0 and SPI_MOSI0 will be set as data input ports.

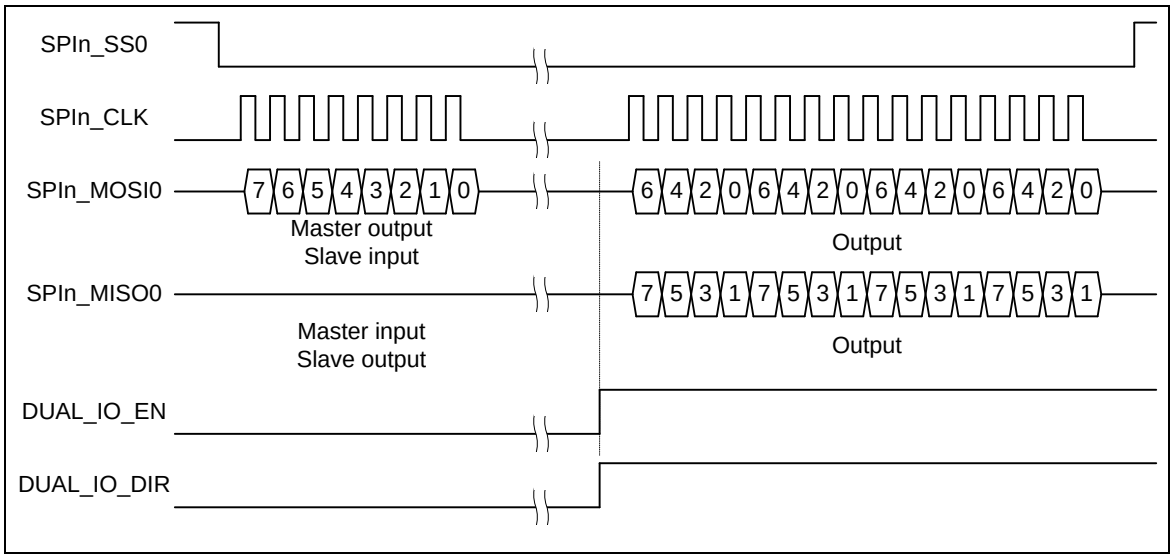


Figure 6-44 Bit Sequence of Dual Output Mode

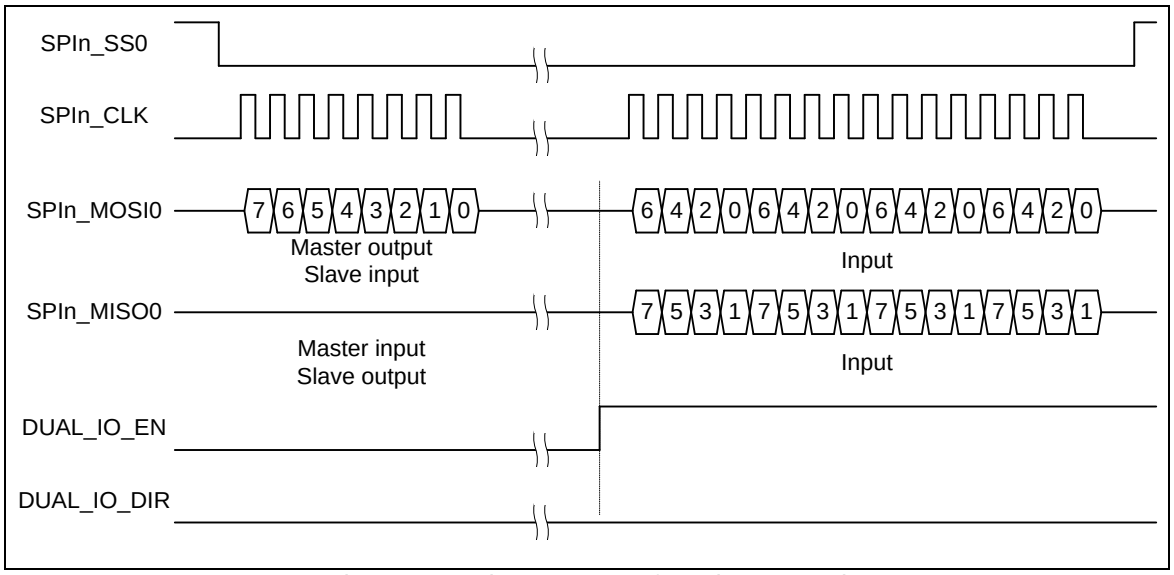


Figure 6-45 Bit Sequence of Dual Input Mode

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

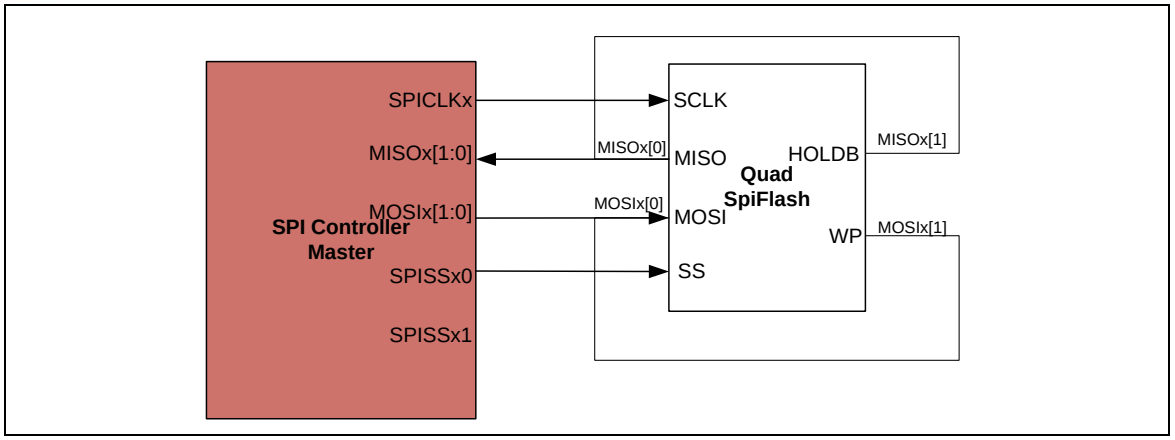


Figure 6-46 Quad ModeSystem Architecture

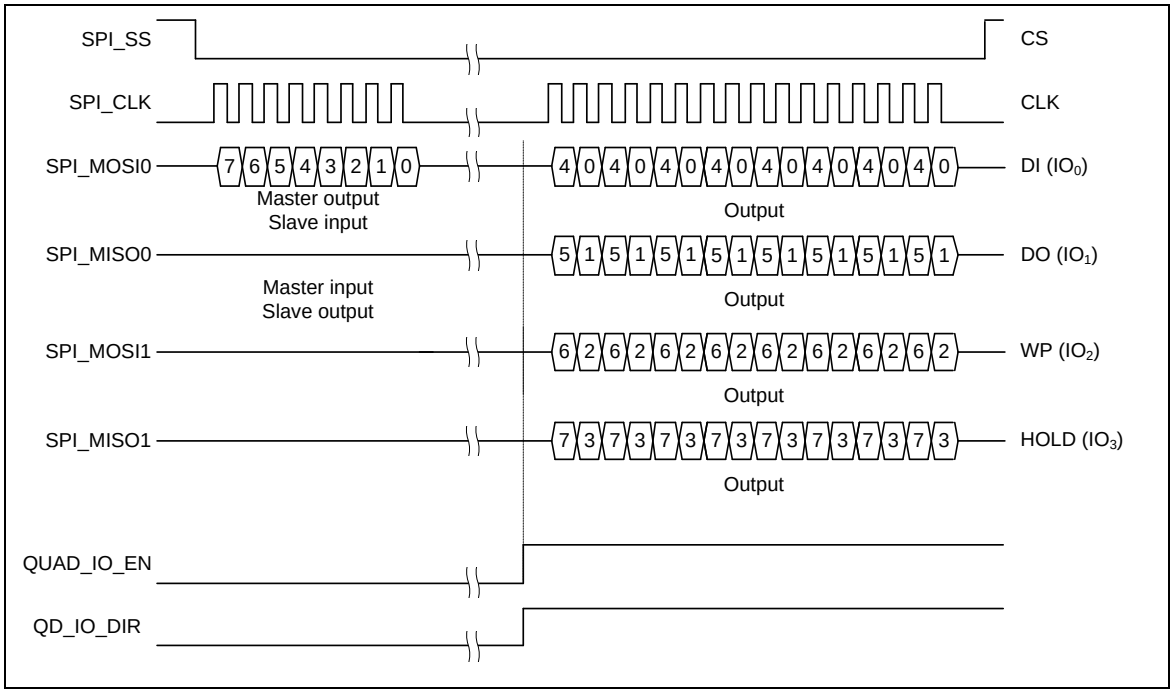


Figure 6-47 Bit Sequence of Quad Output Mode

6.9.4.7 FIFO Mode

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

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

The received FIFO buffer is also an 8-level depth, 32-bit wide, first-in, first-out register buffer. The

receive control logic will store the received data to this buffer. The FIFO buffer data can be read from SPI_RX register by software. There are FIFO related status bits, like RXEMPTY and RXFULL, to indicate the current status of FIFO buffer.

The transmitting and receiving threshold can be set through software by setting the TXTH, SPI_FFCTL[30:24], and RXTH, SPI_FIFOCTL[26:24], settings. When the count of valid data stored in transmit FIFO buffer is less than or equal to TXTH setting, the TXTHIF, SPI_STATUS[18], bit will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RXTH setting, the RXTHIF, SPI_STATUS[10], bit will be set to 1.

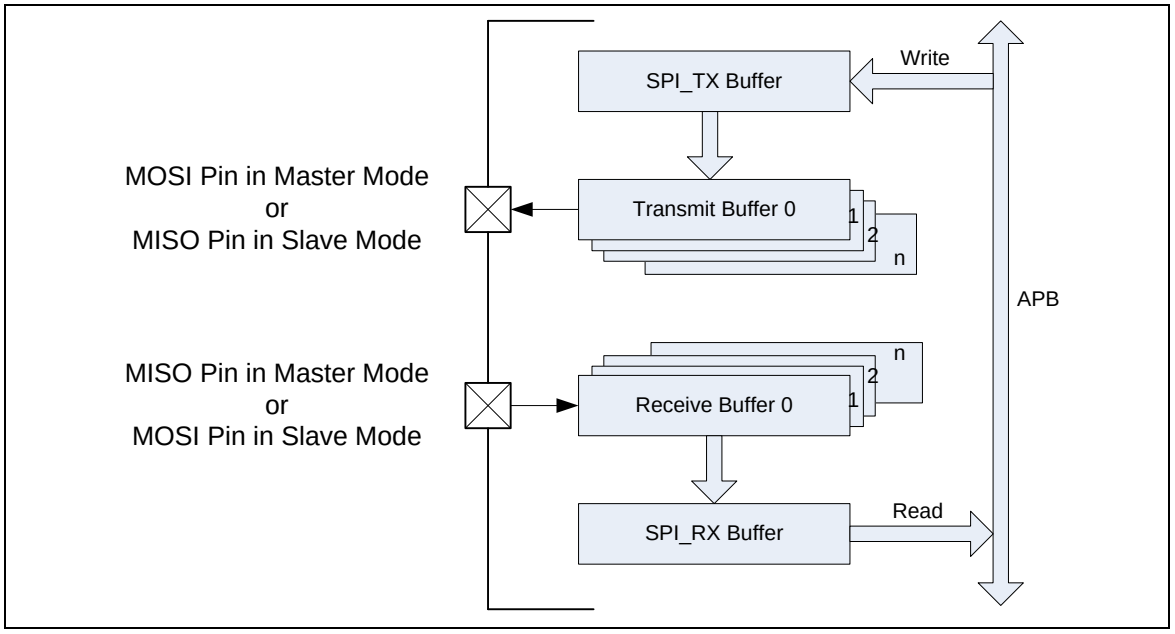


Figure 6-48 FIFO Mode Block Diagram

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

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

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

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

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

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

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

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

6.9.4.8 DMA Receive Mode

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

6.9.5 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.

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

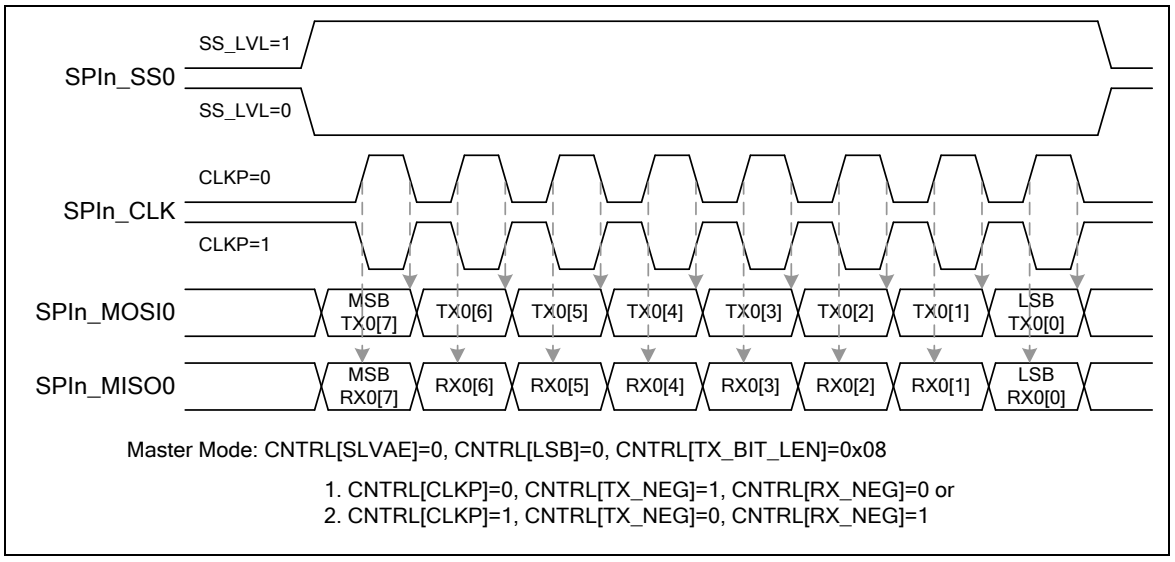


Figure 6-49 SPI Timing in Master Mode

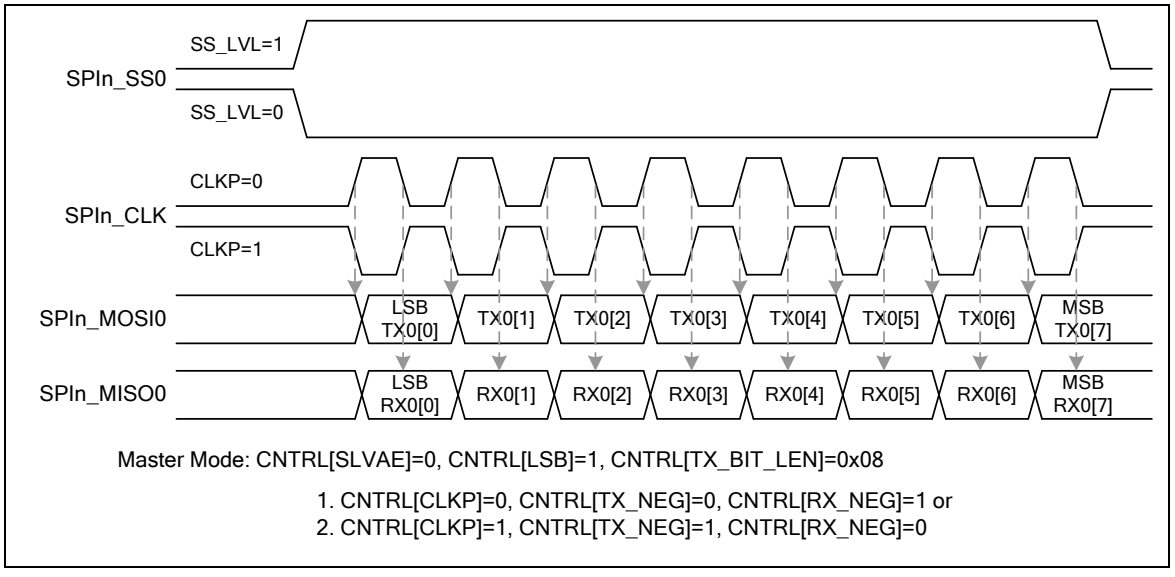


Figure 6-50 SPI Timing in Master Mode (Alternate Phase of SPI Bus Clock)

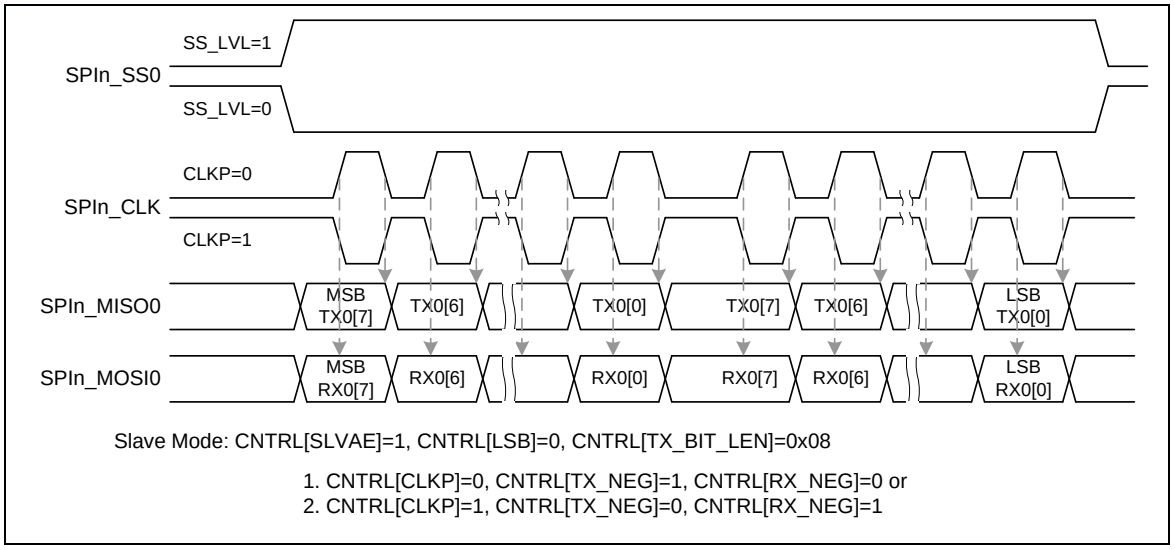


Figure 6-51 SPI Timing in Slave Mode

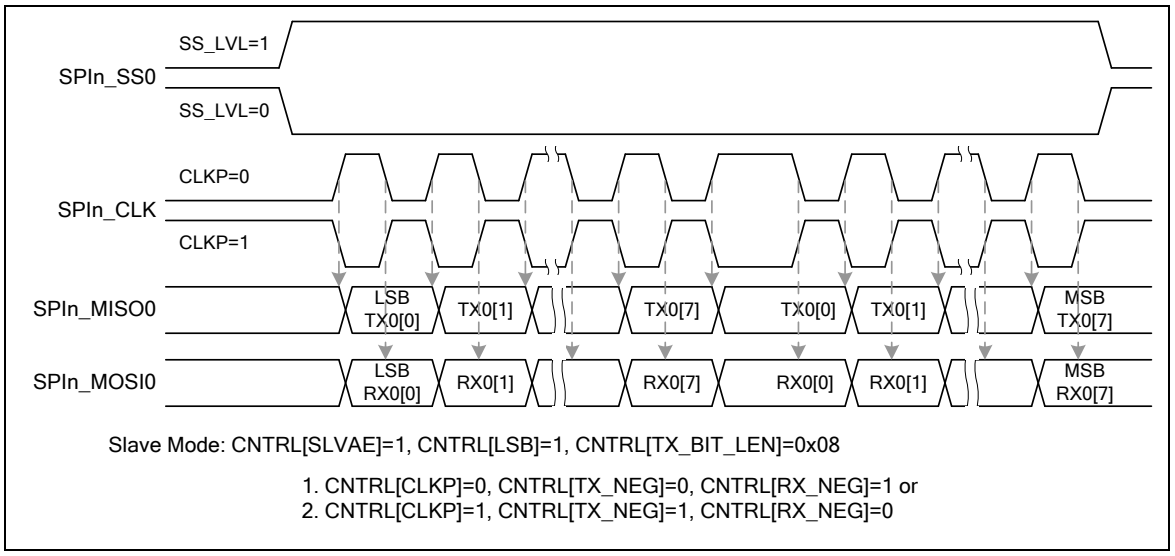


Figure 6-52 SPI Timing in Slave Mode (Alternate Phase of SPI Bus Clock)

6.9.6 Programming Examples

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

- Data bit is latched on positive edge of SPI clock.
- Data bit is driven on negative edge of SPI clock.
- Data is transferred from MSB first.
- SPI bus clock is idle at low state.
- Only one byte of data to be transmitted/received in a transaction.
- Uses the first SPI slave select pin to connect with an off-chip slave device. The slave select signal is active low.
- SCLK frequency is 10MHz

The operation flow is as follows.

1) Write a divisor into the SPI_CLKDIV register to determine the output frequency of serial clock. Driver function DrvSPI_SetClock(0,10000000,0) can be used to achieve this.
2) Configure the SPI_SSCTL register to address device. For example to manually address, set SPI_SSCTL.ASS=0, SPI_SSCTL.SSR_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.RXNEG=0 so that data is latched into device on positive edge of SCLK, set SPI_CTL.DWIDTH=8 and SPI_CTL.TX_NUM=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_TX register. If only doing a receive, write a dummy byte to SPI_TX register.
6) Enable the SPI_CTL.SPIEN bit to start the data transfer over the SPI interface.
7) Wait for SPI transfer to finish. Can be interrupt driven (if the interrupt enable SPI_CTL.IE bit is set) or by polling the GO_BUSY bit which will be cleared to 0 by hardware automatically at end of transmission.
8) Read out the received one byte data from SPI_RX
9) Go to 5) to continue another data transfer or set SPI_SSCTL.SS=0 to deactivate the off-chip slave devices.

Example 2: The SPI controller is set as a slave device and connects with an off-chip master device. The off-chip master device communicates with the on-chip SPI slave controller through the SPI interface with the following specifications:

- Data bit is latched on positive edge of SPI clock.
- Data bit is driven on negative edge of SPI clock.
- Data is transferred from LSB first.
- SPI bus clock is idle at high state.

- Only one byte of data to be transmitted/received in a transaction.
- Slave select signal is high level trigger.

The operation flow is as follows.

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

6.9.7 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_PDMACTL	SPI0_BA + 0x0C	R/W	SPI PDMA Control Register	0x0000_0000
SPI_FIFOCTL	SPI0_BA + 0x10	R/W	FIFO Control/Status Register	0x4400_0000
SPI_STATUS	SPI0_BA + 0x14	R/W	Status Register	0x0005_0110
SPI_RXTSNCNT	SPI0_BA + 0x18	R/W	Receive Transaction Count Register	0x0000_0000
SPI_TX	SPI0_BA + 0x20	W	FIFO Data Transmit Register	0x0000_0000
SPI_RX	SPI0_BA + 0x30	R	FIFO Data Receive Register	0x0000_0000

6.9.8 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							RXMODEEN
23	22	21	20	19	18	17	16
RXTCNTEN	QUADIOEN	DUALIOEN	QDIODIR	REORDER	SLAVE	UNITIEN	TWOBIT
15	14	13	12	11	10	9	8
Reserved		LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**RXMODEEN**	**FIFO Receive Mode Enable** 0 = Disable function. 1 = Enable FIFO receive mode. In this mode SPI transactions will be continuously performed while RXFULL is not active. To stop transactions, set RXMODEEN to 0.
[23]	**RXTCNTEN**	**DMA Receive Transaction Count Enable** 0 = Disable function. 1 = Enable transaction counter for DMA receive only mode. SPI will perform the number of transfers specified in the SPI_RXTSNCNT register, allowing the SPI interface to read ahead of DMA controller.
[22]	**QUADIOEN**	**Quad I/O Mode Enable** 0 =Quad I/O mode Disabled. 1 =Quad I/O mode Enabled.
[21]	**DUALIOEN**	**Dual I/O Mode Enable** 0 = Dual I/O mode Disabled. 1 = Dual I/O mode Enabled.
[20]	**QDIODIR**	**Quad Or Dual I/O Mode Direction Control** 0 =Quad or Dual Input mode. 1 =Quad or Dual Output mode.
[19]	**REORDER**	**Byte Reorder Function Enable** 0 = Byte reorder function Disabled. 1 = Byte reorder function Enabled.A byte suspend interval will be inserted between each byte. The period of the byte suspend interval depends on the setting of SUSPITV. **Note:** Byte reorder function is only available if DWIDTH is defined as 16, 24, and 32 bits. REORDER is only available for Receive mode in DUAL and QUAD transactions.

		For DUAL and QUAD transactions with REORDER, SUSPITV must be set to 0.
[18]	**SLAVE**	**Master Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**UNITIEN**	**Unit Transfer Interrupt Enable** 0 = Disable SPI Unit Transfer Interrupt. 1 = Enable SPI Unit Transfer Interrupt to CPU.
[16]	**TWOBIT**	**Two Bits Transfer Mode** 0 = Disable two-bit transfer mode. 1 = Enable two-bit transfer mode. When 2-bit mode is enabled, the first serial transmitted bit data is from the first FIFO buffer data, and the 2nd serial transmitted bit data is from the second FIFO buffer data. As the same as transmitted function, the first received bit data is stored into the first FIFO buffer and the 2nd received bit data is stored into the second FIFO buffer at the same time.
[15:14]	**Reserved**	Reserved.
[13]	**LSB**	**LSB First** 0 = The MSB is transmitted/received first (which bit in TX and RX FIFO depends on the DWIDTH field). 1 = The LSB is sent first on the line (bit 0 of TX FIFO]), and the first bit received from the line will be put in the LSB position in the SPIn_RX FIFO (bit 0 SPIn_RX). **Note:** For DUAL and QUAD transactions with LSB must be set to 0.
[12:8]	**DWIDTH**	**DWIDTH – Data Word Bit Length** This field specifies how many bits are transmitted in one transmit/receive. Up to 32 bits can be transmitted. DWIDTH = 0x01 … 1 bit DWIDTH = 0x02 … 2 bits …… DWIDTH = 0x1f … 31 bits DWIDTH = 0x00 … 32 bits
[7:4]	**SUSPITV**	**Suspend Interval (Master Only)** The four bits provide configurable suspend interval between two successive transmit/receive transactions in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. SUSPITV is available for standard SPI transactions, it must be set to 0 for DUAL and QUAD mode transactions. (SUSPITV[3:0] + 0.5) * period of SPICLK clock cycle Example: SUSPITV = 0x0 … 0.5 SPICLK clock cycle SUSPITV = 0x1 … 1.5 SPICLK clock cycle …… SUSPITV = 0xE … 14.5 SPICLK clock cycle SUSPITV = 0xF … 15.5 SPICLK clock cycle Note: For DUAL and QUAD transactions with SUSPITV must be set to 0.
[3]	**CLKPOL**	**Clock Polarity** 0 = SCLK idle low. 1 = SCLK idle high.

[2]	**TXNEG**	**Transmit At Negative Edge** 0 = The transmitted data output signal is changed at the rising edge of SCLK. 1 = The transmitted data output signal is changed at the falling edge of SCLK.
[1]	**RXNEG**	**Receive At Negative Edge** 0 = The received data input signal is latched at the rising edge of SCLK. 1 = The received data input signal is latched at the falling edge of SCLK.
[0]	**SPIEN**	**SPI Transfer Enable** 0 = Disable SPI Transfer. 1 = Enable SPI Transfer. In Master mode, the transfer will start when there is data in the FIFO buffer after this is set to 1. In Slave mode, the device is ready to receive data when this bit is set to 1. Note: All configuration should be set before writing 1 to this SPIEN bit. (e.g.: TXNEG, RXNEG, DWIDTH, LSB, CLKP, and so on).

SPI Divider Register (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
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER[7:0]							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DIVIDER**	**Clock Divider Register** The value in this field is the frequency divider for generating the SPI engine clock,f_{spi_sclk},and the SPI serial clock of SPI master. The frequency is obtained according to the following equation. $f_{spi\_eclk} = \frac{f_{spi\_clock\_src}}{(DIVIDER+1)}$ where $f_{spi\_clock\_src}$ is the SPI peripheralclock source, which is defined in the CLK_SEL1 register.

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

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

Bits	Description	
[31:16]	**SLVTOCNT**	**Slave Mode Time-Out Period** In Slave mode, these bits indicate the time out period when there is serial clock input during slave select active. The clock source of the time out counter is Slave engine clock. If the value is 0, it indicates the slave mode time-out function is disabled.
[15:14]	**Reserved**	Reserved.
[13]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable** 0 = Slave select inactive interrupt Disable. 1 = Slave select inactive interrupt Enable.
[12]	**SSACTIEN**	**Slave Select Active Interrupt Enable** 0 = Slave select active interrupt Disable. 1 = Slave select active interrupt Enable.
[11:10]	**Reserved**	Reserved.
[9]	**SLVUDRIEN**	**Slave Mode Error 1 Interrupt Enable** 0 = Slave mode error 1 interrupt Disable. 1 = Slave mode error 1 interrupt Enable.
[8]	**SLVBCEIEN**	**Slave Mode Error 0 Interrupt Enable** 0 = Slave mode error 0 interrupt Disable. 1 = Slave mode error 0 interrupt Enable.
[7]	**Reserved**	Reserved.
[6]	**SLVTORST**	**Slave Mode Time-Out FIFO Clear** 0 = Function disabled. 1 = Both the FIFO clear function, TXRST and RXRST, are activated automatically when there is a slave mode time-out event.
[5]	**SLVTOIEN**	**Slave Mode Time-Out Interrupt Enable** 0 = Slave mode time-outinterrupt Disabled. 1 = Slave mode time-out interrupt Enabled.

[4]	**SLV3WIRE**	**Slave 3-Wire Mode Enable** This is used to ignore the slave select signal in Slave mode. The SPI controller can work with 3-wire interface consisting of SPI_CLK, SPI_MISO, and SPI_MOSI. 0 = 4-wire bi-directional interface. 1 = 3-wire bi-directional interface.
[3]	**AUTOSS**	**Automatic Slave Select Function Enable (Master Only)** 0 = If this bit is cleared, slave select signals will be asserted/de-asserted by setting/clearing the corresponding bits ofSPI_SSCTL[1:0]. 1 = If this bit is set, SPI_SS0/1 signals will be generated automatically. It means that device/slave select signal, which is set in SPI_SSCTL[1:0], will be asserted by the SPI controller when transmit/receive is started, and will be de-asserted after each transmit/receive is finished.
[2]	**SSACTPOL**	**Slave Select Active Level** This bit defines the active status of slave select signal (SPI_SS0/1). 0 = The slave select signal SPI_SS0/1 is active on low-level/falling-edge. 1 = The slave select signal SPI_SS0/1 is active on high-level/rising-edge.
[1:0]	**SS**	**Slave Select Control Bits (Master Only)** If AUTOSS bit is cleared, writing 1 to any bit of this field sets the proper SPISSx0/1 line to an active state and writing 0 sets the line back to inactive state. If the AUTOSS bit is set, writing 0 to any bit location of this field will keep the corresponding SPI_SS0/1 line at inactive state; writing 1 to any bit location of this field will select appropriate SPI_SS0/1 line to be automatically driven to active state for the duration of the transmit/receive, and will be driven to inactive state for the rest of the time. The active state of SPI_SS0/1 is specified in SSACTPOL. Note:SPI_SS0 is defined as the slave select input in Slave mode.

SPI DMA Control Register (SPI_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPI_PDMACTL	SPI0_BA+0x0C	R/W	SPI PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logicof theSPI controller. This bit will be cleared to 0 automatically.
[1]	**RXPDMAEN**	**Receive PDMA Enable** Setting this bit to 1 will start the receive PDMA process. TheSPI controller will issue request to PDMA controller automatically when the SPI receive buffer is not empty. This bit will be clearedto 0 by hardware automatically after PDMA transfer is done.
[0]	**TXPDMAEN**	**Transmit DMA Enable** Setting this bit to 1 will start the transmit PDMA process. SPI controller will issue request to PDMA controller automatically. Hardware will clear this bit to 0 automatically after PDMA transfer done.

SPI FIFO Control Register (SPI_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
SPI_FIFOCTL	SPI0_BA+0x10	R/W	FIFO Control/Status Register	0x4400_0000

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

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TXTH**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TXTH setting, the TXTHIF bit will be set to 1, else the TXTHIF bit will be cleared to 0.
[27]	**Reserved**	Reserved.
[26:24]	**RXTH**	**Receive FIFO Threshold** If the valid data count of the receive FIFO buffer is larger than the RXTH setting, the RXTHIF bit will be set to 1, else the RXTHIF bit will be cleared to 0.
[23:8]	**Reserved**	Reserved.
[7]	**TXUDFIEN**	**Slave Transmit Under Run Interrupt Enable** 0 = Slave Transmit FIFO under-run interrupt Disabled. 1 = Slave Transmit FIFO under-run interrupt Enabled.
[6]	**TXUDFPOL**	**Transmit Under-Run Data Out** 0 = The SPI data out is 0 if there is transmit under-run event in Slave mode. 1 = The SPI data out is 1 if there is transmit under-run event in Slave mode. Note: The under run event is active after the serial clock input and the hardware synchronous, so that the first 1~3 bit (depending on the relation between system clock and the engine clock) data out will be the last transaction data. Note: If the frequency of system clock approach the engine clock, they may be a 3-bit time to report the transmit under-run data out.
[5]	**RXOVIEN**	**Receive FIFO Overrun Interrupt Enable** 0 = Receive FIFO overrun interrupt Disabled. 1 = Receive FIFO overrun interrupt Enabled.
[4]	**RXTOIEN**	**Slave Receive Time-Out Interrupt Enable** 0 = Receive time-out interrupt Disabled. 1 = Receive time-out interrupt Enabled.
[3]	**TXTHIEN**	**Transmit FIFO Threshold Interrupt Enable**

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

SPI Status Register (SPI_STATUS)

Register	Offset	R/W	Description	Reset Value
SPI_STATUS	SPI0_BA+0x14	R/W	Status Register	0x0005_0110

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

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**FIFO CLR Status (Read Only)** 0 = Done the FIFO buffer clear function of TXRST and RXRST. 1 = Doing the FIFO buffer clear function of TXRST or RXRST. Note:Both the TXRST, RXRST, need 3 system clock + 3 engine clocks, the status of this bit allows the user to monitor whether the clear function is busy or done.
[22:20]	**Reserved**	Reserved.
[19]	**TXUFIF**	**Slave Transmit FIFO Under-Run Interrupt Status (Read Only)** When the transmit FIFO buffer is empty and further serial clock pulses occur, data transmitted will be the value of the last transmitted bit and this under-run bit will be set. Note: This bit will be cleared by writing 1 to itself.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count of the transmit FIFO buffer is larger than the setting value of TXTH. 1 = The valid data count of the transmit FIFO buffer is less than or equal to the setting value of TXTH. Note: If TXTHIEN = 1 and TXTHIF = 1, the SPI controller will generate a SPI interrupt request.
[17]	**TXFULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.
[16]	**TXEMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.

[15]	**SPIENSTS**	**SPI Enable Bit Status (Read Only)** 0 = Indicate the transmit control bit is disabled. 1 = Indicate the transfer control bit is active. Note: The clock source of SPI controller logic is engine clock, it is asynchronous with the system clock. In order to make sure the function is disabled in SPI controller logic, this bit indicates the real status of SPIEN in SPI controller logic for user.
[14:13]	**Reserved**	Reserved.
[12]	**RXTOIF**	**Receive Time-Out Interrupt Status** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI clock period in Master mode or over 576 SPI engine clock period in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. Note: This bit will be cleared by writing 1 to itself.
[11]	**RXOVIF**	**Receive FIFO Overrun Status** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. Note: This bit will be cleared by writing 1 to itself.
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Status (Read Only)** 0 =The valid data count within the Rx FIFO buffer is smaller than or equal to the setting value of RXTH. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RXTH. Note: If RXTHIEN = 1 and RXTHIF = 1, the SPI controller will generate a SPI interrupt request.
[9]	**RXFULL**	**Receive FIFO Buffer Full Indicator (Read Only)** 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[8]	**RXEMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[7]	**SLVURIF**	**Slave Mode Error 1 Interrupt Status (Read Only)** In Slave mode, transmit under-run occurs when the slave select line goes to inactive state. 0 = No Slave mode error 1 event. 1 = Slave mode error 1 occurs.
[6]	**SLVBEIF**	**Slave Mode Error 0 Interrupt Status (Read Only)** In Slave mode, there is bit counter mismatch with DWIDTH when the slave select line goes to inactive state. 0 = No Slave mode error 0 event. 1 = Slave mode error 0 occurs. Note: If the slave select active but there is no any serial clock input, the SLVBEIF also active when the slave select goes to inactive state.
[5]	**SLVTOIF**	**Slave Time-Out Interrupt Status (Read Only)** When the Slave Select is active and the value of SLVTOCNT is not 0 and the serial clock input, the slave time-out counter in SPI controller logic will be start. When the value of time-out counter greater or equal than the value of SPI_SSCTL.SLVTOCNT, during before one transaction done, the slave time-out interrupt event will active. 0 = Slave time-out is not active. 1 = Slave time-out is active. Note: If the DWIDTH is set 16, one transaction is equal 16 bits serial clock period.

[4]	**SSLINE**	**Slave Select Line Bus Status (Read Only)** 0 = Indicates the slave select line bus status is 0. 1 = Indicates the slave select line bus status is 1. Note: If SPI_SSCTL.SSACTPOL is set 0, and the SSLINE is 1, the SPI slave select is in inactive status.
[3]	**SSINAIF**	**Slave Select Inactive Interrupt Status** 0 = Slave select inactive interrupt is clear or not occur. 1 = Slave select inactive interrupt event has occur. Note: This bit will be cleared by writing 1 to itself.
[2]	**SSACTIF**	**Slave Select Active Interrupt Status** 0 = Slave select active interrupt is clear or not occur. 1 = Slave select active interrupt event has occur. Note: This bit will be cleared by writing 1 to itself.
[1]	**UNITIF**	**Unit Transfer Interrupt Status** 0 = No transaction has been finished since this bit was cleared to 0. 1 = SPI controller has finished one unit transfer. Note: This bit will be cleared by writing 1 to itself.
[0]	**BUSY**	**SPI Unit Bus Status (Read Only)** 0 = No transaction in the SPI bus. 1 = SPI controller unit is in busy state. The following listing are the bus busy conditions: SPIEN = 1 and the TXEMPTY = 0. For SPI Master, the TXEMPTY = 1 but the current transaction is not finished yet. For SPI Slave receive mode, the SPIEN = 1 and there is serial clock input into the SPI core logic when slave select is active. For SPI Slave transmit mode, the SPIEN = 1 and the transmit buffer is not empty in SPI core logic event if the slave select is inactive.

SPI Receive Transaction Count (SPI_RXTSNCNT)

Register	Offset	R/W	Description	Reset Value
SPI_RXTSNCNT	SPI0_BA+0x18	R/W	Receive Transaction Count Register	0x0000_0000

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

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

SPI Data Transmit Register (SPI_TX)

Register	Offset	R/W	Description	Reset Value
SPI_TX	SPI0_BA+0x20	W	FIFO Data Transmit Register	0x0000_0000

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

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

SPI Data Receive Register (SPI_RX)

Register	Offset	R/W	Description	Reset Value
SPI_RX	SPI0_BA+0x30	R	FIFO Data Receive Register	0x0000_0000

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

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

6.10 Timer Controller (TIMER)

6.10.1 Overview

The ISD9300 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.

6.10.2 Features

- Independent clock source for each channel(TMR0_CLK, TMR1_CLK).
- Time-out period = (Period of timer clock input) * (8-bit prescale counter + 1) * (24-bit CMPDAT)
- Maximum counting cycle time = (1 / T MHz) * (2^8) * (2^{24}), T is the period of timer clock
- 24-bitup counter value is readable through TIMERx_CNT (Timer Data Register)

6.10.3 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 6-53 Timer Controller Block Diagramfor the timer controller block diagram. There are five options of clock source for each channel, Figure 6-54 Clock Source of Timer Controllerillustrate the clock source control function.

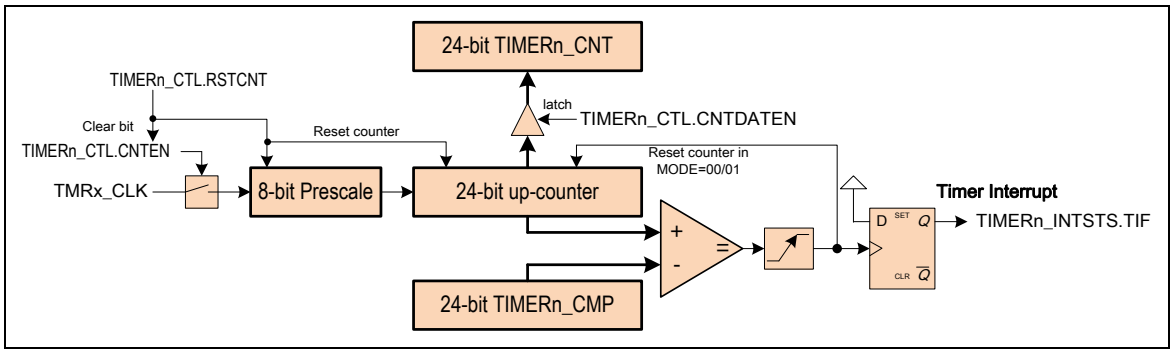


Figure 6-53 Timer Controller Block Diagram

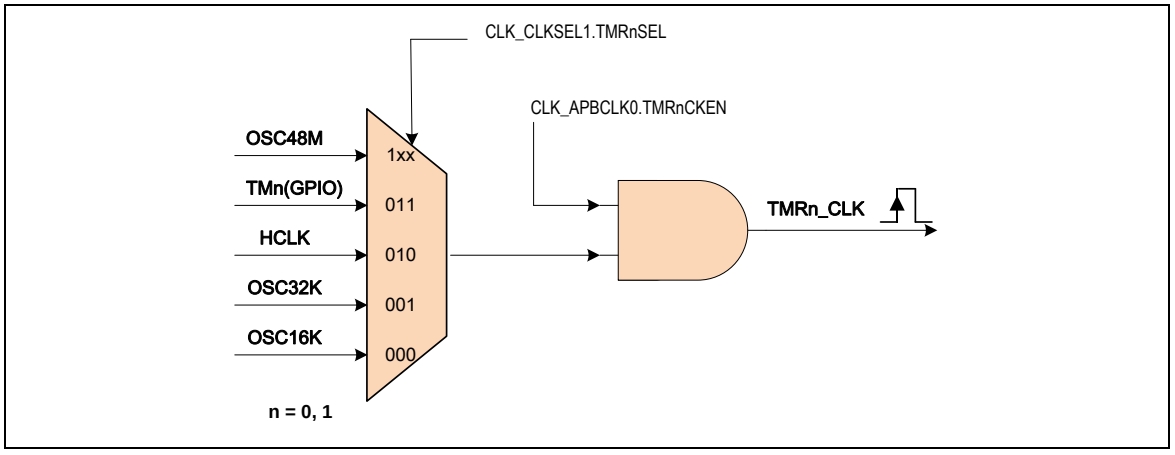


Figure 6-54 Clock Source of Timer Controller

6.10.4 Functional Description

6.10.4.1 Timer Interrupt Flag

Timer controller supports interrupt flags; TIF flag set while timer counter value (CNT) matches the timer compared value (CMPDAT).

6.10.4.2 One–shot Mode

If timer controller is configured at one-shot mode (OPMODE[28:27] is 00) and CNTEN (TIMERx_CTL[30]) bit is set, the timer counter starts up counting. Once the CNT value reaches CMPDAT value, the TIF flag will be set to 1, CNT value and CNTEN bit is cleared by timer controller then timer counting operation stops. In the meantime, if the INTEN (TIMERx_CTL[29]) bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also.

6.10.4.3 Periodic Mode

If timer controller is configured at periodic mode (OPMODE[28:27] is 01) and CNTEN bit is set, the timer counter starts up counting. Once the CNT value reaches CMPDAT value, the TIF flag will be set to 1, CNT value will be cleared by timer controller and timer counter operates counting again. In the meantime, if the IE bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. In this mode, timer controller operates counting and compares with CMPDAT value periodically until the CNTEN bit is cleared by software.

6.10.4.4 Continuous Counting Mode

If timer controller is configured at continuous counting mode (OPMODE[28:27] is 11) and CNTEN bit is set, the timer counter starts up counting. Once the CNT value reaches CMPDAT value, the TIF flag will be set to 1 and CNT value keeps up counting. In the meantime, if the INTEN bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. User can change different CMPDAT value immediately without disabling timer counting and restarting timer counting in this mode.

For example, CMPDAT value is set as 80, first. The TIF flag will set to 1 when CNT value is equal to 80, timer counter is kept counting and CNT value will not goes back to 0, it continues to count 81, 82, 83,⋯ to 2^{24} -1, 0, 1, 2, 3, ⋯ to 2^{24} -1 again and again. Next, if software programs CMPDAT value as 200 and clears TIF flag, the TIF flag will set to 1 again when CNT value reaches to 200. At last, software programs CMPDAT as 500 and clears TIF flag, the TIF flag will set to 1 again when CNT value reaches to 500.

In this mode, the timer counting is continuous. So, this operation mode is called as continuous counting mode.

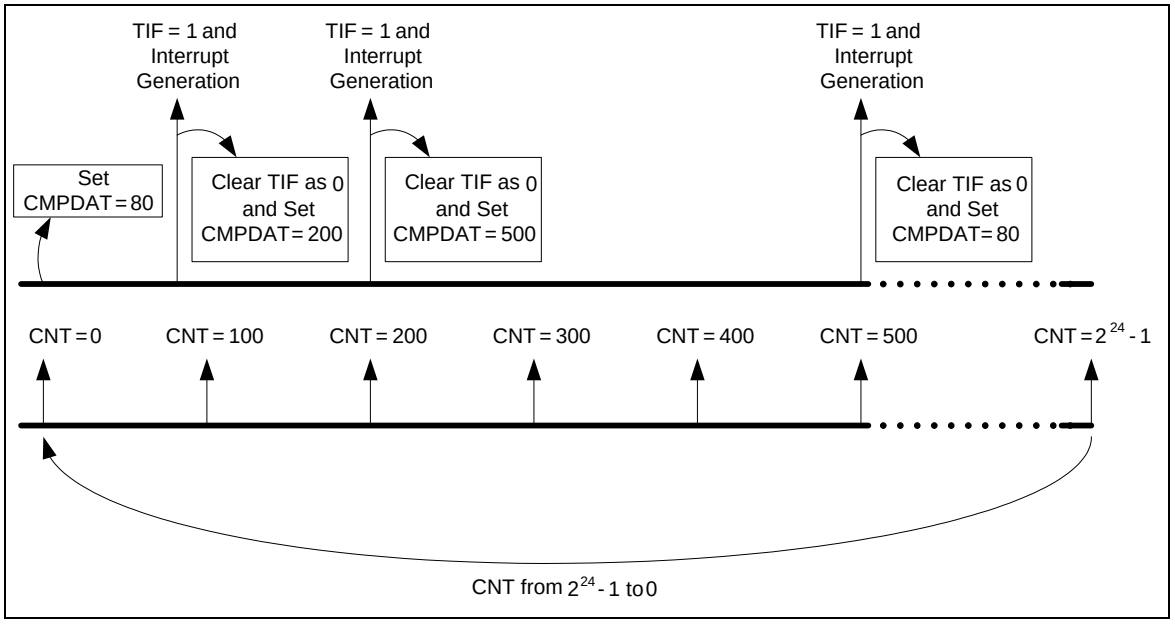


Figure 6-55 Continuous Counting Mode

6.10.5 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

6.10.6 Register Description

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]							

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**CNTEN**	**Counter Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. Note1: Setting CNTEN=1 enables 24-bit counter. It continues count from last value. Note2: This bit is auto-cleared by hardware in one-shot mode (OPMODE =00b) when the timer interrupt is generated (INTEN=1b).
[29]	**INTEN**	**Interrupt Enable Bit** 0 = Disable TIMER Interrupt. 1 = Enable TIMER Interrupt. If timer interrupt is enabled, the timer asserts its interrupt signal when the count is equal to TIMERx_CMP.
[28:27]	**OPMODE**	**Timer Operating Mode** 0 = The timer is operating in the one-shot mode. The associated interrupt signal is generated once (if INTEN isenabled) 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 isenabled). 2 = RESERVED. 3 = The timer is operating in continuous counting mode. The associated interrupt signal is generated when CNT = TIMERx_CMP (if INTEN is enabled); however, the 24-bit up-countercounts continuously without reset.
[26]	**RSTCNT**	**Counter Reset Bit** Set this bit will reset the timer counter, pre-scale and also force CNTEN to 0. 0 = No effect. 1 = Reset Timer's pre-scale counter, internal 24-bit up-counter and CNTEN bit.
[25]	**ACTSTS**	**Timer Active Status Bit (Read Only)** This bit indicates the counter status of timer.

		0 = Timer is not active. 1 = Timer is active.
[24:17]	**Reserved**	Reserved.
[16]	**CNTDATEN**	**Data Latch Enable** When CNTDATEN is set, TIMERx_CNT (Timer Data Register) will be updated continuously with the 24-bit up-counter value as the timer is counting. 1 = Timer Data Register update enable. 0 = Timer Data Register update disable.
[15:8]	**Reserved**	Reserved.
[7:0]	**PSC**	**Pre-Scale Counter** Clock input is divided by PSC+1 before it is fed to the counter. If PSC =0, then there is no scaling.

Timer Compare Register (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]							

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

Timer Interrupt Status Register (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**

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]							

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.

6.11 Watchdog Timer (WDT)

6.11.1 Overview

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.

6.11.2 Features

- 18-bit free running up counter for Watchdog Timer time-out interval.
- Selectable time-out interval (2^4 ~ 2^{18}) WDT_CLK cycle and the time-out interval period is 104 ms ~ 26.3168 s if WDT_CLK = 10 kHz.
- System kept in reset state for a period of (1 / WDT_CLK) * 63
- SupportsWatchdog Timer reset delay period
  - Selectableit includes (1026、130、18 or 3) * WDT_CLK reset delay period.

6.11.3 Block Diagram

The Watchdog Timer clock control and block diagram are shown as follows.

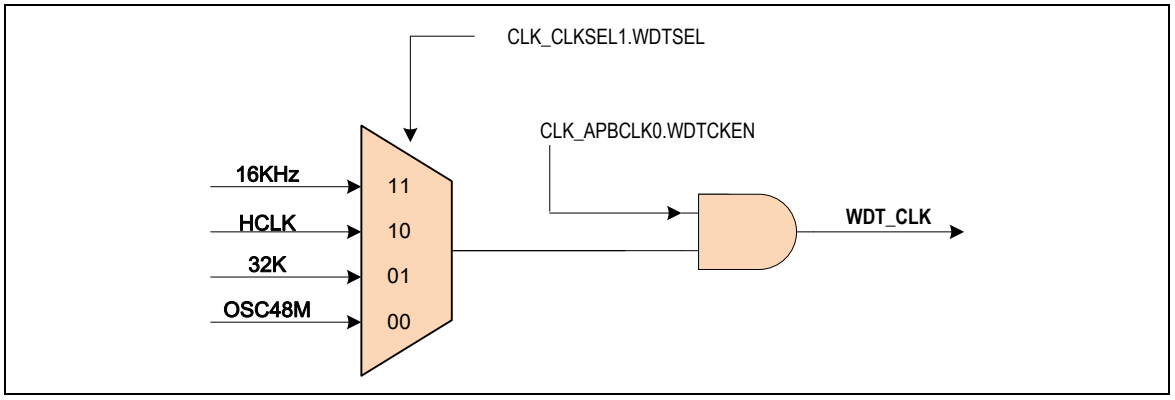


Figure 6-56 Watchdog Timer Clock Control

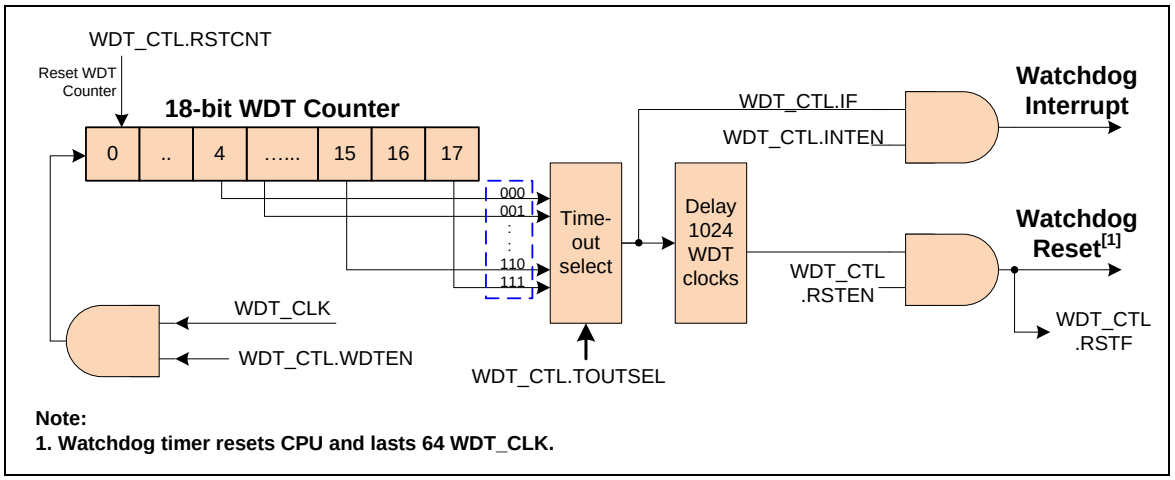


Figure 6-57 Watchdog Timer Block Diagram

6.11.4 Functional Description

The Watchdog Timer (WDT) includes an 18-bit free running up counter with programmable time-out intervals. Table 6-6 Watchdog Timer Time-out Interval Period Selectionand Figure 6-56 Watchdog Timer Clock Controlshows the WDT time-out interval and reset period timing.

- WDT Time-out Interrupt

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.

- WDT Reset Delay Period and Reset System

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.

- WDT Wake-up

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.

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	21.2us	31.7 ms
001	2^6 WDT_CLK	(2^6 + 1024) WDT_CLK	22.1 us	33.2 ms
010	2^8 WDT_CLK	(2^8 + 1024) WDT_CLK	26.0 us	39 ms
011	2^{10} WDT_CLK	(2^{10} + 1024) WDT_CLK	41.7 us	64 ms
100	2^{12} WDT_CLK	(2^{12} + 1024) WDT_CLK	104.2 us	160 ms
101	2^{14} WDT_CLK	(2^{14} + 1024) WDT_CLK	354.2 us	544 ms
110	2^{16} WDT_CLK	(2^{16} + 1024) WDT_CLK	1.4 ms	2080 ms
111	2^{18} WDT_CLK	(2^{18} + 1024) WDT_CLK	5.4 ms	8224 ms

Table 6-6 Watchdog Timer Time-out Interval Period Selection

6.11.5 Register Map

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

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

6.11.6 Register Description

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 =2^(2xTOUTSEL+4) x WDT_CLK Reset Timeout = (2^(2xTOUTSEL+4) +1024) x WDT_CLK Where WDT_CLK is the period of the Watchdog Timer clock source.
[7]	**WDTEN**	**Watchdog Timer Enable** 0 = Disable the Watchdog timer (This action will reset the internal counter). 1 = Enable the Watchdog timer.
[6]	**INTEN**	**Watchdog Timer Interrupt Enable** 0 = Disable the Watchdog timer interrupt. 1 = Enable the Watchdog timer interrupt.
[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

6.12 UART Interface Controller (UART)

6.12.1 Overview

The ISD9300 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).

6.12.2 Features

- Full duplex, asynchronous communications
- Separates receive / transmit 8 bytes entry FIFO for data payloads
- PDMA access support
- Auto flow control function (/CTS, /RTS) supported.
- Programmable baud-rate generator.
- Fully programmable serial-interface characteristics
  - Programmable data bit length, 5-, 6-, 7-, 8-bit character
  - Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
  - Programmable stop bitlength, 1, 1.5, or 2 stop bit generation
  - Baud rate generation
  - False start bit detection
- IrDA SIR function mode
  - Supports 3-/16-bit duration for normal mode
- LIN function mode
  - Supports LIN master mode
  - Supports programmable break generation function for transmitter
  - Supports break detect function for receiver

6.12.3 Block Diagram

The UART clock control and block diagram are shown in Figure 6-57 and Figure 6-58 respectively.

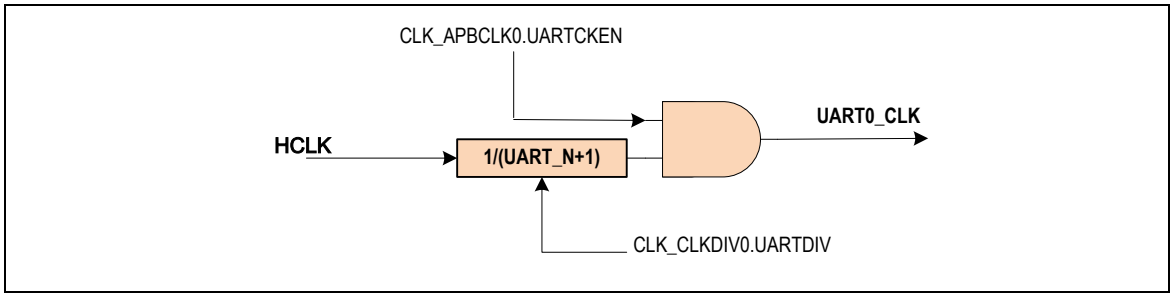


Figure 6-58 UART Clock Control Diagram

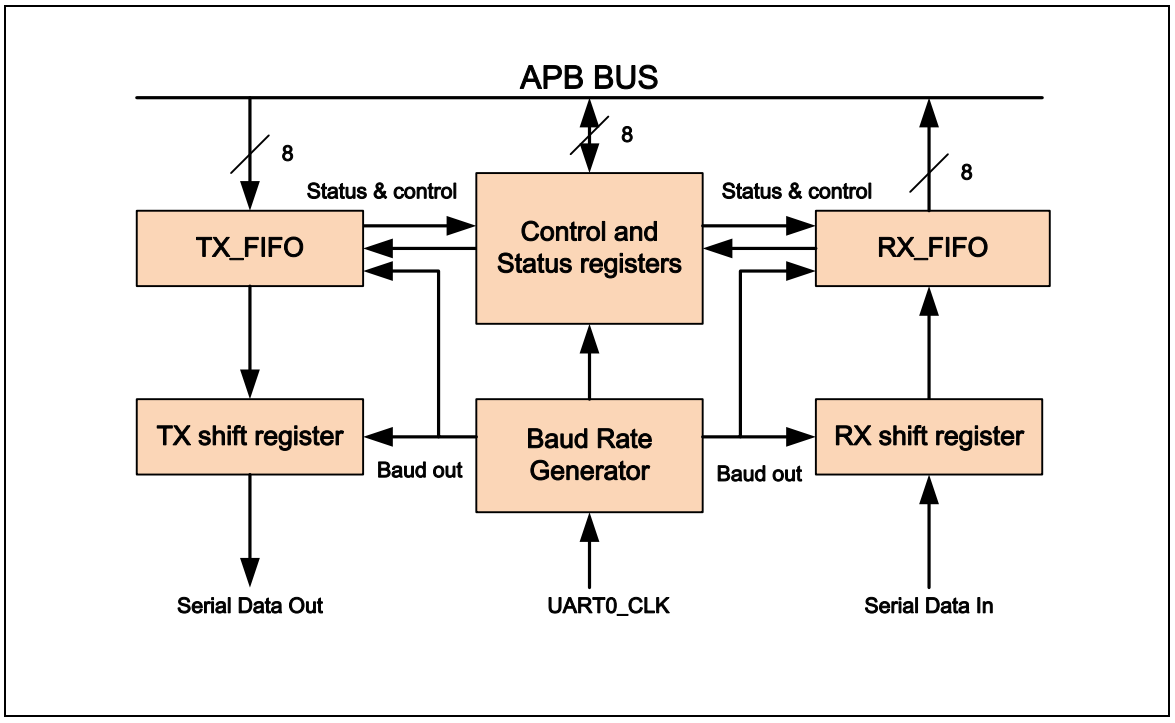


Figure 6-59 UART Block Diagram

Each block is described in detail as follows:

TX_FIFO
The transmitter is buffered with a8 byte FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO
The receiver is buffered with a8 byte FIFO (plus three error bits per byte) to reduce the number of interrupts presented to the CPU.

TX shift Register
This block is the shifting the transmitting data out of serially control.

RX shift Register
This block is the shifting the receiving data in of serially control.

Modem Control Register
This register controls the interface to the MODEM or data set (or a peripheral device emulating a MODEM).

Baud Rate Generator
Divide the external clock by the divisor to get the desired baud rate. Refer to baud rate equation.

IrDA Encode
This block is IrDA encode control block.

IrDA Decode
This block is IrDA decode control block.

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).

6.12.4 Basic Configuration

The UART Controller function pins are configured in SYS_GPA_MFPregister.

The UART Controller clock is enabled in UART_BAUD.

UART Interface Controller Pin description is shown as following:

Pin	Type	Description
UART_TXD	Output	UART transmit
UART_RXD	Input	UART receive
UART_nCTS	Input	UART modem clear to send
UART_nRTS	Output	UART modem request to send

Table 6-7 UART Interface Controller Pin

6.12.5 Functional Description

The UART Controller supports four function modes including UART, IrDA and LIN. User can select a function by setting the UART_FUNCSEL register. The function modes will be described in following section.

6.12.5.1 UART Controller Baud Rate Generator

The UART Controller includes a programmable baud rate generator capable of dividing clock input by divisors to produce the serial clock that transmitter and receiver need. The baud rate equation is Baud Rate = UART_CLK / M * [BRD + 2], where M and BRD are defined in Baud Rate Divider Register (UART_BAUD). The following tables list the UART baud rate equations in the various conditionsand UART baud rate parameter settings. There is no error for the baud rate results calculated through the baud rate parameter and register setting below. In IrDA function mode, the baud rate generator must be set in Mode 0.

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

Table 6-8 UART Baud Rate Equation

6.12.5.2 UART Controller FIFO Control and Status

The UART0 is built-in with a8-byte transmitter FIFO (TX_FIFO) and a 8-byte receiver 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 operation. The reported status information includes the type and condition of the transfer operations being performed by the UART, as well as 3 error conditions (parity error, framing error, break interrupt) probably occur while receiving data. This FIFO control and status also support all of UART, IrDA and LIN function mode.

6.12.5.3 UART Controller Interrupt and Status

Each UART Controller supports six types of interrupts including:

- Transmitter FIFO empty interrupt (THREINT)
- Line status interrupt (parity error, frame error or break interrupt) (RLSINT)
- MODEM/Wake-up status interrupt (MODEMINT)
- Receiver buffer time-out interrupt (TOUTINT)
- Buffer error interrupt (BUFERRINT)
- LIN bus interrupt (LININT)

The following tables describe the interrupt sources and flags. The interrupt is generated when the interrupt flag is generated and the interrupt enable bit is set. User must clear the interrupt flag after the interrupt is generated.

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator to Interrupt Controller	Interrupt Flag	Flag Cleared by
LIN RX Break Field Detected interrupt	LINIEN	DLININT	DLINIF	Write ‘1’ to LINIF
Buffer Error Interrupt BUFERRINT	BUFERRIEN	DBERRINT	DBERRIF = (TXOVIF or RXOVIF)	Write ‘1’ to TXOVIF/ RXOVIF
Rx Timeout Interrupt RXTOINT	RXTOIEN	DRXTOINT	DRXTOIF	Read data FIFO
Modem Status Interrupt MODEMINT	MODEMIEN	DMODEMI	DMODEMIF = (CTSDETF)	Write ‘1’ to CTSDETF

Table 6-9 UART Controller Interrupt Source and Flag List 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	LININT	LINIF	Write ‘1’ to LINIF
Buffer Error Interrupt BUF_ERR_INT	BUFERRIEN	BUFERRINT	BUFERRIF = (TXOVIF or RXOVIF)	Write ‘1’ to TXOVIF/ RXOVIF
Rx Timeout Interrupt RXTOINT	RXTOIEN	RXTOINT	RXTOIF	Read data FIFO
Modem Status Interrupt	MODEMIEN	MODEMINT	MODENIF =	Write ‘1’ to CTSDETF

MODEMINT			(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

Table 6-10 Controller Interrupt Source and Flag in Software ModeList

6.12.5.4 UART Function Mode

UART Auto-Flow Control Function

The UART supports auto-flow control function that uses two 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 is not allowed to receive data until the UART asserts RTS to external device. When the number of bytes in the RX FIFO the RTS is de-asserted. The UART sends data out when UART detects CTS is asserted from external device. If the valid asserted CTS is not detected, the UART will not send data out.

The following diagram demonstrates the auto-flow control block.

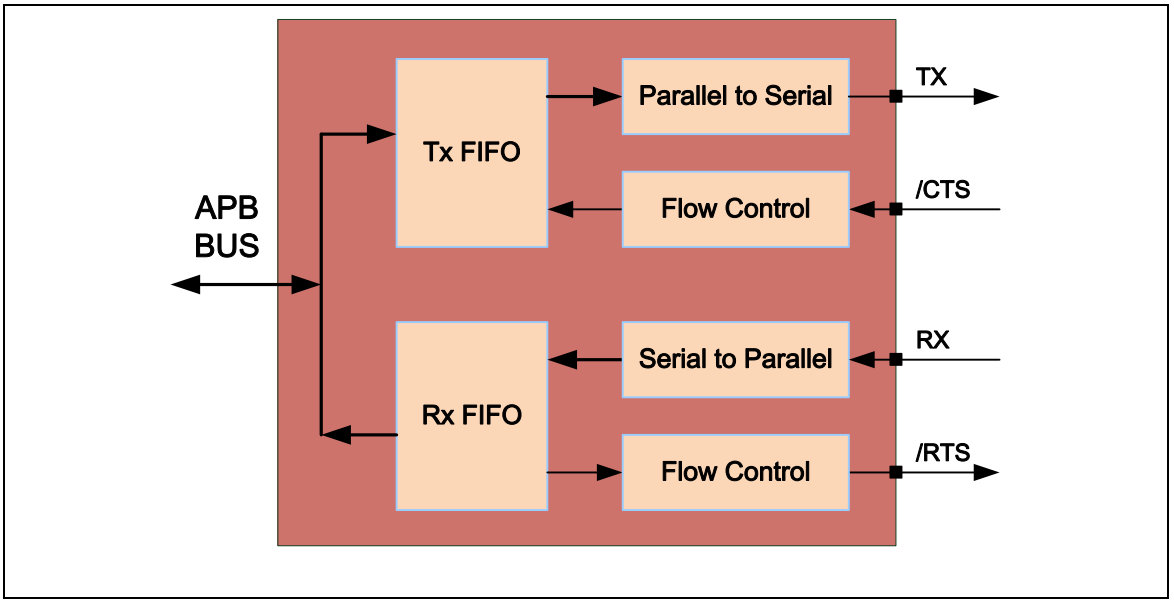


Figure 6-60 Auto Flow Control Block Diagram

The following diagram demonstrates the CTS auto flow control of UART function mode. User must set ATOCTSEN (UART_INTEN [13]) to enable CTS auto flow control function. The TX data will be automatically transmitted from TX FIFO.

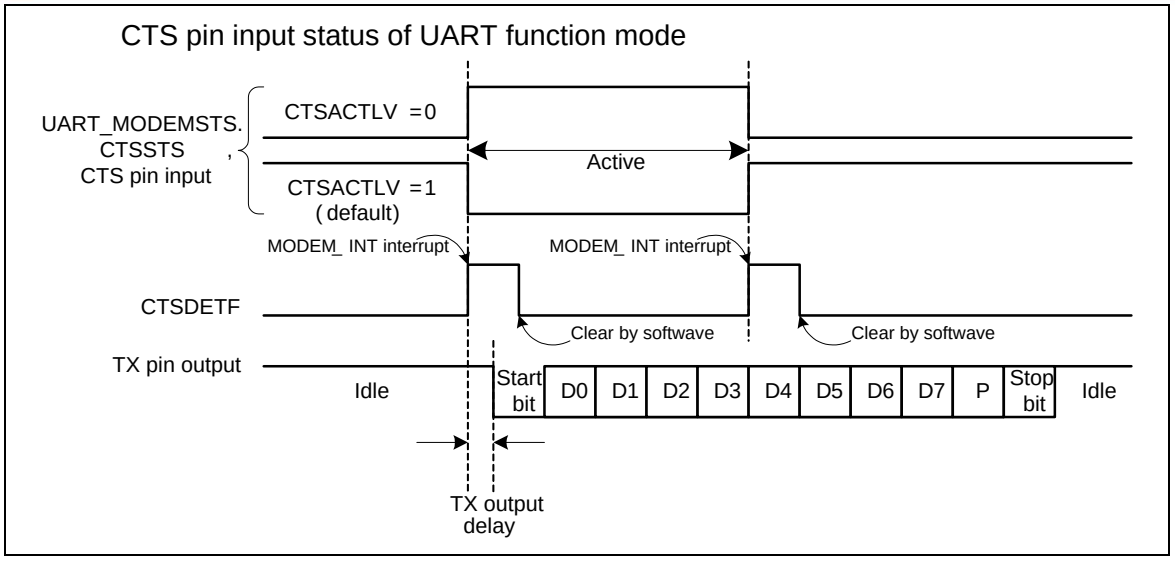


Figure 6-61 UART CTS Auto Flow Control Enabled

As shown in the following figure, in UART RTS Auto Flow control mode (AUTORTSEN(UART_INTEN[12])=1), the RTS internal signal is controlled by UART FIFO controller.

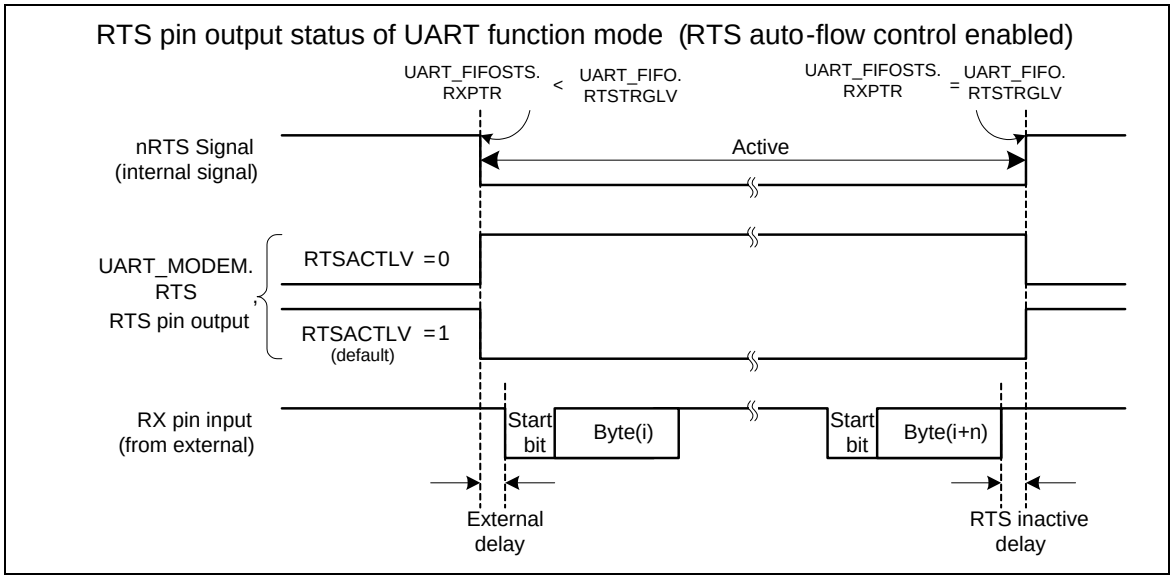


Figure 6-62 UART RTS Auto Flow Control Enabled

As shown in the following figure, in software mode (AUTORTSEN(UART_INTEN[12])=0) the RTS flow is directly controlled by software programming of RTS control bit.

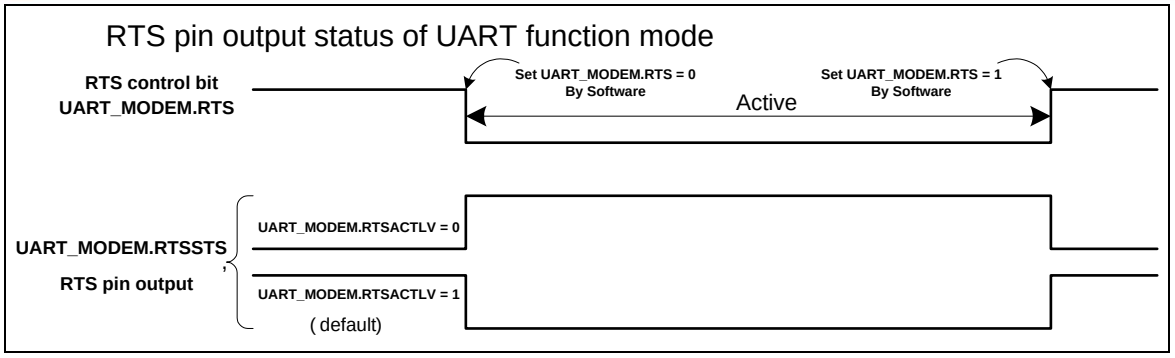


Figure 6-63 UART RTS Flow with Software Control

6.12.5.5IrDA Function Mode

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

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

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

The following diagram demonstrates the IrDA control block diagram.

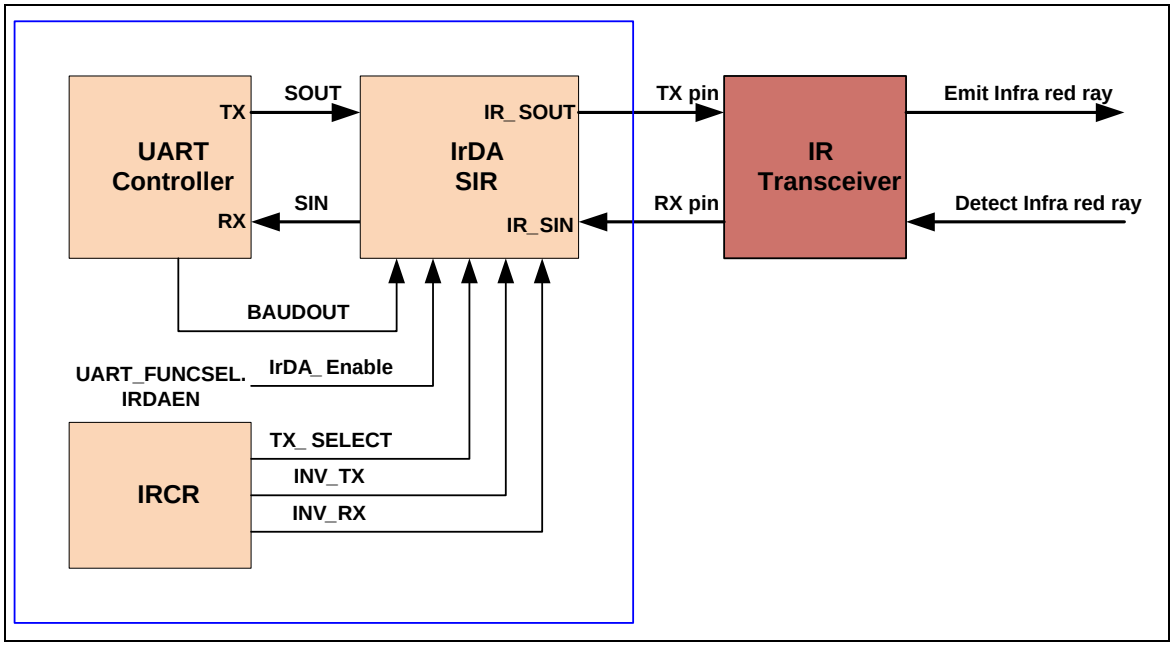


Figure 6-64 IrDA Control Block Diagram

IrDA SIR Transmit Encoder

The IrDA SIR Transmit Encoder modulates Non-Return-to Zero (NRZ) transmit bit stream output from UART. The IrDA SIR physical layer specifies the use of Return-to-Zero, Inverted (RZI) modulation scheme which represents logic 0 as an infra light pulse. The modulated output pulse stream is transmitted to an external output driver and infrared light emitting diode.

In Normal mode, the transmitted pulse width is specified as 3/16 period of baud rate.

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 decoder input is LOW.

IrDA SIR Operation

The IrDA SIR encoder/decoder provides functionality which converts between UART data stream and half-duplex serial SIR interface. The following diagram is IrDA encoder/decoder waveform.

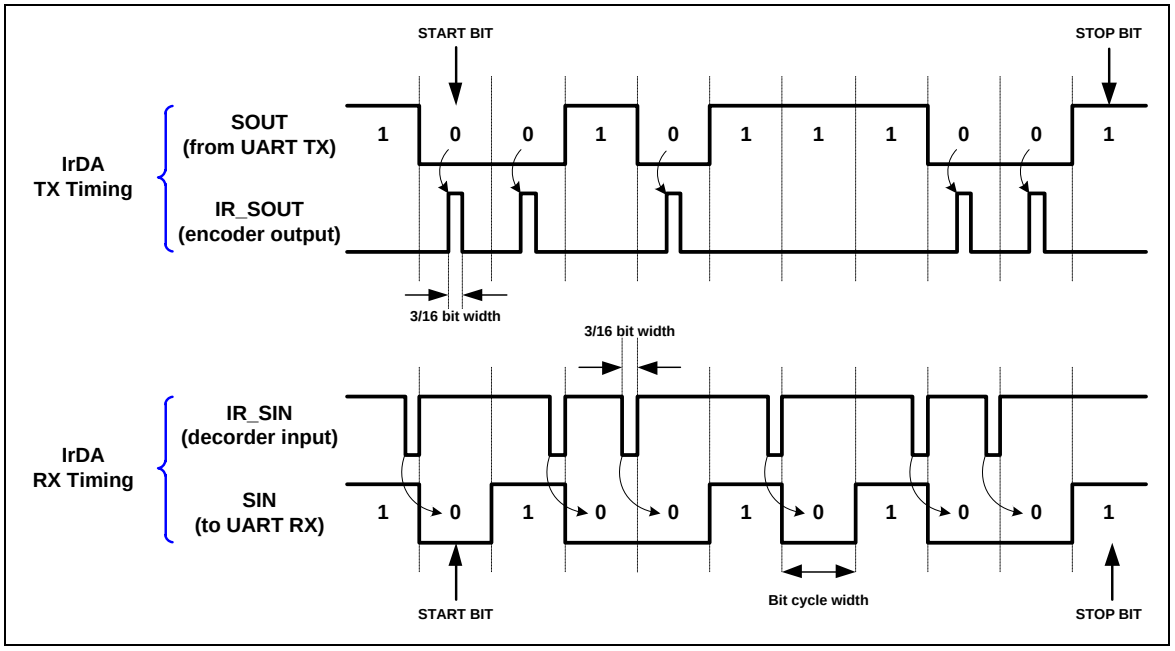


Figure 6-65 IrDA TX/RX Timing Diagram

6.12.5.6LIN (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/).

6.12.5.6.1 Structure of LIN Frame

According to the LIN protocol, all information transmitted is packed as frames; a frame consists of a header (provided by the master task)and a response (provided by a slave task), followed by a response(provided by a slave task). The header (provided by the master task) consists of a break field and a sync field followed by a frame identifier (frame ID). The frame identifier uniquely defines the purpose of the frame. The slave task is appointed for providing the response associated with the frame ID. The response consists of a data field and a checksum field. The following diagram is the structure of LIN Frame.

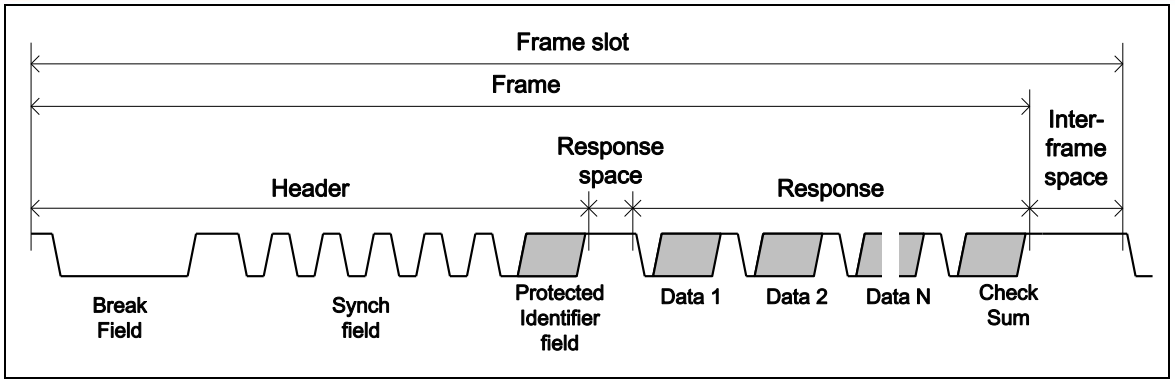


Figure 6-66 Structure of LIN Frame

6.12.5.6.2 Structure of LIN Byte

In LIN mode, each byte field is initiated by a START bit with value 0 (dominant), followed by 8 data bits and no parity bit, LSB is first and ended by 1 stop bit with value 1 (recessive) in accordance with the LIN standard. The structure of Byte is shown as follows.

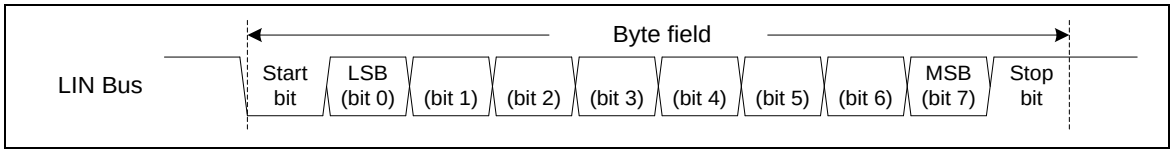


Figure 6-67 Structure of LIN Byte

6.12.5.6.3 LIN Master Mode

The UART0 controller supports LIN Master mode. To enable and initialize the LIN Master mode, the following steps are necessary:

1. Setting the UART_IRDA register to select the desired baud rate.
3. Setting FUNSEL (UART_FUNSEL[1]) to “1” to select LIN function mode operation.

The program flow of LIN Bus Transmittransfer (Tx) is shown as following.

The procedure without software error monitoring in Master mode:

1. Set the UART_FUNCSEL.LINEN bit to enable LIN Bus mode.
2. SetUART_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 toUART_DAT
5. Set the UART_ALTCTL.LINTXEN bit to start transmission (When break filed operation is finished, LINTX_EN will be cleared automatically).
6. When the STOP bit of the last byte UART_DAT has been sent to bus, hardware will set flag UART_FIFOSTS.TXEMPTYF to 1.
7. Fill N bytes data and Checksum to UART_DAT then repeat step 5 and 6 to transmit the data.

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

Procedure with software error monitoring in Master mode:

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

6.12.6 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

6.12.7 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							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DAT**	**Receive FIFO Register** Reading this register will return data from the receive data FIFO. By reading this register, the UART will return the 8-bit data received from Rx pin (LSB first).

Interrupt Enable Register (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

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 = EnableLin 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**

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 transmit internal state machine and pointers. Note: This bit will auto-clear after 3 UART engine clock cycles.
[1]	**RXRST**	**Receive FIFO Reset** When RXRST is set, all the bytes in the receive FIFO are cleared and receive internal state machine is reset. 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset the receiving internal state machine and pointers. Note: This bit will auto-clear after 3 UART engine clock cycles.

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

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[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

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

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CTSACTLV**	**Clear-To-Send (CTS) Active Trigger Level** This bit can change the CTS trigger level. 0= CTS is active low level. 1= CTS is active high level.
[7:5]	**Reserved**	Reserved.
[4]	**CTSSTS**	**CTS Pin Status (Read Only)** This bit is the pin status of CTS.
[3:1]	**Reserved**	Reserved.
[0]	**CTSDETF**	**Detect CTS State Change Flag** This bit is set whenever CTS input has state change. It will generate Modem interrupt to CPU when UART_INTEN.MODEMIEN=1 NOTE: This bit is cleared by writing 1 to itself.

FIFO Status Register (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

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	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THERINT	RDAINT
7	6	5	4	3	2	1	0
LINIF	Reserved	BUFERRIF	RXTOIF	MODENIF	RLSIF	THREIF	RDAIF

Bits	Description	
[31]	**DLININT**	**DMA MODE LIN Bus Rx Break Field Detected Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DLINIF.
[30]	**Reserved**	Reserved.
[29]	**DBERRINT**	**DMA MODE Buffer Error Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DBERRIF.
[28]	**DRXTOINT**	**DMA MODE Time Out Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DRXTOIF.
[27]	**DMODEMI**	**DMA MODE MODEM Status Interrupt Indicator To Interrupt** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DMODENIF.
[26]	**DRLSINT**	**DMA MODE Receive Line Status Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.DMARXEN or UART_INTEN.DMATXEN and DRLSIF.
[25]	**Reserved**	Reserved.
[24]	**Reserved**	Reserved.
[23]	**DLINIF**	**DMA MODE LIN Bus Rx Break Field Detected Flag** This bit is set when LIN controller detects a break field. This bit is cleared by writing a 1.
[22]	**Reserved**	Reserved.
[21]	**DBERRIF**	**DMA MODE Buffer Error Interrupt Flag (Read Only)** This bit is set when either the Tx or Rx FIFO overflows (UART_FIFOSTS.TXOVIF or UART_FIFOSTS.RXOVIF is set). When BUFERRIF is set, the serial transfer may be corrupted. If UART_INTEN.BUFERRIEN is enabled a CPU interrupt request will be generated. NOTE: This bit is cleared when both UART_FIFOSTS.TXOVIF and UART_FIFOSTS.RXOVIF are cleared.
[20]	**DRXTOIF**	**DMA MODE Time Out Interrupt Flag (Read Only)** This bit is set when the Rx FIFO is not empty and no activity occurs in the Rx FIFO and the time out counter equal to TOIC. If UART_INTEN.TOUT_IEN is enabled a CPU

		interrupt request will be generated. NOTE: This bit is read only and user can read FIFO to clear it.
[19]	**DMODEMIF**	**DMA MODE MODEM Interrupt Flag (Read Only)** This bit is set when the CTS pin has changed state (UART_MODEMSTS.CTSDETF=1). If UART_INTEN.MODEMIEN is enabled, a CPU interrupt request will be generated. NOTE: This bit is read only and reset when bit UART_MODEMSTS.CTSDETF is cleared by a write 1.
[18]	**DRLSIF**	**DMA MODE Receive Line Status Interrupt Flag (Read Only)** This bit is set when the Rx receive data has a parity, framing or break error (at least one of, UART_FIFOSTS.BIF, UART_FIFOSTS.FEF and UART_FIFOSTS.PEF, is set). If UART_INTEN.RLSIEN is enabled, the RLS interrupt will be generated. NOTE: This bit is read only and reset to 0 when all bits of BIF, FEF and PEF are cleared.
[17:16]	**Reserved**	Reserved.
[15]	**LININT**	**LIN Bus Rx Break Field Detected Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.LINIEN and LINIF.
[14]	**Reserved**	Reserved.
[13]	**BUFERRINT**	**Buffer Error Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.BUFERRIEN and BUFERRIF.
[12]	**RXTOINT**	**Time Out Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.RXTOIEN and RXTOIF.
[11]	**MODEMINT**	**MODEM Status Interrupt Indicator To Interrupt** Logical AND of UART_INTEN.MODEMIEN and MODENIF.
[10]	**RLSINT**	**Receive Line Status Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.RLSIEN and RLSIF.
[9]	**THERINT**	**Transmit Holding Register Empty Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.THREIEN and THREIF.
[8]	**RDAINT**	**Receive Data Available Interrupt Indicator To Interrupt Controller** Logical AND of UART_INTEN.RDAIEN and RDAIF.
[7]	**LINIF**	**LIN Bus Rx Break Field Detected Flag** This bit is set when LIN controller detects a break field. This bit is cleared by writing a 1.
[6]	**Reserved**	Reserved.
[5]	**BUFERRIF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when either the Tx or Rx FIFO overflows (UART_FIFOSTS.TXOVIF or UART_FIFOSTS.RXOVIF is set). When BUFERRIF is set, the serial transfer may be corrupted. If UART_INTEN.BUFERRIEN is enabled a CPU interrupt request will be generated. NOTE: This bit is cleared when both UART_FIFOSTS.TXOVIF and UART_FIFOSTS.RXOVIF are cleared.
[4]	**RXTOIF**	**Time Out Interrupt Flag (Read Only)** This bit is set when the Rx FIFO is not empty and no activity occurs in the Rx FIFO and the time out counter equal to TOIC. If UART_INTEN.TOUT_IEN is enabled a CPU interrupt request will be generated. NOTE: This bit is read only and user can read FIFO to clear it.
[3]	**MODENIF**	**MODEM Interrupt Flag (Read Only)** This bit is set when the CTS pin has changed state (UART_MODEMSTS.CTSDETF=1). If UART_INTEN.MODEMIEN is enabled, a CPU interrupt request will be generated. NOTE: This bit is read only and reset when bit UART_MODEMSTS.CTSDETF is cleared

		by a write 1.
[2]	**RLSIF**	**Receive Line Status Interrupt Flag (Read Only)** This bit is set when the Rx receive data has a parity, framing or break error (at least one of, UART_FIFOSTS.BIF, UART_FIFOSTS.FEF and UART_FIFOSTS.PEF, is set). If UART_INTEN.RLSIEN is enabled, the RLS interrupt will be generated. NOTE: This bit is read only and reset to 0 when all bits of BIF, FEF and PEF are cleared.
[1]	**THREIF**	**Transmit Holding Register Empty Interrupt Flag (Read Only)** This bit is set when the last data of Tx FIFO is transferred to Transmitter Shift Register. If UART_INTEN.THREIEN is enabled, the THRE interrupt will be generated. NOTE: This bit is read only and it will be cleared when writing data into the Tx FIFO.
[0]	**RDAIF**	**Receive Data Available Interrupt Flag (Read Only)** When the number of bytes in the Rx FIFO equals UART_FIFO.RFITL then the RDAIF will be set. If UART_INTEN.RDAIEN is enabled, the RDA interrupt will be generated. NOTE: This bit is read only and it will be cleared when the number of unread bytes of Rx FIFO drops below the threshold level (RFITL).

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**						

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:0]	**TOIC**	**Time Out Interrupt Comparator** The time out counter resets and starts counting whenever the Rx FIFO receives a new data word. Once the content of time out counter is equal to that of time out interrupt comparator (TOIC), a receiver time out interrupt (RXTOINT) is generated if UART_INTEN.RXTOIEN is set. A new incoming data word or RX FIFO empty clears RXTOIF. The period of the time out counter is the baud rate.

Baud Rate Divider Register (UART_BAUD)

Register	Offset	R/W	Description	Reset Value
UART_BAUD	UART0_BA + 0x24	R/W	UART0 Baud Rate Divisor Register	0x0F00_0000

31	30	29	28	27	26	25	24
Reserved		BAUDM1	BAUDM0	EDIVM1			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BRD							
7	6	5	4	3	2	1	0
BRD							

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**BAUDM1**	**Divider X Enable** The baud rate equation is: Baud Rate =UART_CLK / [M * (BRD + 2)] ; The default value of M is 16. 0 = Disable divider X (M=16). 1 = Enable divider X (M =EDIVM1+1, with EDIVM1 ≥ 8). Refer to Table 6-8 UART Baud Rate Equationfor more information. NOTE: When in IrDA mode, this bit must disabled.
[28]	**BAUDM0**	**Divider X Equal 1** 0: M =EDIVM1+1, with restriction EDIVM1 ≥ 8. 1: M =1, with restriction BRD[15:0] ≥ 3. Refer to Table 6-8 UART Baud Rate Equationfor more information.
[27:24]	**EDIVM1**	**Divider X** The baud rate divider M =EDIVM1+1.
[23:16]	**Reserved**	Reserved.
[15:0]	**BRD**	**Baud Rate Divider** Refer to Table 6-8 UART Baud Rate Equationfor more information.

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

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

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

UART LIN Network Control Register (UART_ALTCTL)

Register	Offset	R/W	Description	Reset Value
UART_ALTCTL	UART0_BA + 0x2C	R/W	UART0 LINControlRegister.	0x0000_0000

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

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

UART Function Select Register (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

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.

6.13 I²S Controller (I²S)Audio PCM Controller

6.13.1 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.

6.13.2 Features

- SupportsMaster mode andSlave mode
- Master clock generation for slave device synchronization
- Capable of handling 16, 24 and 32 bit word sizes
- Mono and stereo audio data supported
- Supports I²S and MSB justified data format
- Provides two 8-word FIFO data buffers, one for transmitting and the other for receiving
- Generates interrupt requests when buffer levels crosses programmable boundary
- Two DMA requests, one for transmit and one for receive

6.13.3 Block Diagram

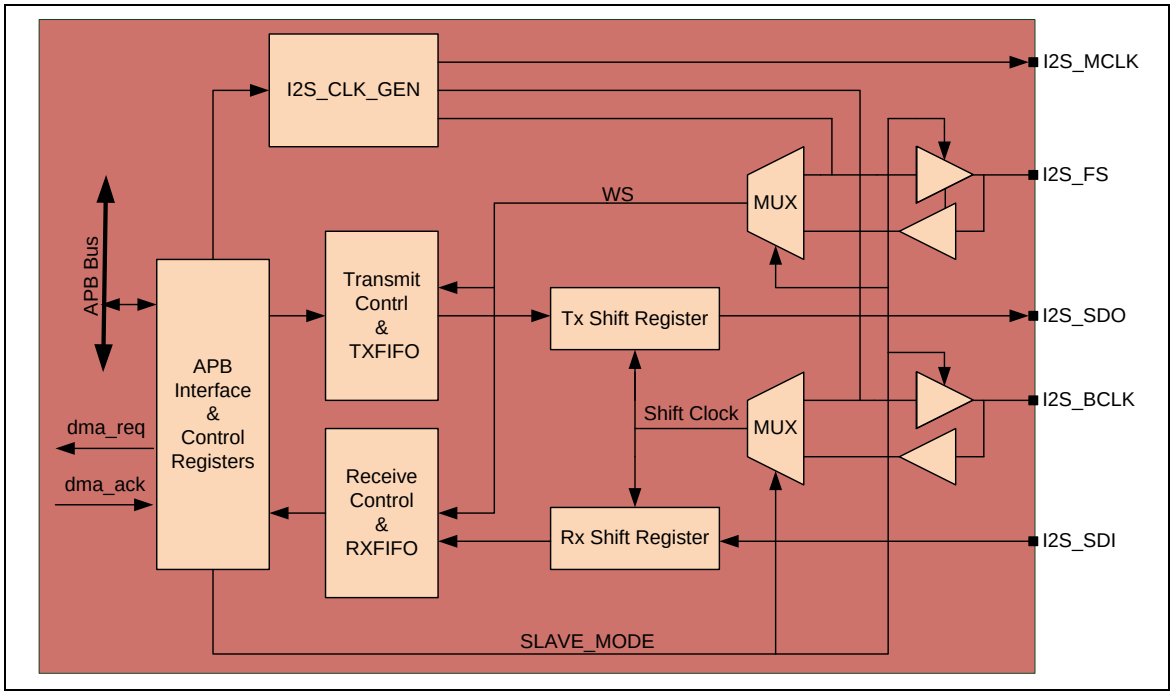


Figure 6-68 I²S Controller Block Diagram

6.13.4 Functional Description

6.13.4.1 I²S Clock

The I²S controller has four clock sources selected by I2S_S (CLKSEL2[1:0]). The I²S clock rate must be slower than or equal to system clock rate.

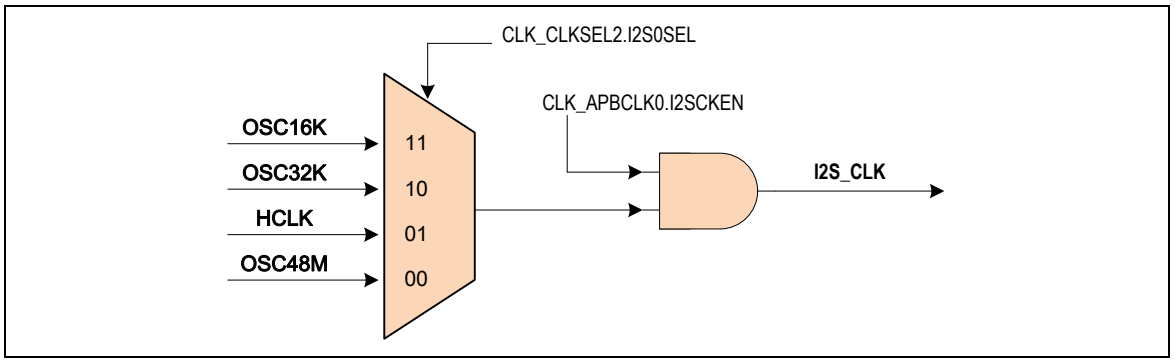


Figure 6-69 I²S Clock Control Diagram

6.13.4.2 I²S Operation

The I²S controller supports MSB justified and I²S data format. The I2SLRCLK signal indicates which audio channel is in transferring. The bit count of an audio channel is determined by WDWIDTH (I2S_CTL[5:4]). The transfer sequence is always first from the most significance bit, MSB. Data are read on rising clock edge and are driven on falling clock edge.

In I²S data format, the MSB is sent and latched on the second clock of an audio channel.

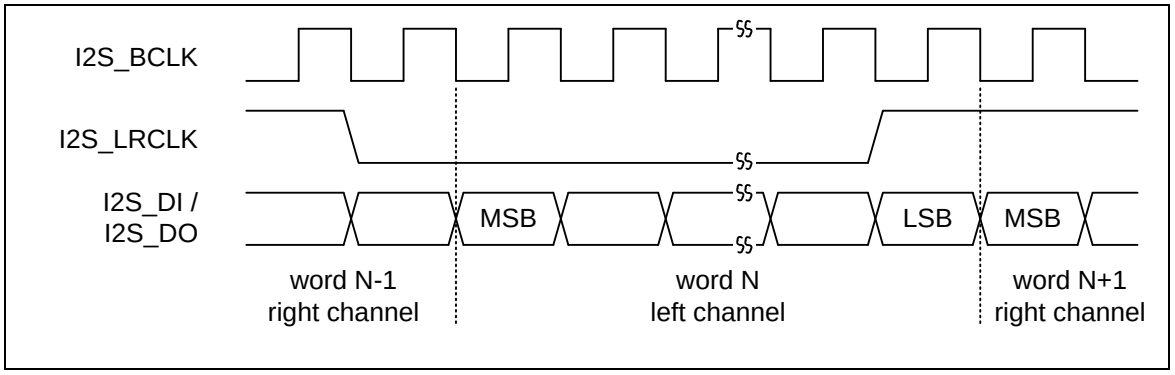


Figure 6-70 I²S Data Format Timing Diagram

In MSB justified data format, the MSB is sent and latched on the first clock of an audio channel.

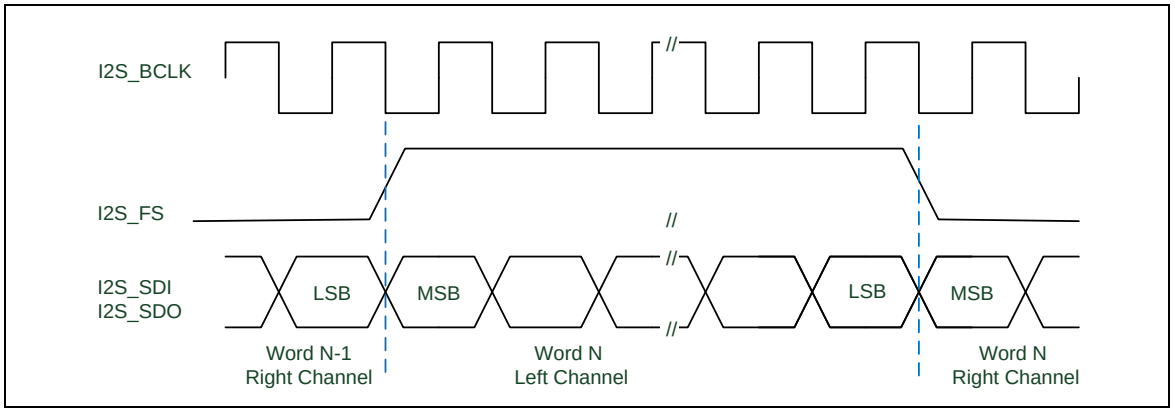


Figure 6-71 MSB Justified Data Format Timing Diagram

6.13.4.3 I^2S Interrupt Sources

The I^2S controller supports left channel zero-cross interrupt, right channel zero-cross interrupt, transmit FIFO threshold level interrupt, transmit FIFO overflow interrupt and transmit FIFO underflow interrupt in transmit operation. In receive operation, it supports receive FIFO threshold level interrupt, receive FIFO overflow interrupt and receive FIFO underflow interrupt. When I^2S interrupt occurs, user can check I2S_STATUS flags to recognize the interrupt sources.

6.13.4.4 FIFO Operation

The word width of an audio channel can be 8, 16, 24 or 32 bits. The memory arrangements for various settings are shown below.

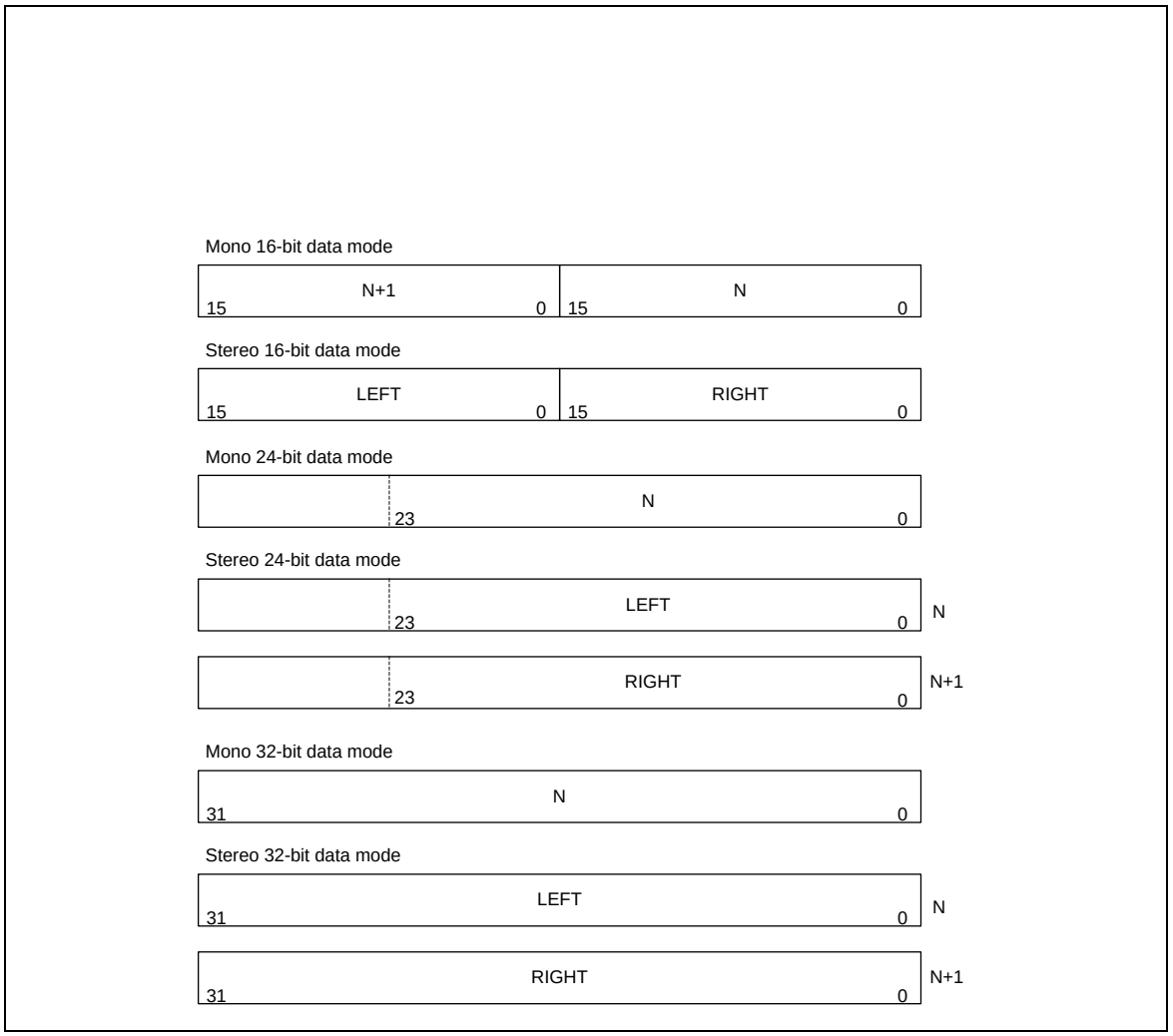


Figure 6-72 FIFO Contents for Various I^{2}S Modes

6.13.4.5Zero Cross Detection

When playing audio by I²S function, the output data comes from the memory by PDMA or by CPU. However, it may result some pop noise if the playing gain level is changed by user at any time. Because, the output data is not zero, and the output data cross the gain change will generate a sharp pop noise. Therefore, the zero-cross flag will help to reduce this situation. If enable the zero cross detection function, hardware will detect the next transfer data is zero or sign change. If the next data is zero or sign change, zero-cross flag will be set to high, and the output data will be mute automatically, until the flag is cleared by software.

6.13.4.6PDMA Mode

The I²S function can use PDMA function to access the data. In transmit mode,when PDMA function is enabled, if TX FIFO is not full, the I²S will generate the request signal and get a data from memory by PDMA automatically, until the TX FIFO is full. However, in receive mode,when PDMA function is enabled, if the RX FIFO is not empty, the I²S will generate the request signal and move a receive data to memory by PDMA automatically, until the RX FIFO is empty. User can enable PDMA function to ease CPU's' loading.

6.13.4.7Master/Slave Interface

If I²S controller is configured as Master mode, it drives I2S_MCLK, I2S_BCLK and I2S_LRCLK for a device slave, such as an audio CODEC.

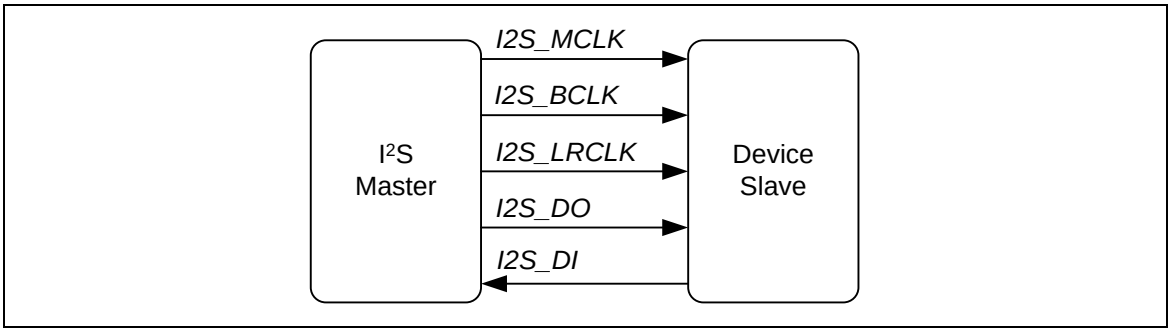


Figure 6-73 Master Mode Interface

If I²S controller is configured as Slave mode, I2S_MCLK, I2S_BCLK and I2S_LRCLK are driven by a device master, such as an audio CODEC.

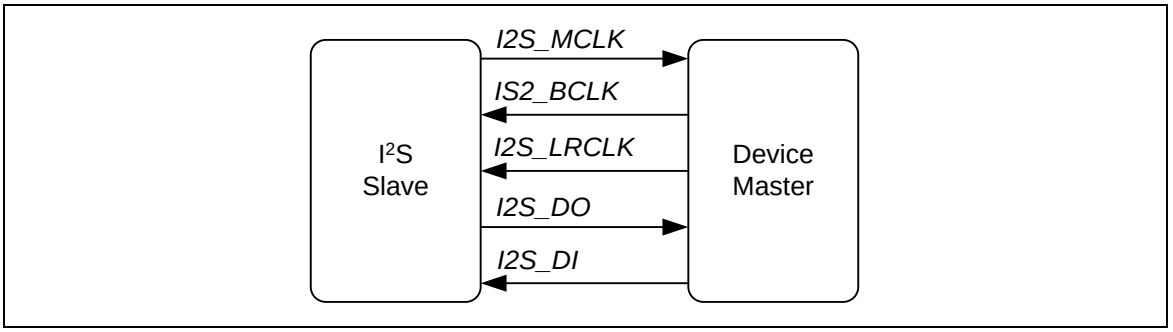


Figure 6-74 Slave Mode Interface

6.13.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
I²SBase 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

6.13.6 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		RXPDMAEN	TXPDMAEN	RXCLR	TXCLR	LZCEN	RZCEN
15	14	13	12	11	10	9	8
MCLKEN	RXTH			TXTH			SLAVE
7	6	5	4	3	2	1	0
FORMAT	MONO	WDWIDTH		MUTE	RXEN	TXEN	I2SEN

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

[16]	**RZCEN**	**Right Channel Zero Cross Detect Enable** If this bit is set to 1, when right channel data sign bit changes, or data bits are all zero, the RZCIF flag in I2S_STATUS register will be set to 1. 0 = Disable right channel zero cross detect. 1 = Enable right channel zero cross detect.
[15]	**MCLKEN**	**Master Clock Enable** The ISD9300 can generate a master clock signal to an external audio CODEC to synchronize the audio devices. If audio devices are not synchronous, then data will be periodically corrupted. Software needs to implement a way to drop/repeat or interpolate samples in a jitter buffer if devices are not synchronized. The master clock frequency is determined by the I2S_CLKDIV.MCLKDIV[2:0] register. 0 = Disablemasterclock. 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 ISD9300. 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 **Error! Reference source not found.** and **Error! Reference source not found.** for timing differences.
[6]	**MONO**	**Monaural Data** This parameter sets whether mono or stereo data is processed. See **Error! Reference source not found.** 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 **Error! Reference source not found.** 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 Register (I2S_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2S_CLKDIV	I2S_BA + 0x04	R/W	I2S Clock Divider Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**BCLKDIV**	**Bit Clock Divider** If I2S operates in master mode, bit clock is provided by ISD9300. Software can program these bits to generate bit clock frequency for the desired sample rate. For sample rate Fs, the desired bit clock frequency is: F(BCLK) =Fs x Word_width_in_bytes x 16 For example if Fs=16kHz, and word width is 2-bytes (16bit) then desired bit clock frequency is 512kHz. The bit clock frequency is given by: F(BCLK) =F(I2S_CLKDIV) / 2x(BCLKDIV+1) Or, BCLKDIV =F(I2S_CLKDIV) / (2 x F(BCLK)) -1 So if F(I2S_CLKDIV) =HCLK =49.152MHz,desired F(BCLK)=512kHzthen BCLKDIV =47
[7:3]	**Reserved**	Reserved.
[2:0]	**MCLKDIV**	**Master Clock Divider** ISD9300 can generate a master clock to synchronously drive an external audio device. If MCLKDIV is set to 0, MCLK is the same as I2S_CLKDIV clock input, otherwise MCLK frequency is given by: F(MCLK) =F(I2S_CLKDIV) / (2xMCLKDIV) Or, MCLKDIV =F(I2S_CLKDIV) / (2 x F(MCLK)) If the desired MCLK frequency is 254Fs and Fs=16kHz then MCLKDIV =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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**LZCIEN**	**RZCIEN**	**TXTHIEN**	**TXOVIEN**	**TXUDIEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**RXTHIEN**	**RXOVIEN**	**RXUDIEN**

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

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

I²S Status Register (I2S_STATUS)

Register	Offset	R/W	Description	Reset Value
I2S_STATUS	I2S_BA + 0x0C	R/W	I2S Status Register	0x0014_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

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 Register (I2S_TX)

Register	Offset	R/W	Description	Reset Value
I2S_TX	I2S_BA + 0x10	W	I2S Transmit FIFO Register	0x0000_0000

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

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 Register (I2S_RX)

Register	Offset	R/W	Description	Reset Value
I2S_RX	I2S_BA + 0x14	R	I2S Receive FIFO Register	0x0000_0000

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

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.

6.14 PDMA Controller (PDMA)

6.14.1 Overview

The ISD9300 series DMA contains nine-channel peripheral direct memory access (PDMA) controller and a cyclic redundancy check (CRC) generator.

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

The DMA controller contains a cyclic redundancy check (CRC) generator that can perform CRC calculation with programmable polynomial settings.

6.14.2 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, wrap-around and spanned addressing
- Cyclic Redundancy Check (CRC)
  - Supports four common polynomials CRC-CCITT, CRC-8, CRC-16, and CRC-32
    - CRC-CCITT: $X^{16} + X^{12} + X^5 + 1$
    - CRC-8: $X^8 + X^2 + X + 1$
    - CRC-16: $X^{16} + X^{15} + X^2 + 1$
    - CRC-32: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
  - Supports programmable CRC seed value.
  - Supports programmable order reverse setting for input data and CRC checksum.
  - Supports programmable 1's complement setting for input data and CRC checksum.
  - Supports CPU PIO mode or DMA transfer mode.
  - Supports the follows write data length in CPU PIO mode
    - 8-bit write mode (byte): 1-AHB clock cycle operation.
    - 16-bit write mode (half-word): 2-AHB clock cycle operation.
    - 32-bit write mode (word): 4-AHB clock cycle operation.
  - Supports byte alignment transfer data length and word alignment transfer source address in CRC DMA mode.

6.14.3 Block Diagram

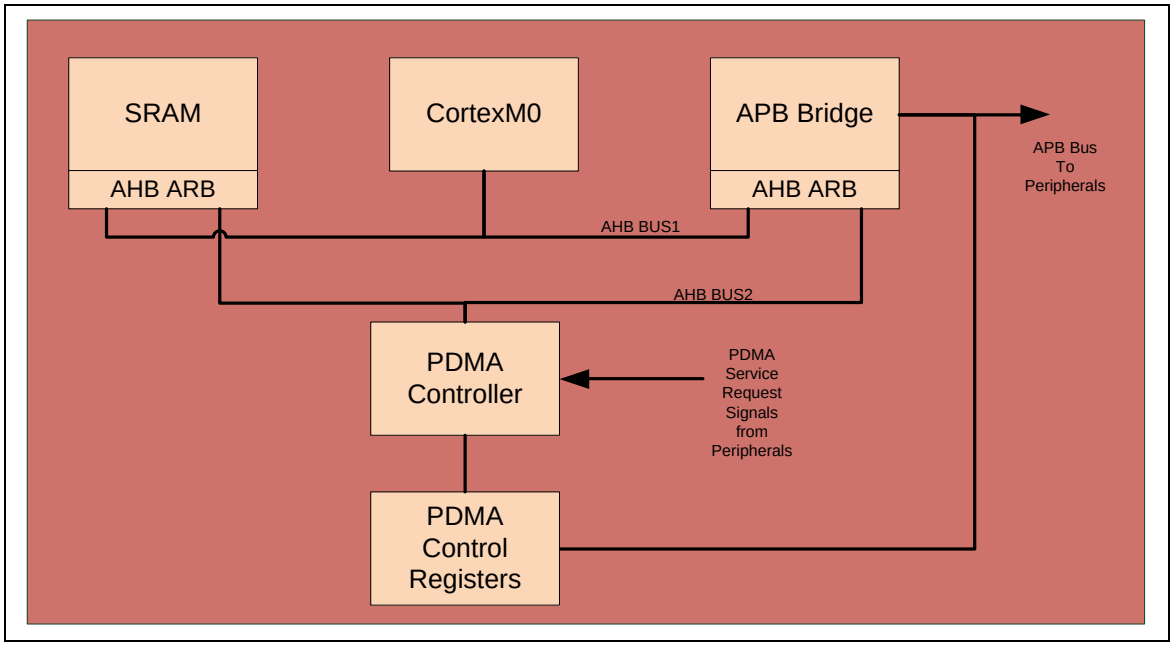


Figure 6-75 DMA Controller Block Diagram

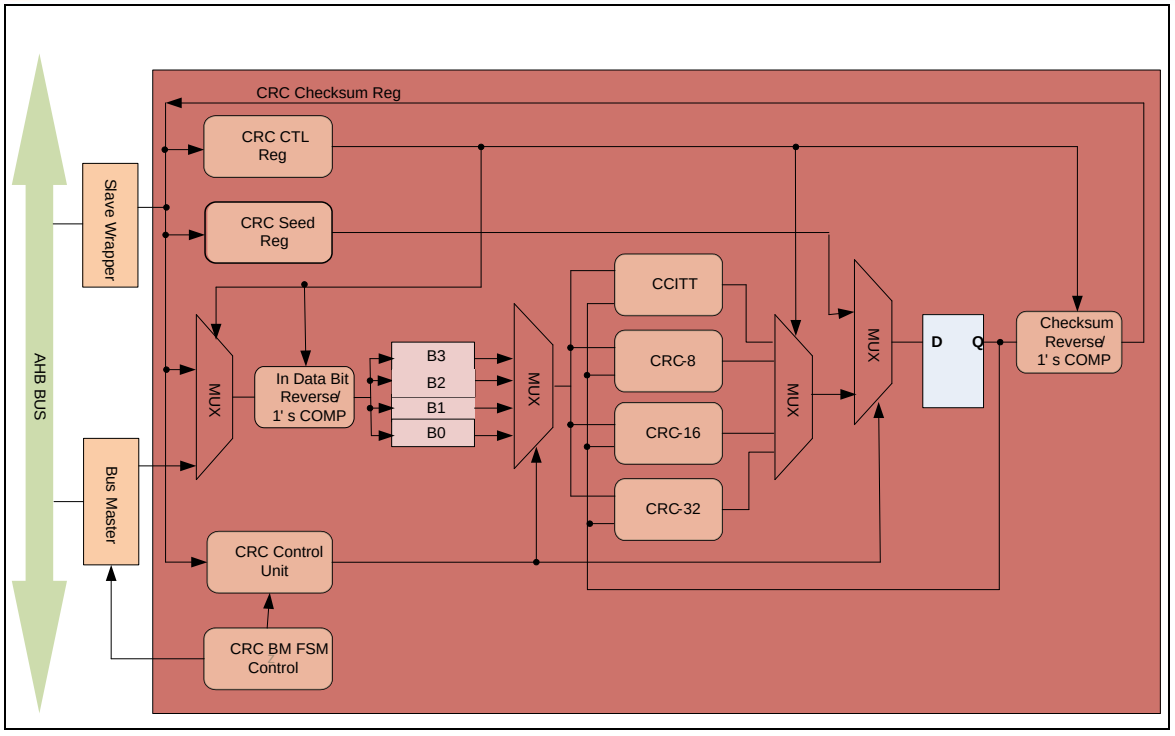


Figure 6-76 CRC Generator Block Diagram

6.14.4 Functional Description

The direct memory access (DMA) controller module transfers data from one address to another address, without CPU intervention. The DMA controller contains nine PDMA (Peripheral-to-Memory or Memory-to-Peripheral or Memory-to-Memory) channels and one CRC generator channel.

The CPU can recognize the completion of a DMA operation by software polling or when it receives an internal DMA interrupt.

6.14.4.1 PDMA

The DMA controller has four channels PDMA (Peripheral-to-Memory or Memory-to-Peripheral or Memory-to-Memory). As to the source and destination address, the PDMA controller has two modes: increased and fixed.

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 following sequence is a program sequence example.

- Enable PDMA peripheral clock by setting PDMA_HCLK*n*_EN bit.
- Configure the channel service setting by setting PDMA_DSCTn_CTL.CHEN register.
- Configure source/destination address by setting PDMA_DSCTn_ENDSA /PDMA_DSCTn_ENDDA registers.
- Configure PDMA_transfer byte count by setting PDMA_TXBCCHn register.
- 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 BYTECNT, 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_DSCT0_CTL.SWRST bit to reset the PDMA channel. After reset the CHEN and TRGEN bits would need to be set to start a new operation.

6.14.4.2 CRC

The DMA controller contains a cyclic redundancy check (CRC) generator that can perform CRC calculations with programmable polynomial settings. Available operation polynomials are CRC-CCITT, CRC-8, CRC-16 and CRC-32; Software can choose the operation polynomial mode by setting CRCMODE fields in PDMA_CRCCTL register.

The CRC engine support CPU PIO mode (PDMA_CRCCTL [CRCEN] = 1,

PDMA_CRCCTL[TRGEN] = 0) and DMA transfer mode (PDMA_CRCCTL [CRCEN] = 1, PDMA_CRCCTL [TRGEN] = 1).

Procedure when operating in CPU PIO mode:

- Enable CRC engine by setting CRCEN bit in PDMA_CRCCTL register.
- Initial Setting. Set the data format (DATREV, CHKSREV, DATFMT and CHKSFMT by setting PDMA_CRCCTL register), initial seed value (PDMA_CRCSEED) and select the data length by setting PDMA_CRCCTL [DATLEN] register.
- Set CRCRST to load the initial seed value to CRC circuit.
- Write data to PDMA_CRCDAT to perform CRC calculation.
- Then, get the CRC checksum results by reading the CRC checksum register (PDMA_CRCCHKS).

Procedure when operating in CRC DMA mode:

- Enable CRC engine by setting CRCEN bit in PDMA_CRCCTL register.
- Initial Setting. Set the data format (DATREV, CHKSREV, DATFMT and CHKSFMT by setting PDMA_CRCCTL register), initial seed value (PDMA_CRCSEED).
- Give a valid source address and transfer count by setting PDMA_CRCSA and PDMA_CRCBC.
- Enable PDMA_CRCCTL [TRGEN] and then hardware will reset the seed value and then read memory data to perform CRC calculation.
- Wait CRC DMA transfer and CRC calculation done and then retrieve the CRC checksum result by reading PDMA_CRCCHKS register.

6.14.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMA_BA = 0x5000_8000**				
PDMA_DSCT0_CTL	PDMA_BA+0x00	R/W	PDMA ControlRegister 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_SPANRCH0	PDMA_BA+0x34	R	PDMA Span Increment Register of Channel 0	0x0000_0000
PDMA_CURSRCH0	PDMA_BA+0x38	R/W	PDMA Current Span Increment Register of Channel 0	0x0000_0000
PDMA_DSCT1_CTL	PDMA_BA+0x100	R/W	PDMA ControlRegister 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_SPANRCH1	PDMA_BA+0x134	R	PDMA Span Increment Register of Channel 1	0x0000_0000
PDMA_CURSRCH1	PDMA_BA+0x138	R/W	PDMA Current Span Increment Register of Channel 1	0x0000_0000
PDMA_DSCT2_CTL	PDMA_BA+0x200	R/W	PDMA ControlRegister of Channel 2	0x0000_0000

PDMA_DSCT2_ENDSA	PDMA_BA+0x204	R/W	PDMA Transfer Source Address Register of Channel 2	0x0000_0000
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_SPANRCH2	PDMA_BA+0x234	R	PDMA Span Increment Register of Channel 2	0x0000_0000
PDMA_CURSRCH2	PDMA_BA+0x238	R/W	PDMA Current Span Increment Register of Channel 2	0x0000_0000
PDMA_DSCT3_CTL	PDMA_BA+0x300	R/W	PDMA ControlRegister 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_SPANRCH3	PDMA_BA+0x334	R	PDMA Span Increment Register of Channel 3	0x0000_0000
PDMA_CURSRCH3	PDMA_BA+0x338	R/W	PDMA Current Span Increment Register of Channel 3	0x0000_0000
PDMA_CRCCTL	PDMA_BA+0xE00	R/W	CRC Control Register	0x2000_0000
PDMA_CRCSA	PDMA_BA+0xE04	R/W	CRC DMA Source Address Register	0x0000_0000
PDMA_CRCBC	PDMA_BA+0xE0C	R/W	CRC DMA Transfer Byte Count Register	0x0000_0000
PDMA_CRCCSA	PDMA_BA+0xE14	R	CRC DMA Current Source Address Register	0x0000_0000
PDMA_CRCCBC	PDMA_BA+0xE1C	R	CRC DMA Current Transfer Byte Count Register	0x0000_0000
PDMA_CRCINTEN	PDMA_BA+0xE20	R/W	CRC DMA Interrupt Enable Register	0x0000_0001

PDMA_CRCINTF	PDMA_BA+0xE24	R/W	CRC DMA Interrupt Status Register	0x0000_0000
PDMA_CRCDAT	PDMA_BA+0xE80	R/W	CRC Write Data Register	0x0000_0000
PDMA_CRCSEED	PDMA_BA+0xE84	R/W	CRC Seed Register	0xFFFF_FFFF
PDMA_CRCCHKS	PDMA_BA+0xE88	R	CRC Checksum Register	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

6.14.6 Register Description

PDMA ControTXENl 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 Registerof 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		TXWIDTH		Reserved		
15	14	13	12	11	10	9	8
WAINTSEL				Reserved			
7	6	5	4	3	2	1	0
DASEL		SASEL		MODESEL		SWRST	CHEN

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**TXEN**	**Trigger Enable – Start A PDMA Operation** 0 = Write: no effect. Read: Idle/Finished. 1 = Enable PDMA data read or write transfer. Note:When PDMA transfer completed, this bit will be cleared automatically. If a bus error occurs, all PDMA transfer will be stopped. Software must reset PDMA channel, and then trigger again.
[22:21]	**Reserved**	Reserved.
[20:19]	**TXWIDTH**	**Peripheral Transfer Width Select** This parameter determines the data width to be transferred each PDMA transfer operation. 00 = One word (32 bits) is transferred for every PDMA operation. 01 = One byte (8 bits) is transferred for every PDMA operation. 10 = One half-word (16 bits) is transferred for every PDMA operation. 11 = RESERVED. Note: This field is meaningful only when MODESEL is IP to Memory mode (APB-to-Memory) or Memory to IP mode (Memory-to-APB).
[18:16]	**Reserved**	Reserved.

[15:12]	**WAINTSEL**	**Wrap Interrupt Select** x1xx: If this bit is set, and wraparound mode is in operation a Wrap Interrupt can be generated when half each PDMA transfer is complete. For example if BYTECNT=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=32 then an interrupt could be generated when 32 bytes were sent and PDMA wraps around. x1x1: Both half and w interrupts generated.
[11:8]	**Reserved**	Reserved.
[7:6]	**DASEL**	**Destination Address Select** This parameter determines the behavior of the current destination address register with each PDMA transfer. It can either be fixed, incremented or wrapped. 00 = Transfer Destination Address is incremented. 01 = RESERVED. 10 = Transfer Destination Address is fixed (Used when data transferred from multiple addresses to a single destination such as peripheral FIFO input). 11 = Transfer Destination Address is wrapped. When PDMA_CURBCCH (Current Byte Count) equals zero, the PDMA_CURDACH (Current Destination Address) and PDMA_CURBCCH 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 PDMA_EN=0. When PDMA_EN is disabled, the PDMA will complete the active transfer but the remaining data in the SBUF will not be transferred to the destination address.
[5:4]	**SASEL**	**Source Address Select** This parameter determines the behavior of the current source address register with each PDMA transfer. It can either be fixed, incremented or wrapped. 00 = Transfer Source address is incremented. 01 = RESERVED. 10 = Transfer Source address is fixed. 11 = Transfer Source address is wrapped. When PDMA_CURBCCH (Current Byte Count) equals zero, the PDMA_CURSACH (Current Source Address) and PDMA_CURBCCH registers will be reloaded from the SAR (Source Address) and PDMA_TXBCCH (Byte Count) registers automatically and PDMA will start another transfer. Cycle continues until software sets PDMA_EN=0. When PDMA_EN is disabled, the PDMA will complete the active transfer but the remaining data in the SBUF will not be transferred to the destination address.
[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

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

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

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

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]							

Bits	Description	
[31:16]	**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_TXBCCHn.MODESEL=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				BURPTR			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**BURPTR**	**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

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

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 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

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

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

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

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 BYTECNT 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**

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

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 =Current transfer finished flag (CURBC==0). 0100 = Current transfer half complete flag (CURBC==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 Span Increment Register (PDMA_SPANRCHn) (n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_SPANRCH0	PDMA_BA+0x34	R	PDMA Span Increment Register of Channel 0	0x0000_0000
PDMA_SPANRCH1	PDMA_BA+0x134	R	PDMA Span Increment Register of Channel 1	0x0000_0000
PDMA_SPANRCH2	PDMA_BA+0x234	R	PDMA Span Increment Register of Channel 2	0x0000_0000
PDMA_SPANRCH3	PDMA_BA+0x334	R	PDMA Span Increment 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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPANREG							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**SPANREG**	**Span Increment Register** This is a signed number in range [-128,127] for use in spanned address mode. If destination or source addressing mode is set as spanned, then this number is added to the address register each transfer. The size of the transfer is determined by the APB_TW setting. Note that span increment must be a multiple of the transfer width otherwise a memory addressing HardFault will occur. Also SPANREG may be a negative number.

PDMA Current Span Increment Register (PDMA_CURSRCHn) (n=0~3)

Register	Offset	R/W	Description	Reset Value
PDMA_CURSRCH0	PDMA_BA+0x38	R/W	PDMA Current Span Increment Register of Channel 0	0x0000_0000
PDMA_CURSRCH1	PDMA_BA+0x138	R/W	PDMA Current Span Increment Register of Channel 1	0x0000_0000
PDMA_CURSRCH2	PDMA_BA+0x238	R/W	PDMA Current Span Increment Register of Channel 2	0x0000_0000
PDMA_CURSRCH3	PDMA_BA+0x338	R/W	PDMA Current Span Increment 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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CSPANREG							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[15:0]	**CSPANREG**	**Current Span Increment Register** This is a signed read only register for use in spanned address mode. It provides the current address offset from ENDSA or ENDDA if either is set to span mode.

CRC Control Register (PDMA_CRCCTL)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCCTL	PDMA_BA+0xE00	R/W	CRC Control Register	0x2000_0000

31	30	29	28	27	26	25	24
CRCMODE		DATLEN		CHKSFMT	DATFMT	CHKSREV	DATREV
23	22	21	20	19	18	17	16
TRGEN	Reserved						
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CRCRST	CRCEN

Bits	Description	
[31:30]	**CRCMODE**	**CRC Polynomial Mode** 00 = CRC-CCITT Polynomial mode. 01 = CRC-8 Polynomial mode. 10 = CRC-16 Polynomial mode. 11 = CRC-32 Polynomial mode.
[29:28]	**DATLEN**	**CPU Write Data Length** When operation in CPU PIO mode (CRCEN = 1, TRGEN = 0), this field indicates the write data length. 00 = Data length is 8-bit mode 01 = Data length is 16-bit mode 1x = Data length is 32-bit mode Note1: This field is used for CPU PIO mode. Note2: When the data length is 8-bit mode, the valid data is PDMA_CRCDAT [7:0]; if the data length is 16-bit mode, the valid data is PDMA_CRCDAT [15:0].
[27]	**CHKSFMT**	**Checksum Complement** 0 = No 1's complement for CRC checksum. 1 = 1's complement for CRC checksum.
[26]	**DATFMT**	**Write Data Complement** 0 = No 1's complement for CRC write data in. 1 = 1's complement for CRC write data in.
[25]	**CHKSREV**	**Checksum Reverse** 0 = No bit order reverse for CRC checksum. 1 = Bit order reverse for CRC checksum. Note: If the checksum data is 0XDD7B0F2E, the bit order reversed for CRC checksum is 0x74F0DEBB.

[24]	**DATREV**	**Write Data Order Reverse** 0 = No bit order reversed for CRC write data in. 1 = Bit order reversed for CRC write data in (per byte). Note: If the write data is 0xAABBCCDD, the bit order reverse for CRC write data in is 0x55DD33BB
[23]	**TRGEN**	**TRGEN** 0 = No effect. 1 = CRC DMA data read or write transfer Enabled. Note1: If this bit assert indicates the CRC engine operation in CRC DMA mode, do not fill in any data in PDMA_CRCDAT register. Note2: When CRC DMA transfer is completed, this bit will be cleared automatically. Note3: If the bus error occurs, all CRC DMA transfer will be stopped. Software must reset all DMA channel, and then trigger again.
[22:2]	**Reserved**	Reserved.
[1]	**CRCRST**	**CRC Engine Reset** 0 = No effect. 1 = Reset the internal CRC state machine and internal buffer. The contents of control register will not be cleared. This bit will automaticallybe cleared after few clock cycles. Note: When operated in CPU PIO mode, setting this bit will reload the initial seed value.
[0]	**CRCEN**	**CRC Channel Enable** Setting this bit to 1 enables CRC's operation. When operation in CRC DMA mode (TRGEN = 1), if user clears this bit, the DMA operation will be continuous until all CRC DMA operation is done, and the TRGEN bit will asserted until all CRC DMA operation done. But in this case, the PDMA_CRCINTF [TXOKIF] flag will inactive, user can read CRC result by reading PDMA_CRCCHKS register when TRGEN = 0 When operation in CRC DMA mode (TRGEN = 1), if user wants to stop the transfer immediately, user can write 1 to CRCRST bit to stop the transmission.

CRCDMA Transfer Source Address Register (PDMA_CRCSA)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCSA	PDMA_BA+0xE04	R/W	CRC DMA Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**SRCADDR**	**CRC DMA Transfer Source Address Register** This field indicates a 32-bit source address of CRC DMA. Note: The source address must be word alignment.

CRCDMA Transfer Byte Count Register (PDMA_CRCBC)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCBC	PDMA_BA+0xE0C	R/W	CRC DMA Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**BYTECNT**	**CRC DMA Transfer Byte Count Register** This field indicates a 16-bit transfer byte count number of CRC DMA.

CRC DMA Current Source Address Register (PDMA_CRCCSA)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCCSA	PDMA_BA+0xE14	R	CRC DMA Current Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**CURSA**	**CRC DMA Current Source Address Register (Read Only)** This field indicates the source address where the CRC DMA transfer just occurs.

CRC DMA Current Byte Count Register (PDMA_CRCCBC)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCCBC	PDMA_BA+0xE1C	R	CRC DMA Current Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CURBC**	**CRC DMA Current Byte Count Register (Read Only)** This field indicates the current remained byte count of CRC_DMA. Note:CRCRST will clear this register value.

CRC DMA Interrupt Enable Control Register (PDMA_CRCINTEN)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCINTEN	PDMA_BA+0xE20	R/W	CRC DMA Interrupt Enable Register	0x0000_0001

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TXOKIEN**	**CRC DMA Transfer Done Interrupt Enable** 0 = Interrupt generator DisabledwhenCRC DMA transfer is done. 1 = Interrupt generator EnabledwhenCRC DMA transfer is done.
[0]	**TXABTIEN**	**CRC DMA Read/Write Target Abort Interrupt Enable** 0 = Target abort interrupt generation Disabled during CRC DMA transfer. 1 = Target abort interrupt generation Enabled during CRC DMA transfer.

CRC DMA Interrupt Status Register (PDMA_CRCINTF)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCINTF	PDMA_BA+0xE24	R/W	CRC DMA 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						**TXOKIF**	**TXABTIF**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TXOKIF**	**Block Transfer Done Interrupt Flag** This bit indicates that CRC DMA has finished all transfer. 0 = Not finished. 1 = Done. Software can write 1 to clear this bit to zero.
[0]	**TXABTIF**	**CRC DMA Read/Write Target Abort Interrupt Flag** 0 = No bus ERROR response received. 1 = Bus ERROR response received. Software can write 1 to clear this bit to zero.

Note: PDMA_CRCINTF [TXABTIF] indicate if bus master received ERROR response or not.If bus master received ERROR response, it means that target abort is happened. DMA will stop transfer and respond this event to software then go to IDLE state. When target abort occurred, software must reset DMA, and then transfer those data again.

CRC Write Data Register (PDMA_CRCDAT)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCDAT	PDMA_BA+0xE80	R/W	CRC Write Data Register	0x0000_0000

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 Write Data Register** When operated in CPU PIO (PDMA_CRCCTL.CRCEN = 1, PDMA_CRCCTL.TRGEN = 0) mode, software can write data to this field to perform CRC operation; When operated in CRC DMA mode (PDMA_CRCCTL.CRCEN = 1, PDMA_CRCCTL.TRGEN = 0), this field will be used for DMA internal buffer. Note1: When operated in CRC DMA mode, so don't filled any data in this field. Note2: The PDMA_CRCCTL.DATFMT and PDMA_CRCCTL.DATREV bit setting will affect this field; for example, if DATREV = 1, if the write data in PDMA_CRCDAT register is 0xAABBCCDD, the read data from PDMA_CRCDAT register will be 0x55DD33BB.

CRC Seed Register (PDMA_CRCSEED)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCSEED	PDMA_BA+0xE84	R/W	CRC Seed Register	0xFFFF_FFFF

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

Bits	Description	
[31:0]	**SEED**	**CRC Seed Register** This field indicates the CRC seed value.

CRC Checksum Register (PDMA_CRCCHKS)

Register	Offset	R/W	Description	Reset Value
PDMA_CRCCHKS	PDMA_BA+0xE88	R	CRC Checksum Register	0x0000_0000

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

Bits	Description	
[31:0]	**CHECKSUM**	**CRC Checksum Register** This field indicates the CRC checksum.

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**
				CHCKEN			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SWRST**

Bits	Description	
[31:17]	**Reserved**	Reserved.
[11:8]	**CHCKEN[3:0]**	**PDMA Controller Channel Clock Enable Control** To enable clock for channel n CHCKEN[n] must be set. CHCKEN[n]= 1: Enable Channel n clock CHCKEN[n]= 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)

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.

Register	Offset	R/W	Description	Reset Value
PDMA_SVCSEL	PDMA_BA+0xF04	R/W	PDMA Service Selection Control Register	0xFFFF_FFFF

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			

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 UART0peripheral transmit (PDMA destination) request.
[19:16]	UARTRXSEL	**PDMA UART0 Receive Selection** This field defines which PDMA channel is connected to UART0peripheral 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 SPI0peripheral 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

31	30	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							

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**GLOBALIF**	**Interrupt Pin Status (Read Only)** GLOBALIF[n] is the interrupt status of PDMA channel n.

7 FLASH MEMORY CONTROLLER (FMC)

7.1 Overview

The ISD9300 series is available with 145/100/68Kbytes 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 141/96/64K Byte device for a total device memory of 145/100/68K Byte.

7.2 Features

- AHB interface compatible
- Runs up to 98 MHz with zero wait-state for continuous address read access
- Mini-cache to reduce flash access and power consumption
- 141/96/64KB 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

7.3 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:

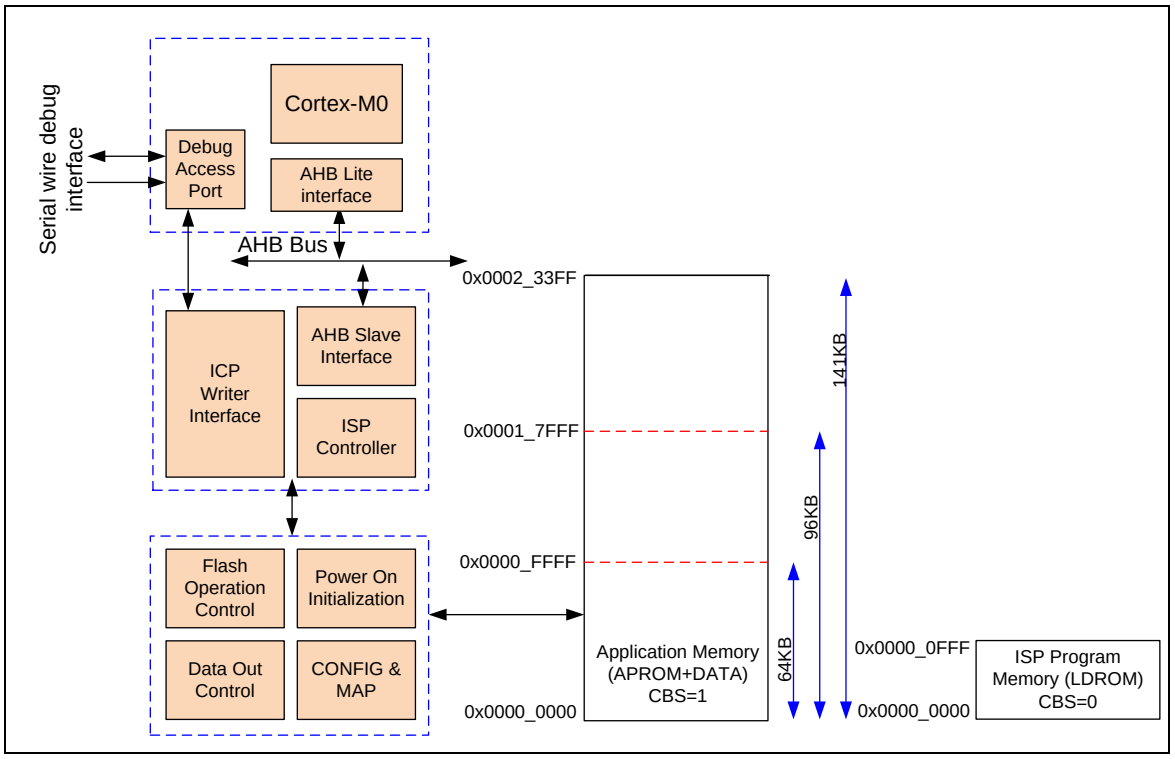


Figure 7-1 Flash Memory Control Block Diagram

7.4 FunctionalDescription

7.4.1 Flash Memory Organization

The ISD9300series flash memory consists of program memory (APROM), Data Flash, ISP loader program memory (LDROM), and user configuration.

Program memory is main memory for user applications and called APROM. User can write their application to APROM and set system to boot from APROM.

ISP loader program memory is designed for a loader to implement In-System-Programming function. LDROM is independent to APROM and system can also be set to boot from LDROM. Therefore, user can use LDROM to avoid system boot fail when code of APROM was corrupted.

Data Flash is used for user to store data. It can be read by ISP read or memory read and programmed through ISP procedure. The size of each erase unit is 512 bytes.

The ISD9300 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.

Block Name	Size	Start Address	End Address
APROM	141 (96/64)KB	0x0000_0000	0x0002_33FF (for 141KB) (or 0x0001_7FFF (for 96KB), or 0x0000_FFFF (for 64KB)) OR DFBADR-1 if DFEN!=0
DATAF	User Configurable	DFBADR	0x0002_33FF (141KB), or 0x0001_7FFF (for 96KB), or 0x0000_FFFF (for 64KB).
LDROM	4 KB	0x0010_0000	0x0010_0FFF
CONFIG	512B	0x0030_0000	0x0030_01FF

Table 7-1 Memory Address Map

The Flash memory organization is shown asFigure 7-2 Flash Memory Organization:

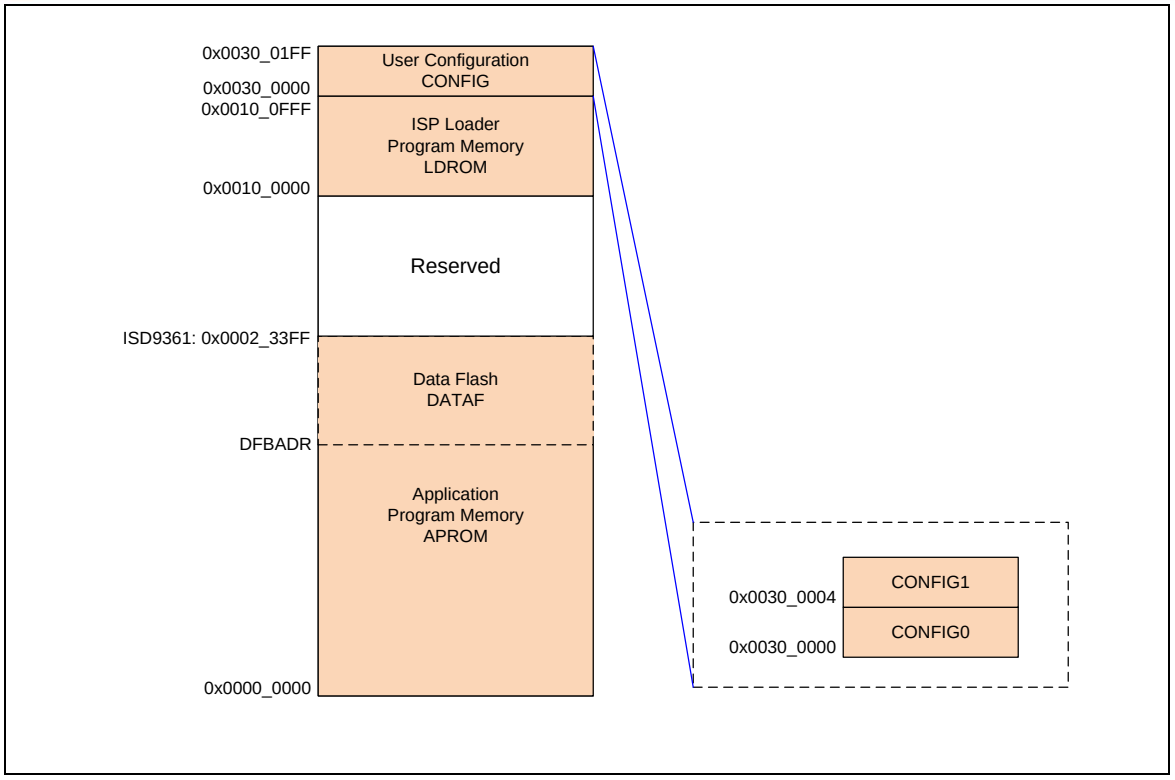


Figure 7-2 Flash Memory Organization

7.4.2 Boot Selection

The ISD9300 provides an in-system programming (ISP) feature to enable user to update the application program memory when the chip is mounted on a PCB. A dedicated 4KBboot loader program memory is used to store ISP firmware. The user customizes this firmware to implement a protocol specific to their system to download updated application code. This firmware could utilize device peripherals such as UART, SPI or I2C to fetch new application code. The memory area from which the ISD9300 boots is controlled by the CBS bit in Config0 register.

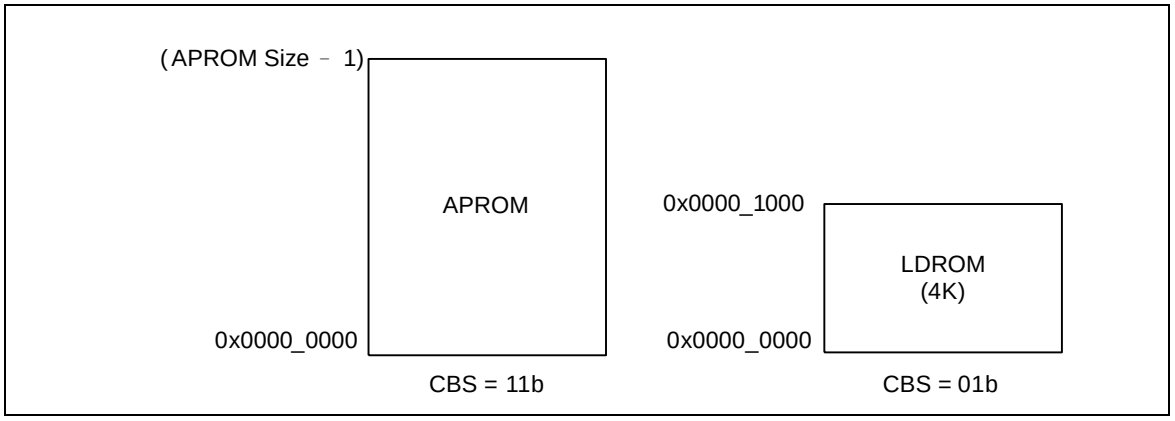


Figure 7-3 Program Executing Range for Booting from APROM and LDROM

7.4.3 Data Flash (DATAF)

The ISD9300 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 DFBADR and application program memory size is (X-N)KB and data flash size is N KB, where X is the total device memory size (141/96/68KBytes) and N is number of Kbytes (sectors) reserved for data flash. In addition, for example 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.

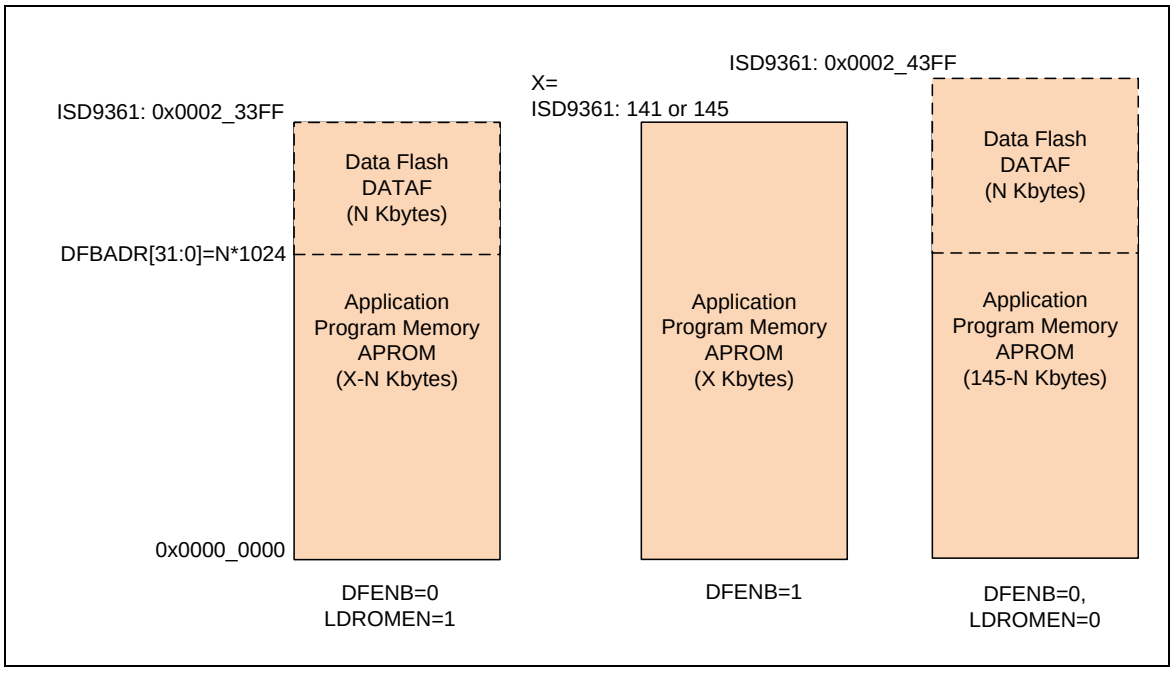


Figure 7-4 Flash Memory Structure

7.4.4 User Configuration

User configuration is internal programmable configuration area for boot options. The user configuration is located at 0x300000 of Flash Memory Organization and they are two 32 bits words. Any change on user configuration will take effect after system reboot.

CONFIG0 (Address = 0x0030_0000)

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

CONFIG0	Address =0x0030_0000	
Bits	**Description**	
[31:23]	**Reserved**	RESERVED for future use
[23]	**CBODEN**	**Brown Out Detector Enable** If set to '0' 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= Enable 1=Disable brown out detect after power on
[22:8]	**Reserved**	RESERVED for future use
[7]	**CBS**	**Configuration Boot Selection** 0 = Chip will boot from LDROM 1 = Chip will boot from APROM
[6:3]	**Reserved**	RESERVED for future use
[2]	**LDROMEN**	**LDROM 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

CONFIG1 (Address = 0x0030_0004)

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

Config	Address =0x0030_0004	
Bits	Description	
[31:20]	**Reserved**	Reserved (It is mandatory to program 0x00 to theseReserved 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.

7.4.5 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 ISD9300 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.

7.4.5.1 ISP Procedure

The ISD9300 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. The boot switching flow by BS bit is shown in the following figure.

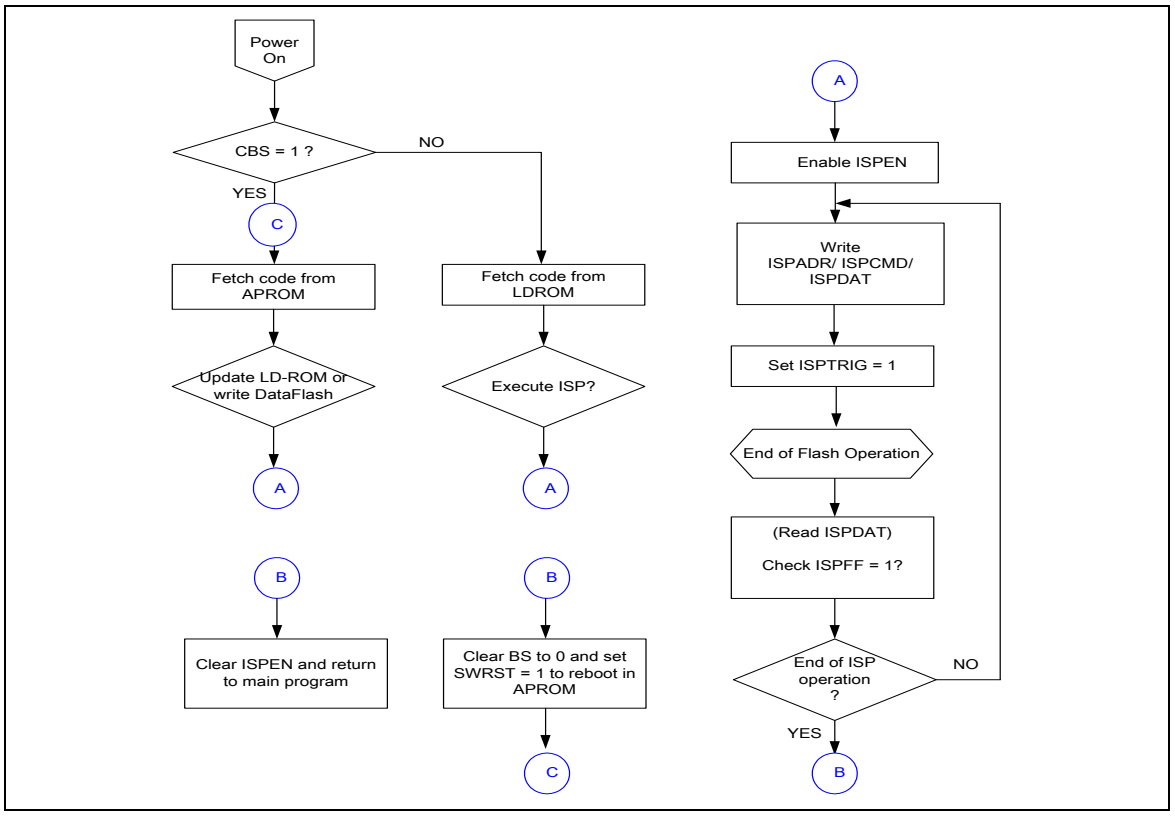


Figure 7-5 Example Flow of Boot Selection by BS Bit

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.

The ISP command set is shown in Table 7-2 ISP Command List**Error! Reference source not found.**. 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.

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_02nn.
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 7-2 ISP Command List

There is an ISP command to read the device ID register. This register returns a code that reports the memory configuration of the ISD9300 series part as given inTable 7-3 Device ID Memory Size

DID[7:0]	RAM Size (KB)
1	-
2	-
3	-
4	16

DID[15:8]	Flash Size (KB)
5	-
6	68
7	100
8	145

Table 7-3 Device ID Memory Size

7.4.6 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

7.4.7 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

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		CACHEDIS	Reserved		WAITCFG		
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
SWRST	ISPFF	LDUEN	CFGUEN	Reserved		BS	ISPEN

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21]	**CACHEDIS**	**Cache Disable** When set to 1, caching of flash memory reads is disabled.
[20:19]	**Reserved**	Reserved.
[18:16]	**WAITCFG**	**Flash Access Wait State Configuration** For M and H speed grade parts this sets the access speed to the flash memory. 0x00: Three wait states. HCLK > 72MHz 0x01: Two wait states. 72MHz > HCLK > 49.152MHz 0x02: One wait state. HCLK <= 49.152MHz Before changing WAITCFG, ensure HCLK speed is < 49.152MHz.
[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 modifydevice 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

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

Bits	Description	
[31:0]	**ISPADDR**	**ISP Address Register** This is the memory address register that a subsequent ISP command will access. ISP operation are carried out on 32bit words only, consequently ISPADDR [1:0] must be 00b for correct ISP operation.

ISP Data Register (FMC_ISPDAT)

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

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

Bits	Description	
[31:0]	**ISPDAT**	**ISP Data Register** Write data to this register before an ISP program operation. Read data from this register after an ISP read operation

ISP Command (FMC_ISPCMD)

Register	Offset	R/W	Description	Reset Value
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**CMD**	**ISP Command** ISP command table is shown below: 0x00 = Read. 0x0B = Read Company ID (0xDA). 0x0C = Read Device ID. 0x21 = Program. 0x22 = Page Erase. 0x3x = Standby.

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

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**ISPGO**	**ISP Start Trigger (Write-Protection Bit)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. 0 = ISP operation finished. 1 = ISP progressed. After triggering an ISP function M0 instruction pipeline should be flushed with a ISB instruction to guarantee data integrity.

Data Flash Base Address Register (FMC_DFBA)

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

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

Bits	Description	
[31:0]	**DFBA**	**Data Flash Base Address** This register reports the data flash starting address. It is a read only register. Data flash size is defined by user configuration; register content is loaded from Config1 when chip is reset.

8 ANALOG SIGNAL PATH BLOCKS

This section describes the functional blocks that perform analog signal functions on the ISD9300. This includes the ADC, DPWM Speaker Driver, PGA Gain Amplifier, Automatic Gain Control and a variety of auxiliary analog functional blocks.

8.1 Audio Analog-to-Digital Converter (ADC)

8.1.1 Overview

The ISD9300 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.

8.1.2 Features

- Front-end PGA providing gain range of -12dB – 35dB
- Boost Gain stage of 0dD or 26dB
- Configurable OVSPLRAT (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 words FIFO, accessible via APB and PDMA

8.1.3 Block Diagram

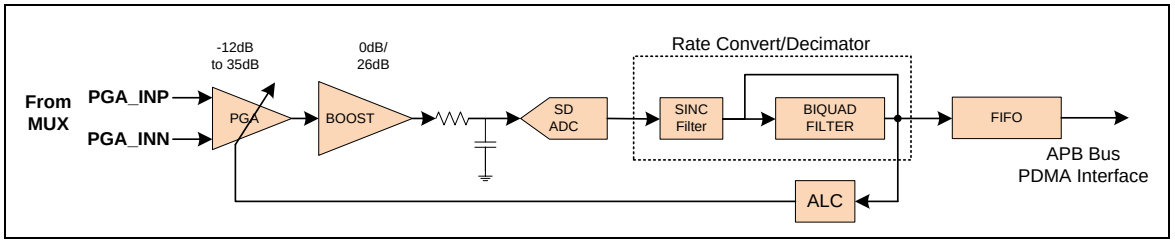


Figure 8-1 ADC Signal Path Block Diagram

8.1.4 Functional Description

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.

8.1.4.1 ADC Clock Generator

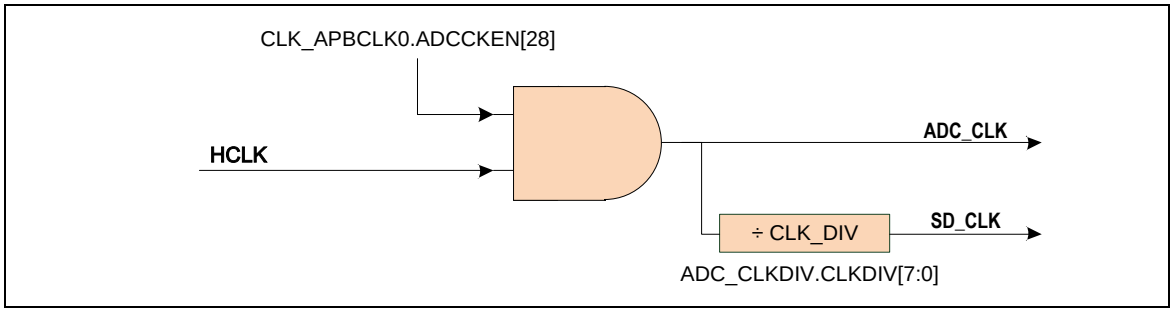


Figure 8-2 ADC Clock Control

8.1.4.2 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.

HCLK=49.152MHz	SDCLK	Sample Rate (Hz) for OVSPLRAT			
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 8-1 Sample Rates for HCLK=49.152MHz

HCLK=32.768MHz	SDCLK	Sample Rate (Hz) for OVSPLRAT			
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 8-2 Sample Rates for HCLK=32.768MHz

HCLK=24.576MHz	SDCLK	Sample Rate (Hz) for OVSPLRAT

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 8-3 Sample Rates for HCLK=24.576MHz

HCLK=16.384MHz	SDCLK	Sample Rate (Hz) for OVSPLRAT			
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

Table 8-4 Sample Rates for HCLK=16.384MHz

8.1.4.3 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.

8.1.4.4 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.

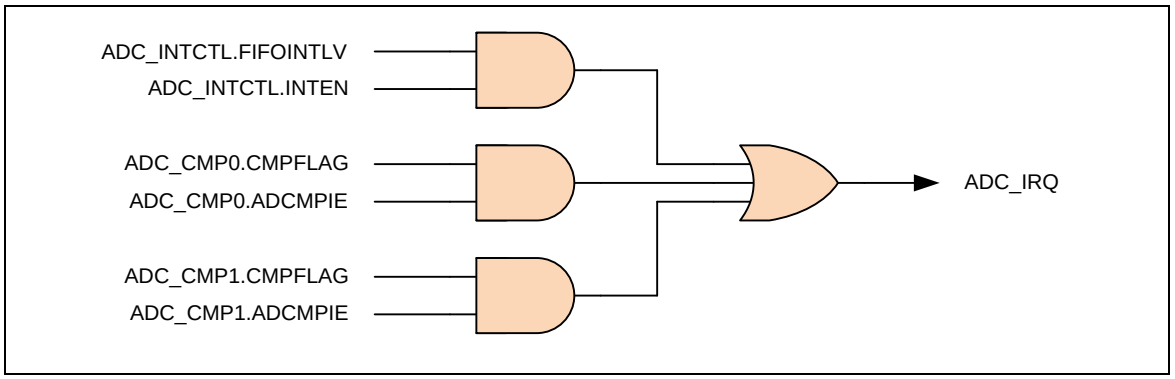


Figure 8-3 SDADC Controller Interrupt

8.1.4.5 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.

8.1.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
ADCBase 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

8.1.6 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

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

Bits	Description	
[31:16]	**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

31	30	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							**CHEN**

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	ADCClock Divider Register	0x0000_0000

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

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

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[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.
[15:4]	**Reserved**	Reserved.
[3:0]	**OVSPLRAT**	**Decimation Over-Sampling Ratio** This term determines the over-sampling ratio of the decimation filter. Valid values are: 0: OVSPLRAT=64 1: OVSPLRAT=128 2: OVSPLRAT=192 3: OVSPLRAT=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

31	30	29	28	27	26	25	24
INTEN	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					FIFOINTLV		

Bits	Description	
[31]	**INTEN**	**Interrupt Enable** If set to '1' an interrupt is generated whenever FIFO level exceeds that set in FIFOINTLV.
[30:3]	**Reserved**	Reserved.
[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

31	30	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							**RXDMAEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[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 0/1 (ADC_CMP0/1)

Register	Offset	R/W	Description	Reset Value
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

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

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.

8.2 Audio Class D Speaker Driver (DPWM)

8.2.1 Overview

The ISD9300 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.

8.2.2 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

8.2.3 Block Diagram

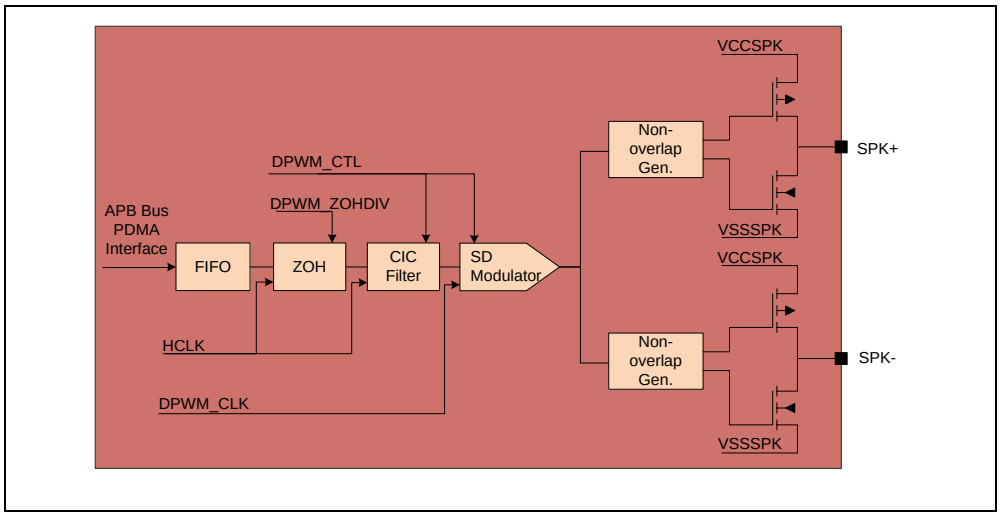


Figure 8-4 DPWM Block Diagram

8.2.4 Functional Description

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. Ultimate SNR (Signal-to-Noise Ratio) is determined by the time resolution of the master clock.

8.2.4.1 Determining Sample Rate

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

$$F_s = HCLK \div DPWM\_ZOHDIV \div 64$$

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

HCLK (MHz)	DPWM_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

Table 8-5 DPWM Sample Rates for Various HCLK

8.2.4.2 Configuring Speaker Driver

To operate the speaker driver the following configuration is recommended:

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

8.2.4.3 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.

8.2.5 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 DATA FIFO Status Register	0x0000_0002
DPWM_DMACTL	DPWM_BA+0x08	R/W	DPWM PDMA Control Register	0x0000_0000
DPWM_DATA	DPWM_BA+0x0C	W	DPWM DATA FIFO Input	0x0000_0000
DPWM_ZOHDIV	DPWM_BA+0x10	R/W	DPWM Zero Order Hold Division Register	0x0000_0030

8.2.6 Register Description

DPWM Control Register (DPWM_CTL)

Register	Offset	R/W	Description	Reset Value
DPWM_CTL	DPWM_BA+0x00	R/W	DPWM Control Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[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 = 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 DATA FIFO Status Register	0x0000_0002

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**EMPTY**	**FIFO Empty** 0= FIFO is not empty. 1= FIFO is empty.
[0]	**FULL**	**FIFO Full** 0 = FIFO is not full. 1 = FIFO is full.

DPWM PDMA Control Register(DPWM_DMACTL)

Register	Offset	R/W	Description	Reset Value
DPWM_DMACTL	DPWM_BA+0x08	R/W	DPWM PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**DMAEN**	**Enable DPWM DMA Interface** 0= Disable PDMA. No requests will be made to PDMA controller. 1= Enable PDMA. Block will request data from PDMA controller whenever FIFO is not empty.

DPWM FIFO Input (DPWM_DATA)

Register	Offset	R/W	Description	Reset Value
DPWM_DATA	DPWM_BA+0x0C	W	DPWM DATA FIFO Input	0x0000_0000

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

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**ZOHDIV**	**DPWM Zero Order Hold, Down-Sampling Divisor** The input sample rate of the DPWM is set by HCLK frequency and the divisor set in this register by the following formula: Fs=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.

8.3 Analog Comparator (ACMP)

8.3.1 Overview

ISD9300 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 inFigure 8-5 Analog Comparator Block Diagram.

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

8.3.2 Features

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

8.3.3 Block Diagram

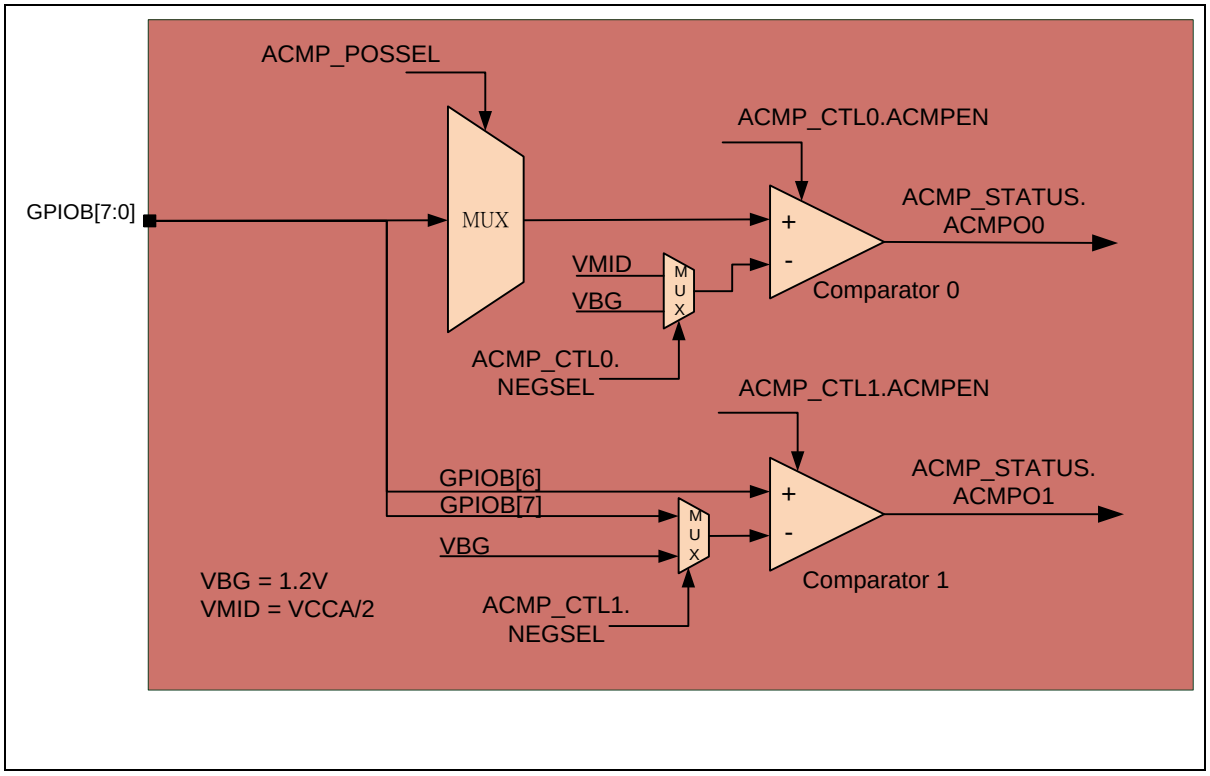


Figure 8-5 Analog Comparator Block Diagram

8.3.4 Functional Description

8.3.4.1 Setup Procedure

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

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

8.3.4.2 Interrupt Sources

The comparator generates an output CO*n* (n=0,1) which is reported inACMP_STATUS register. If CMP*n*CR.IEbit 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.

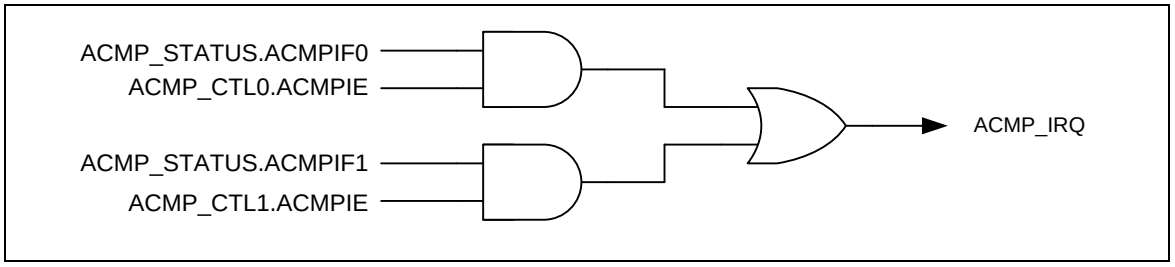


Figure 8-6 Comparator Controller Interrupt Sources

8.3.4.3 Hysteresis Function

The analog comparator provides hysteresis function to make the comparator output transition more stable. If comparator output is 0, it will not change to 1 until the positive input voltage exceeds the negative input voltage by a positive hysteresis voltage. Similarly, if comparator output is 1, it will not change to 0 until the positive input voltage drops below the negative input voltage by a negative hysteresis voltage.

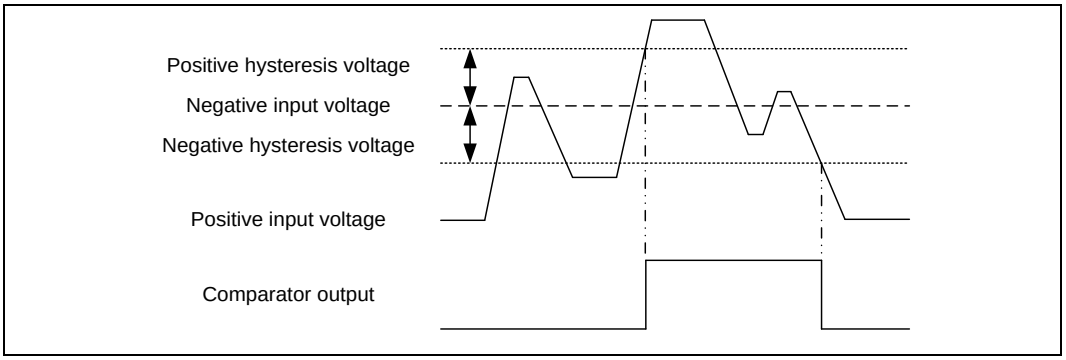


Figure 8-7 Comparator Hysteresis Function

8.3.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
ACMPBase 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

8.3.6 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

31	30	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			**NEGSEL**	**Reserved**		**ACMPIE**	**ACMPEN**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**NEGSEL**	**Comparator0Negative Input Select** 0 = VBG, Bandgap reference voltage = 1.2V. 1 = VMID reference voltage = VCCA/2.
[3:2]	**Reserved**	Reserved.
[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

31	30	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			**NEGSEL**	**Reserved**		**ACMPIE**	**ACMPEN**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**NEGSEL**	**Comparator1Negative Input Select** 0 = GPIOB[7]. 1 = VBG, Bandgap reference voltage is 1.2V.
[3:2]	**Reserved**	Reserved.
[1]	**ACMPIE**	**CMP0 Interrupt Enable** 0 = Disable CMP0 interrupt function. 1 = Enable CMP0 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

31	30	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				ACMPO1	ACMPO0	ACMPIF1	ACMPIF0

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**ACMPO1**	**Comparator1 Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (CMP1EN = 0).
[2]	**ACMPO0**	**Comparator0 Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (CMP0EN = 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

31	30	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					**POSSEL**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**POSSEL**	**Comparator0 GPIO Selection** GPIOB[POSSEL] is the active analog GPIO input selected to Comparator 0 positive input.

8.4 Analog Functional Blocks

8.4.1 Overview

The ISD9300 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.

8.4.2 Features

- VMID reference voltage generation.
- Current source generation for AGPIO (Analog enabled GPIO).
- LDO control for GPIOA[7:0]&& GPIOB[13:12] power domain and external device use.
- Microphone Bias generator.
- Analog Multiplexor.
- Programmable Gain Amplifier (PGA).
- OSC48M Frequency Control.
- CapSense Relaxation Oscillator.
- Oscillator Frequency Measurement block.

8.4.3 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 inFigure 8-8 VMID Reference Generation.

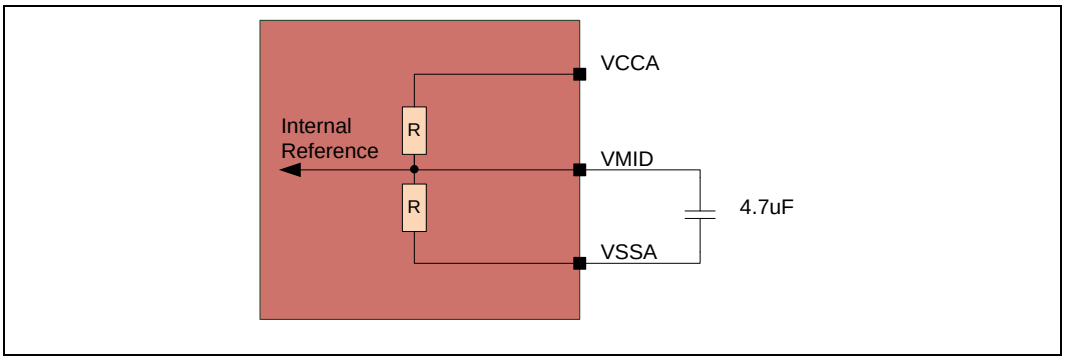


Figure 8-8 VMID Reference Generation

8.4.4 GPIO Current Source Generation

The GPIOB[11:0] and GPIOA[11:8] provide 16 pins 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 sensing in combination with the relaxation oscillator control circuit.

The current generation block consists of a programmable current source controlled by ANA_CURCTL0.VAL and individual switches to each of the GPIO pins as shown inFigure 8-9 GPIO Current Source Generation. Power control for this block is merged with the analog comparator, this block must be enabled to use current source (ACMP->CMP0CR.EN=1).

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

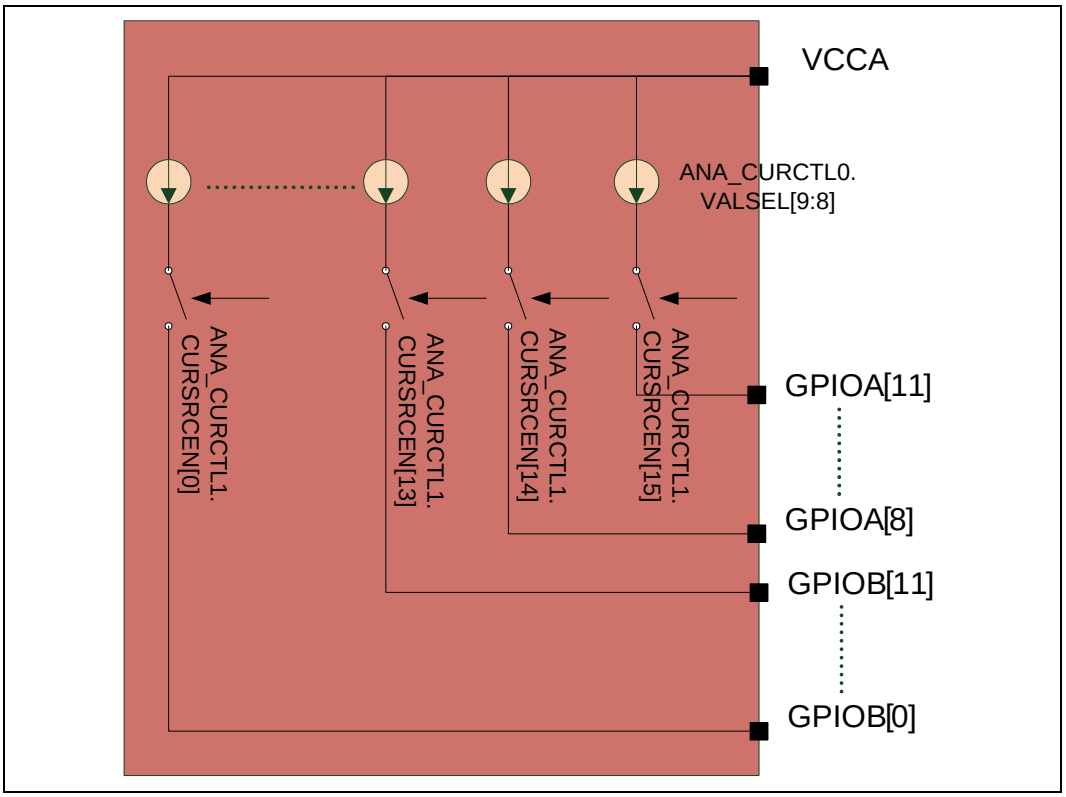


Figure 8-9 GPIO Current Source Generation

8.4.5 LDO Power Domain Control

The ISD9300 provides a Low Dropout Regulator (LDO) that provides power to the I/O domain of GPIOA[7:0]& GPIOB[13:12]. 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]& GPIOB[13:12] 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]& GPIOB[13:12] is usable (unless VD33=VCCD).

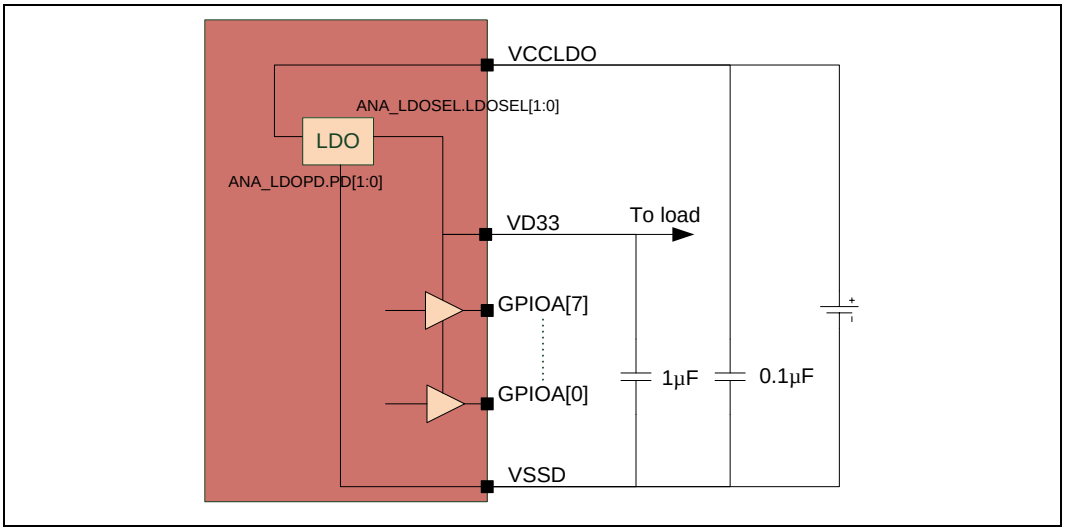


Figure 8-10 LDO Power Domain

8.4.6 Microphone Bias Generator

The ISD9300 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% to90% of the VCCA supply voltage (seedescription 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 ofFigure 8-12 MICBIAS Application Diagram, 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 ISD9300, the minimum total resistance $(R_1 + R_2)$is 4Kohms. MICBIAS output voltage should be such that the following condition is met:

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

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

FromFigure 8-12 MICBIAS Application Diagram, MIC_IN1 and MIC_IN2 areAC coupled to the ISD9300 MIC+ and MIC- respectively for differential inputs. In single-ended operation, MIC_IN1 should go to MIC- of the ISD9300. $C_1$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_2providing additional rejection from ground noise.

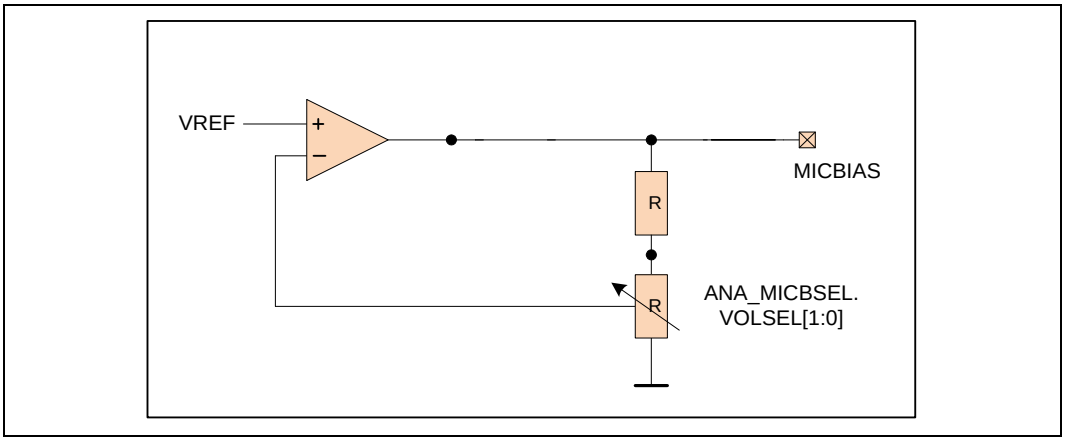


Figure 8-11 MICBIAS Block Diagram

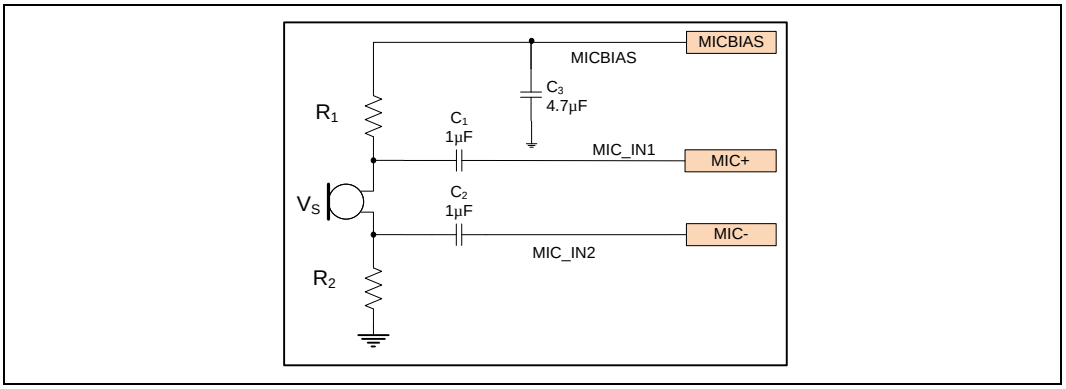


Figure 8-12 MICBIAS Application Diagram

8.4.7 Analog Multiplexer

The ISD9300 provides an analog multiplexer (ANA_MUXCTL) 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 8-13 Analog Multiplexer Control Signalsshows the multiplexer control signals.

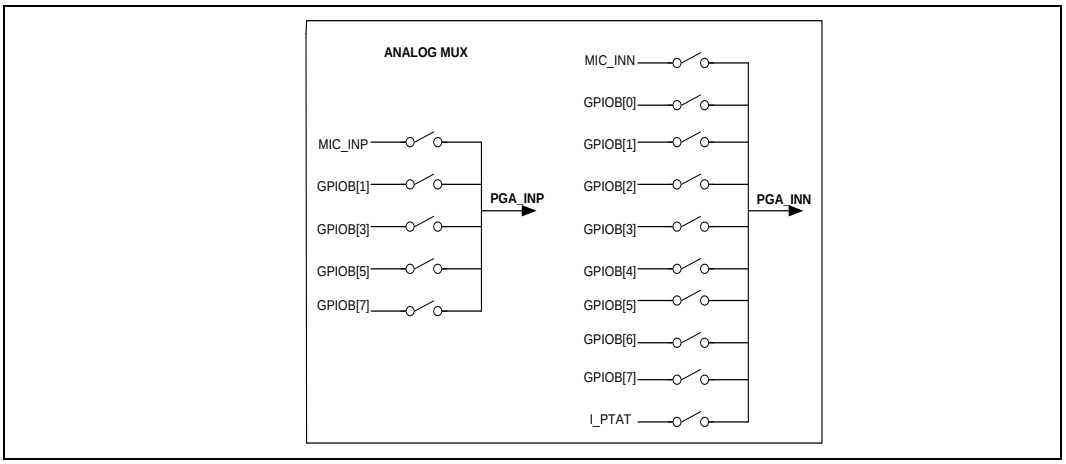


Figure 8-13 Analog Multiplexer Control Signals

8.4.8 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:

- Enable the multiplexer, PGA, IPBOOST, and sigma-delta modulator.
- Have the multiplexer select I_PTAT current as input and choose VBG (bandgapvoltage) as reference (REFSEL).
- Set the 6-bit PGA_GAIN[5:0] gain value to hex 0x17 and choose 0dB gain setting for IPBOOST gain block.
- The temperature can be inferred by the information given in Table 8-6 Temperature Sensor Measurementand equation below.

$$T\ (°C) = 27 + (ADC\_VAL - 0x42EA)/50.$$

The settings corresponding to this configuration are:

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

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

Table 8-6 Temperature Sensor Measurement

8.4.9 Programmable Gain Amplifier

The ISD9300 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 8-14 PGA Signal Path Block Diagramshows 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, PU_IPBOOST). The PGA and IPBOOST 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 8-7 PGA Input Impedance Variation with Gain Settingshows a table of input impedance for different gain setting.

The IPBOOST 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/IPBOOST blocks with an attenuation of -45dB at 6MHz frequency. The signal path defaults to have VCCA/2 as the reference voltage.

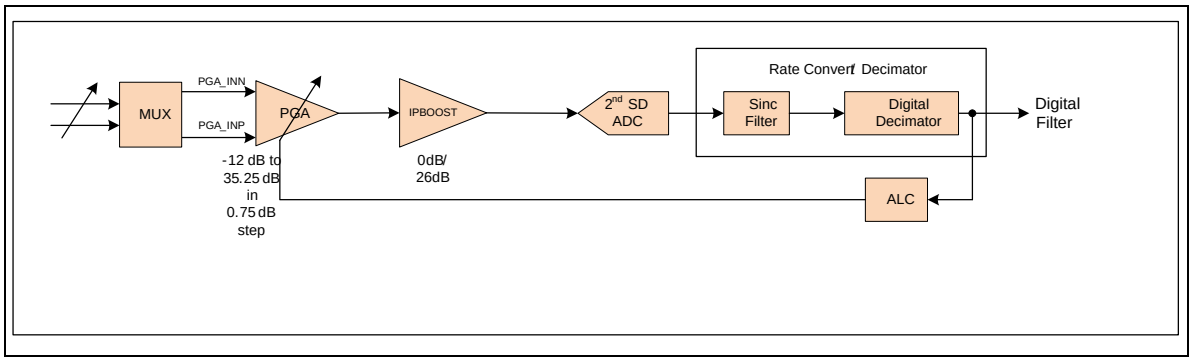


Figure 8-14 PGA Signal Path Block Diagram

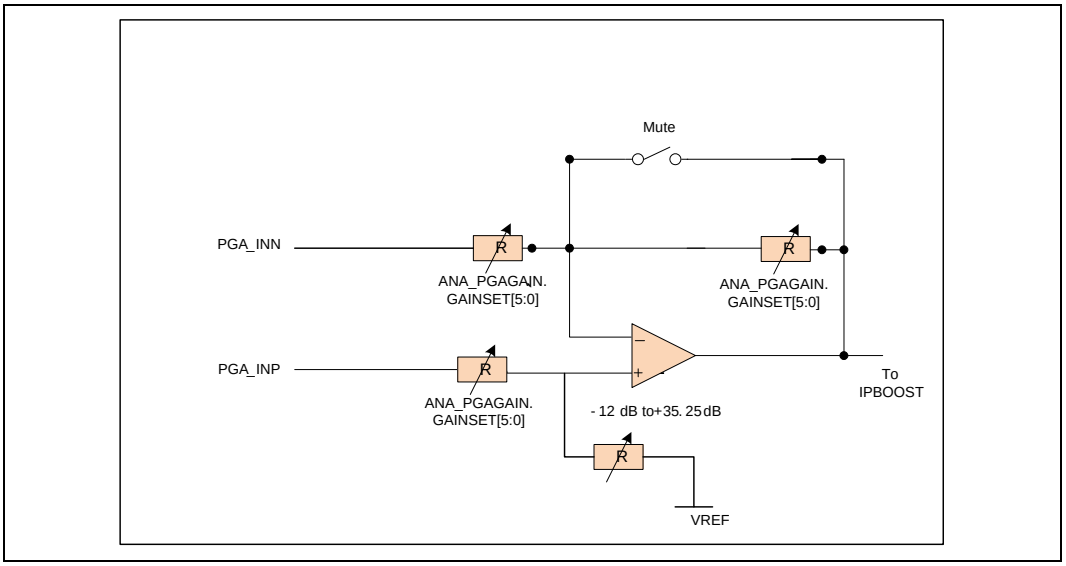


Figure 8-15 PGA Structure

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
MICPImpedance (kΩ)	94	94	94	94	94	94	94	94	94	94	94	94

Table 8-7 PGA Input Impedance Variation with Gain Setting

8.4.10 CapSense Relaxation Oscillator/Counter

The ISD9300 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 CapSense 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 inFigure 8-16 CapSense Function Block Diagram.

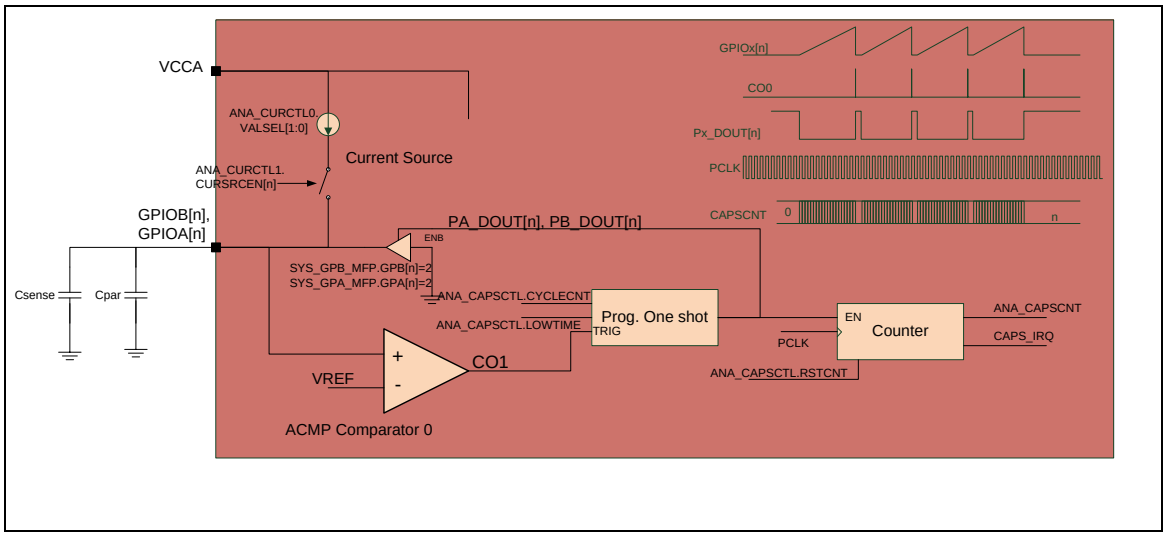


Figure 8-16 CapSense Function Block Diagram

8.4.10.1Functional Description

The principle behind the operation of this block is that a certain capacitance is present on one of the analog enabled GPIO (GPIOB[11:0],GPIOA[11:8]). 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 CapSense mode by setting SYS_GPB_MFP.GPBn = 2 and enabling a current source to this pin (ANA_CURCTL1.CURSREN = 2^n) (n=0:11) or setting SYS_GPA_MFP.GPAm = 2 and enabling a current source to this pin (ANA_CURCTL1.EN = 2^(m+4)) m=8:11. The Analog Comparator 0 is also setup to compare the voltage at the pin to a reference voltage (ACMP_POSSEL = n, ACMP_CMP0CR.EN = 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 CapSense 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^CYCLECNT 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.

8.4.10.2Design Considerations

Selecting parameters for CapSense 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.

8.4.11 Oscillator Frequency Measurement and Control

The ISD9300 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 inFigure 8-17 Oscillator Frequency Measurement Block Diagram.

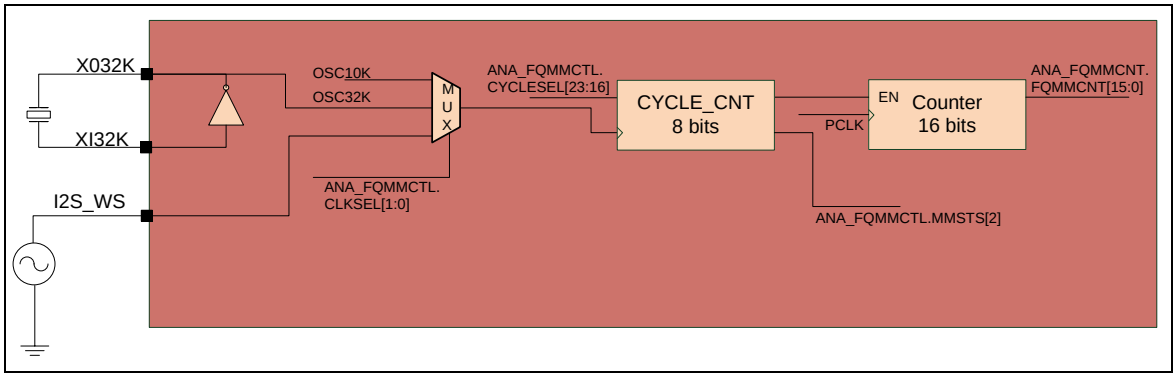


Figure 8-17 Oscillator Frequency Measurement Block Diagram

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. In addition, to obtain frequencies in between SYS_IRCTCTLtrim 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.

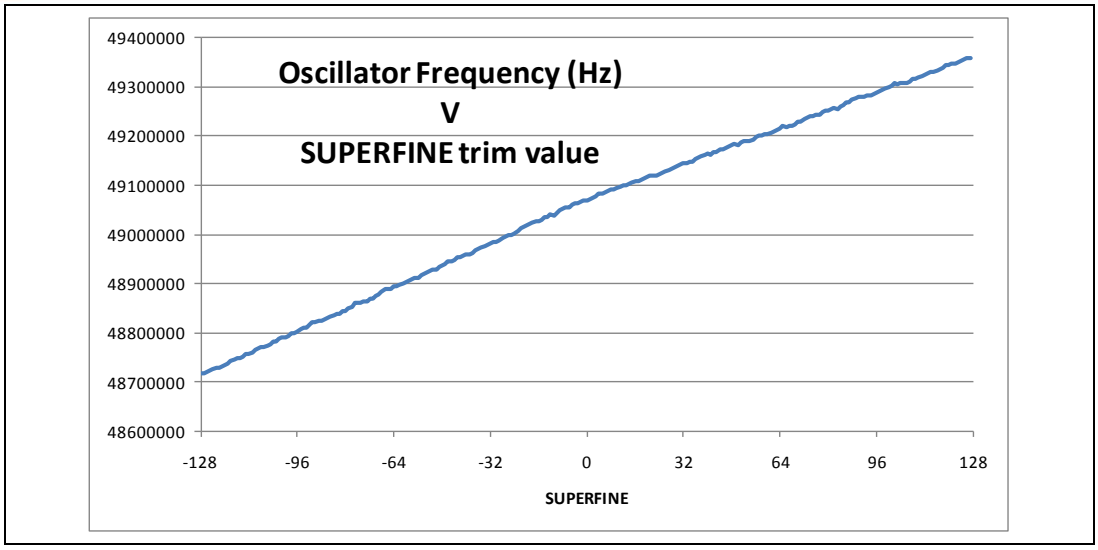


Figure 8-18 Example SUPERFINE Trim Frequency Adjustment

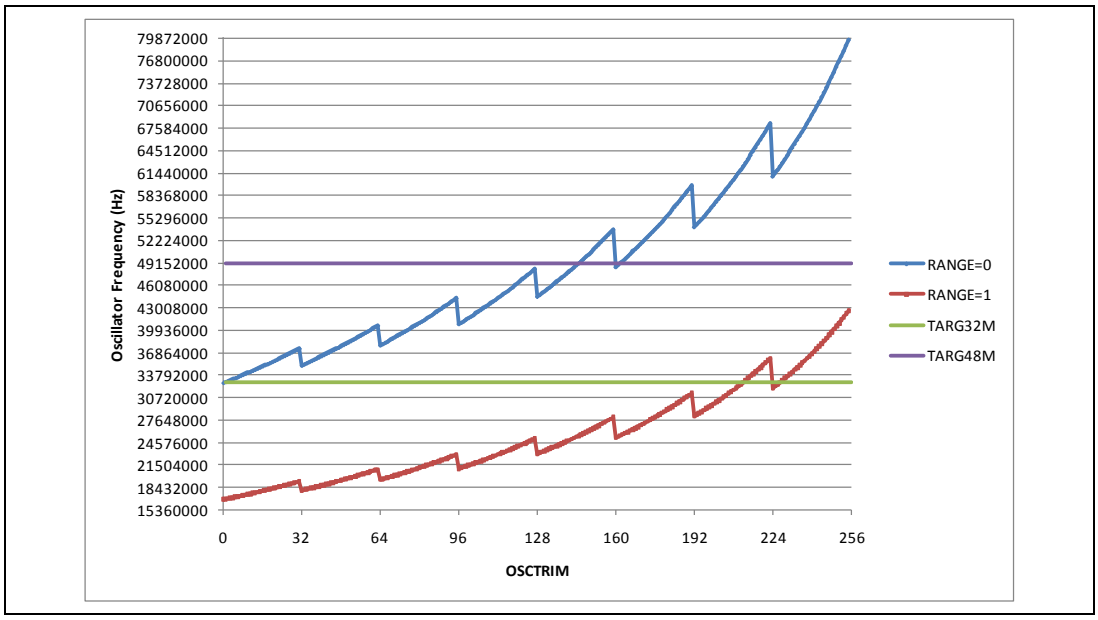


Figure 8-19 Typical Oscillator Frequency versus SYS_IRCTCTL Setting

8.4.12 Register Map

R: read only, **W**: write only, **R/W**: both read and 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_CURCTL1	ANA_BA+0x0C	R/W	Current Source Control Register 1	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
ANA_FQMMCYC	ANA_BA+0x9C	R/W	Frequency Measurement Cycle Register	0x0000_0000

8.4.13 Register Description

VMID Control Register (ANA_VMID)

Register	Offset	R/W	Description	Reset Value
ANA_VMID	ANA_BA+0x00	R/W	VMID Reference Control Register	0x0000_0007

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDHIRES**	**Power Down High (360kΩ) Resistance Reference** 0= Connect the High Resistance reference to VMID. Use this setting for minimum power consumption. 1= The High Resistance reference is disconnected from VMID. Default power down and reset condition.
[1]	**PDLORES**	**Power Down Low (4.8kΩ) Resistance Reference** 0= Connect the Low Resistance reference to VMID. Use this setting for fast power up of VMID. Can be turned off after 50ms to save power. 1= The Low Resistance reference is disconnected from VMID. Default power down and reset condition.
[0]	**PULLDOWN**	**VMID Pulldown** 0= Release VMID pin for reference operation. 1= Pull VMID pin to ground. Default power down and reset condition.

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

31	30	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						VALSEL	
7	6	5	4	3	2	1	0
CURSRCEN[7:0]							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**VALSEL**	**Current Source Value** Select master current for source generation 0= 0.5uA. 1= 1uA. 2= 2.5uA. 3=5uA.
[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].

Current Source Control Register (ANA_CURCTL1)

Register	Offset	R/W	Description	Reset Value
ANA_CURCTL1	ANA_BA+0x0C	R/W	Current Source Control 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**
CURSRCEN[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CURSRCEN[7:0]							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CURSRCEN**	**Enable Current Source To GPIOB[X], GPIOA[X-4]** Individually enable current source to GPIO pins. Each GPIOB[11:0] and GPIOA[11:8] pin has a separate current source. 0 = Disable. 1 = Enable current source to pin GPIOB[x] ,x=11~0; GPIOA[x-4] , x=15~12.

LDO Voltage Control Register (ANA_LDOSEL)

Register	Offset	R/W	Description	Reset Value
ANA_LDOSEL	ANA_BA+0x20	R/W	LDO Voltage Select Register	0x0000_0000

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

Bits	Description	
[31:2]	**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 DownRegister	0x0000_0001

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**DISCHAR**	**Discharge** 0 = 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.

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**	

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= VCCA/2 is reference source. 1 = VBG (bandgap voltage reference) is reference source.
[1:0]	**VOLSEL**	**Select Microphone Bias Voltage** MICBMODE=0 0: 90% VCCA 1: 65% VCCA 2: 75% VCCA 3: 50% VCCA MICBMODE=1 0: 2.4V 1: 1.7V 2: 2.0V 3: 1.3V

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

31	30	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							**MICBEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**MICBEN**	**Enable Microphone Bias Generator** 0 = Powered Down. 1 = Enabled.

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

31	30	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	**MUXEN**	**PGAINSEL**	**PTATCUR**	**POSINSEL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NEGINSEL							

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]=1 then GPIOB[n] is connected to PGA_INN.

PGA Enable Register (ANA_PGACTL)

Register	Offset	R/W	Description	Reset Value
ANA_PGACTL	ANA_BA+0x60	R/W	PGA EnableRegister	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				BSTGAIN	PUBOOST	PUPGA	REFSEL

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**BSTGAIN**	**Boost Stage Gain Setting** 0 = Gain = 0dB. 1 = Gain = 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=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

31	30	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	MUTEBST	MUTEPGA	PUADCOP	PUCURB	PUBUFADC	PUBUFPGA	PUZCDCMP

Bits	Description	
[31:7]	**Reserved**	Reserved.
[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.BQALCKEN) must also be enabled and BIQ_CTL.DLCOEFF=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					

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:8]	**GAINREAD**	**Current PGA Gain Value** Read Only. May be different from GAIN register when AGC is enabled and is controlling the PGA gain.
[7:6]	**Reserved**	Reserved.
[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.

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	

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 CAP_CNT 1: Set high to reset CAP_CNT.
[28:16]	**Reserved**	Reserved.
[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.
[7:6]	**Reserved**	Reserved.
[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 dischargeexternal 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

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CAPSCNT**	**Counter Read Back Value Of Capacitive Touch Sensing Block**

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							

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

Bits	Description	
[31]	**FQMMEN**	**FQMMEN** 0 = Disable/Reset block. 1 = Start Frequency Measurement.
[30:24]	**Reserved**	Reserved.
[23:16]	**CYCLESEL**	**Frequency Measurement Cycles** Number of reference clock periods plus one to measure target clock (PCLK). For example if reference clock is OSC32K (T is 30.5175us), set CYCLESEL to 7, then measurement period would be 30.5175*(7+1), 244.1us.
[7:3]	**Reserved**	Reserved.
[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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FQMMCNT**	**Frequency Measurement Count** When MMSTS=1 and FQMMEN=1, this is number of PCLK periods counted for frequency measurement. The frequency will be PCLK=FQMMCNT * Fref /(CYCLESEL+1) Hz Maximum resolution of measurement is Fref /(CYCLESEL+1)*2 Hz

Frequency Measurement Cycle (ANA_FQMMCYC)

Register	Offset	R/W	Description	Reset Value
ANA_FQMMCYC	ANA_BA+0x9C	R/W	Frequency Measurement Cycle Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**FQMMCYC**	**Frequency Measurement Cycles** Number of reference clock periods plus one to measure target clock (PCLK). For example if reference clock is OSC32K (T=30.5175µs), FQMMCYC=7, then measurement period would be 30.5175*(7+1)=244.1µs. This address access same register as ANA_FQMMCTL but allows access to more bits of register.

8.5 Automatic Level Control (ALC)

8.5.1 Overview

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.

8.5.2 Block Diagram

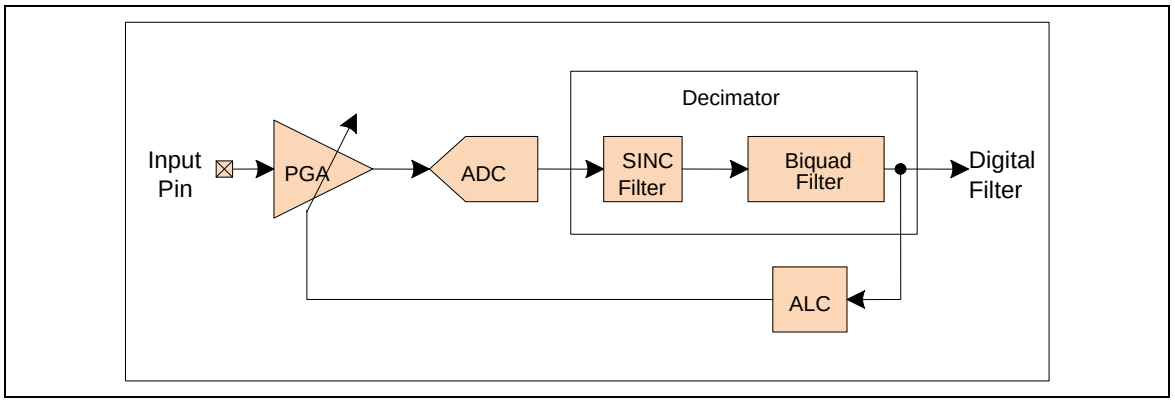


Figure 8-20 ALC Block Diagram

8.5.3 Basic Configuration

The ALC is enabled by setting ALCEN. The ALC shares a clock source with the Biquad filter so CLK_APBCLK0.BQALCKENmust 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)

8.5.4 Functional Description

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 PGASEL[5:0]. A digital peak detector monitors the input signal amplitude and compares it to a register defined threshold level TARGETLV[3:0].

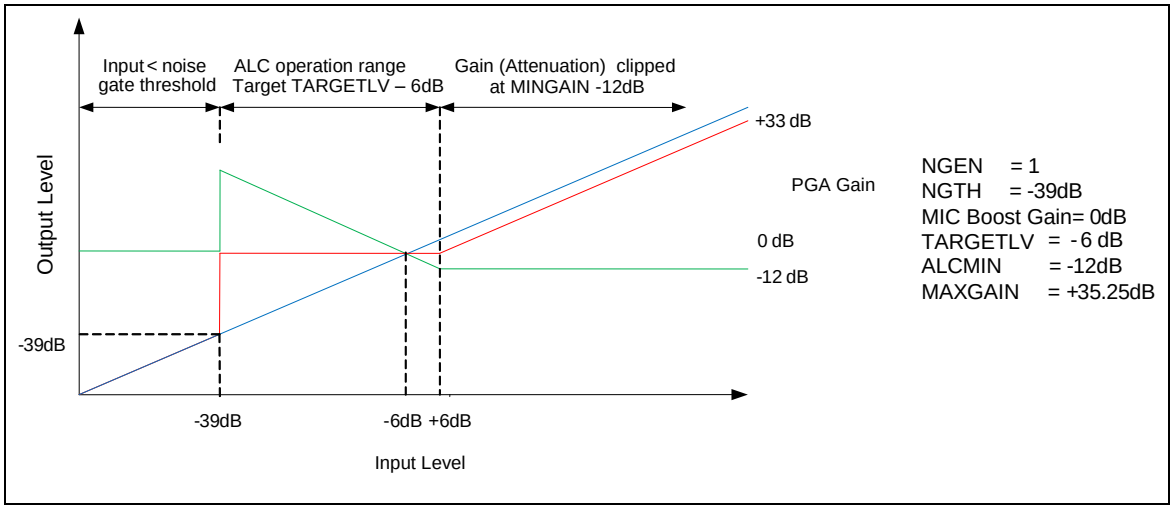


Figure 8-21 ALC Response Graph

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.

8.5.4.1 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.

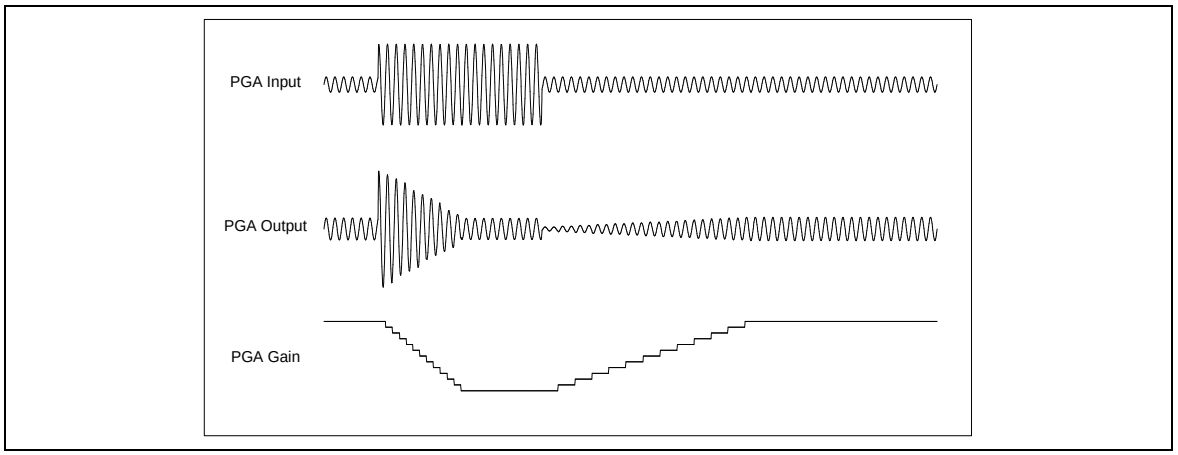


Figure 8-22 ALC Normal Mode Operation

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.

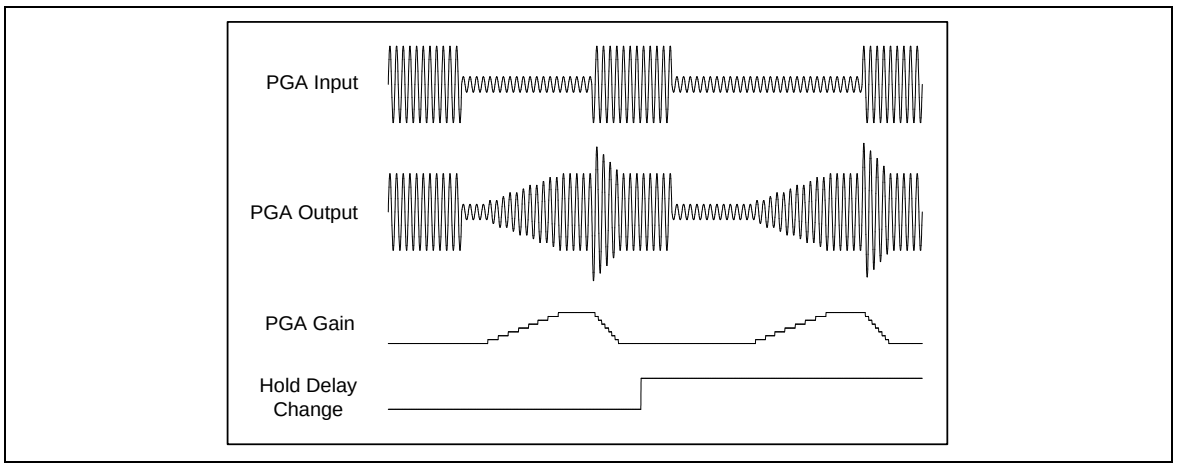


Figure 8-23 ALC Hold Time

8.5.4.2 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.

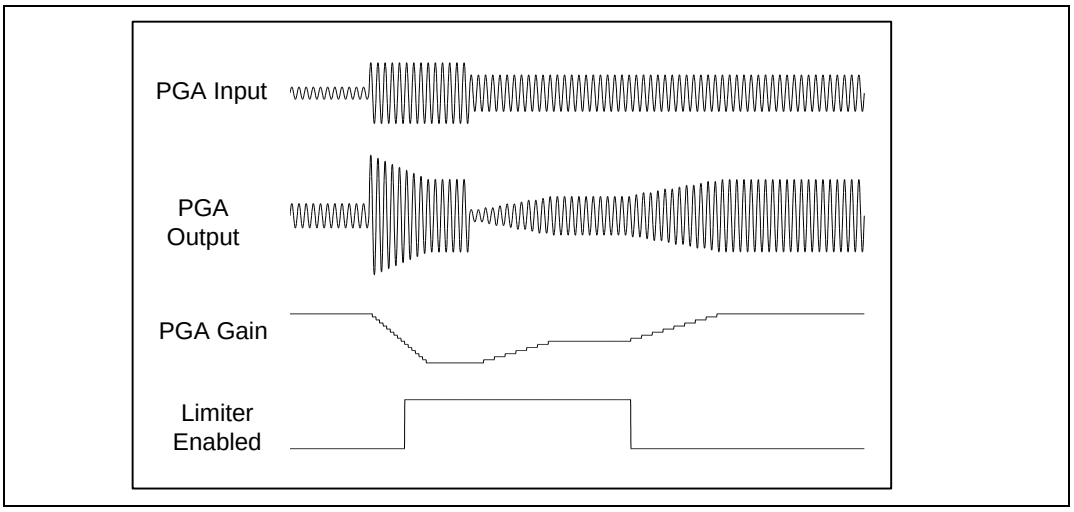


Figure 8-24 ALC Limit Mode Operation

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.

8.5.4.3 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.

8.5.4.4 Decay Times

The decay time DECAYSELis the time constant used when the gain is increasing. In limiter mode, the time constants are faster than in ALC mode.

8.5.4.5 Noise gate (normal mode only)

A noise gate is used when there is no input signal or the noise level is below the noise gate threshold. The noise gate is enabled by setting NGEN to HIGH. It does not remove noise from the signal. The noise gate threshold NGTH is set to a desired level so when there is no signal or a very quiet signal (pause), which is composed mostly of noise, the ALC holds the gain constant instead of amplifying the signal towards the target threshold. The noise gate only operates in conjunction with the ALC (ALCEN HIGH) and ONLY in Normal mode. The noise gate flag is asserted when

(Signal at ADC – PGA gain – MIC Boost gain) < NGTH (dB)

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

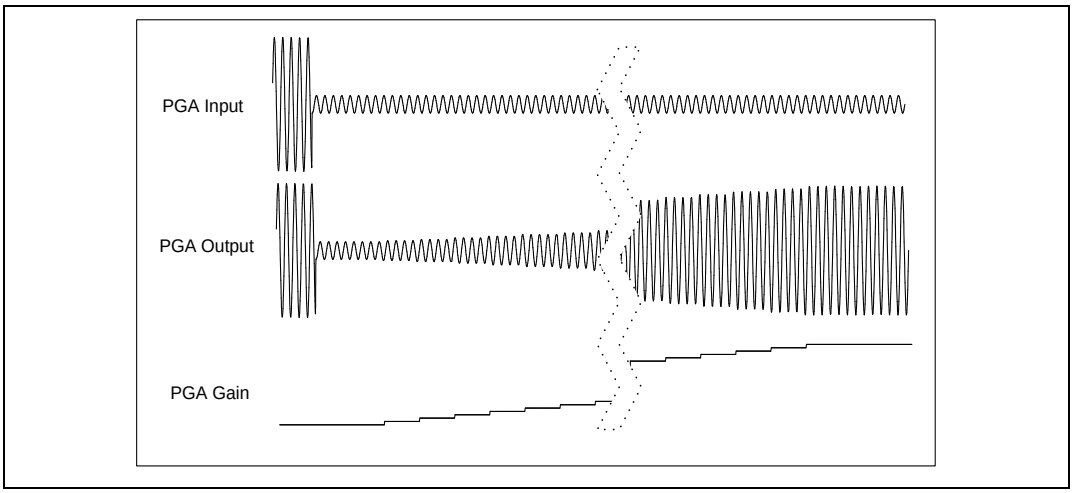


Figure 8-25 ALC Operation with Noise Gate disabled

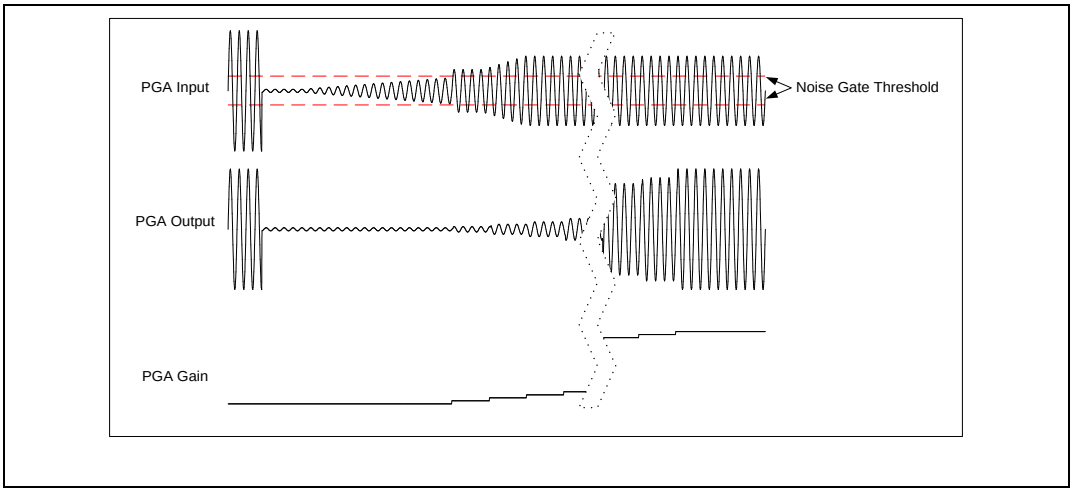


Figure 8-26 ALC Operation with Noise Gate Enabled

8.5.4.6 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.

8.5.5 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 statusregister	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

8.5.6 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

31	30	29	28	27	26	25	24
PKLIMEN	PKSEL	NGPKSEL	ALCEN	MAXGAIN			MINGAIN[2]
23	22	21	20	19	18	17	16
MINGAIN[1:0]		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 = -12dB. 1 = -6dB. 2 = 0dB.

		3 = 6dB. 4 = 12 dB. 5 = 18 dB. 6 = 24 dB. 7 = 30dB.
[21]	**ZCEN**	**ALC Zero Crossing** 0 = zero crossing disabled. 1 = zero crossing enabled.
[20:17]	**HOLDTIME**	**ALC Hold Time** (Value: 0~10). Hold Time = (2^HOLDTIME)ms
[16:13]	**TARGETLV**	**ALC Target Level** 0 = -28.5 dB. 1 = -27 dB. 2 = -25.5 dB. 3 = -24dB. 4 = -22.5 dB. 5 = -21dB. 6 = -19.5 dB. 7 = -18dB. 8 = -16.5 dB. 9 = -15dB. 10 = -13.5 dB. 11 = -12dB. 12 = -10.5 dB. 13 = -9dB. 14 = -7.5 dB. 15 = -6 dB.
[12]	**MODESEL**	**ALC Mode** 0 = ALC normal operation mode. 1 = ALC limiter mode.
[11:8]	**DECAYSEL**	**ALC Decay Time** **(Value: 0~10)** When MODESEL=0, Range: 125us to 128ms When MODESEL= 1, Range: 31us to 32ms (time doubles with every step)
[7:4]	**ATKSEL**	**ALC Attack Time** **(Value: 0~10)** When MODESEL=0, Range: 500us to 512ms When MODESEL=1,Range: 125us to 128ms (Both ALC time doubles with every step)
[3]	**NGEN**	**Noise Gate Enable** 0 = Noise gate disabled. 1 = Noise gate enabled.
[2:0]	**NGTHBST**	**Noise Gate Threshold** Boost disabled: Threshold = (-81+6xNGTHBST) dB Boost enabled: Threshold = (-87+6xNGTHBST) dB

ALC Status Register (ALC_STS)

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

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 NGTH
[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

31	30	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	
[31:1]	**Reserved**	Reserved.
[0]	**INTFLAG**	**ALC Interrupt** 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

31	30	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	
[31:1]	**Reserved**	Reserved.
[0]	**INTEN**	**ALC Interrupt Enable** 0 = INTEN disabled. 1 = INTEN enabled.

8.6 Biquad Filter (BIQ)

8.6.1 Overview

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.

$$H(z) = \frac{\mathrm{b}_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + \mathrm{a}_1 \mathrm{z}^{-1} + \mathrm{a}_2 \mathrm{z}^{-2}}$$

Upon power on reset or when the BIQ_CTL.DLCOEFF=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.

8.6.2 Basic Configuration

Note that the fixed point coefficients have the format of 3.16 (19 bits) and are stored in the coefficient RAM under normal operation. It takes 32 internal system clocks for the automatic write to finish when the BIQ_CTL.DLCOEFF bit is released; it is important that the processor has enough delay before start the coefficient programming or enabling biquad (BIQ_CTL.EN). Attempting to program the coefficients before the auto programming is done will result in unsuccessful programming. The default coefficient setting is a low pass filter with 3db cut-off frequency 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 sample rate of BIQ can be calculated by equation HCLK/(SRDIV+1) = 16Khz. The output sample rate of BIQ in the ADC path further down-sampled by the BIQ_CTL.DPWMPUSR register so final sample rate presented to the FIFO is HCLK/(SRDIV+1)/(DPWMPUSR+1).

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*4 = 64Khz.

8.6.3 Functional Description

The BIQ filter is in reset state in default. To use the BIQ function, the following sequence is recommended:

- Set BIQ_CTL.DLCOEFF bit. By releasing the reset, the filter controller will download default coefficients automatically to the RAM.
- Turn on the BIQ_CTL.PRGCOEFF bit if intending to change the coefficients. Otherwise skip to next step.
- Setup the BIQ operation sample rate by program DPWMPUSR or SRDIV register bits if necessary.

- 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).
- Turn on BIQ_CTL.EN. BIQ will start filter function.

8.6.4 Register Map

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

Register	Offset	R/W	Description	Reset Value
BIQ Base Address: **BIQ_BA = 0x400B_0000**				
BIQ_COEFF0	BIQ_BA + 0x00	R/W	Coefficient b0 In H(z) Transfer Function (3.16 format) - 1st stage BIQ Coefficients	0x0000_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

8.6.5 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

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	
[31:29]	**Reserved**	Reserved.
[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.
[15:7]	**Reserved**	Reserved.
[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

31	30	29	28	27	26	25	24
COEFFDAT[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
COEFFDAT[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
COEFFDAT[15:8]							

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

Bits	Description	
[31:0]	COEFFDAT	Coefficient Data

9 APPLICATION DIAGRAM

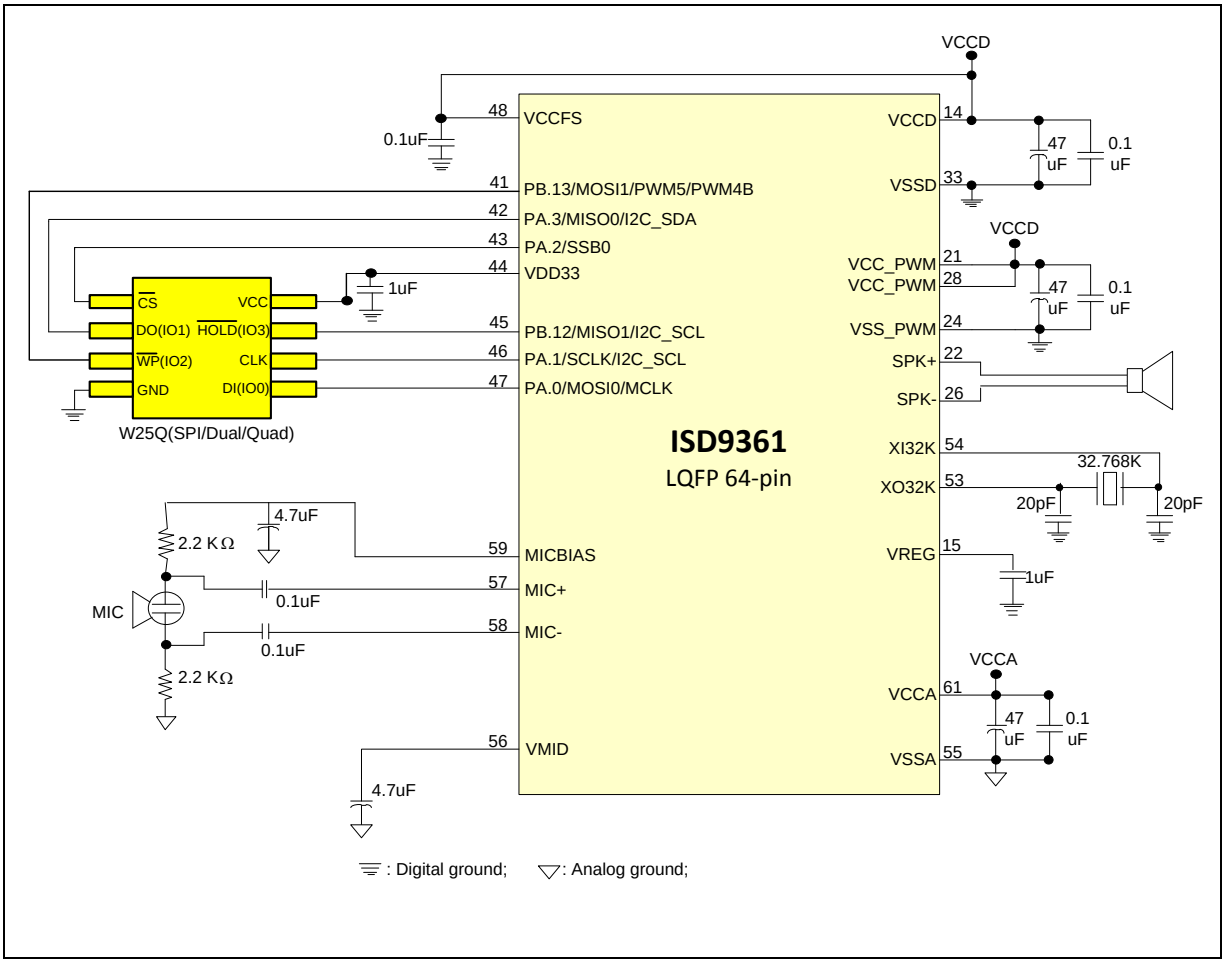


Figure 9-1 Application Diagram

10 ELECTRICAL CHARACTERISTICS

10.1 Absolute Maximum Ratings

SYMBOL	PARAMETER	MIN	MAX	UNIT
DC Power Supply	VDD–VSS	-0.3	+6.0	V
Input Voltage	VIN	VSS-0.3	VDD+0.3	V
Oscillator Frequency	$1/t_{CLCL}$	0	40	MHz
Operating Temperature	TA	-40	+85	°C
Storage Temperature	TST	-55	+150	°C
Maximum Current into V_{DD}		-	120	mA
Maximum Current out of V_{SS}			120	mA
Maximum Current sunk by a I/O pin			35	mA
Maximum Current sourced by a I/O pin			35	mA
Maximum Current sunk by total I/O pins			100	mA
Maximum Current sourced by total I/O pins			100	mA

Note: Exposure to conditions beyond those listed under absolute maximum ratings may adversely affects the lift and reliability of the device. Functional operation is not implied at these conditions.

Table 10-1 Absolute Maximum Ratings

10.2 DC Electrical Characteristics

(VDD-VSS=3.3V, TA = 25°C, F_{OSC} = 49.152 MHz unless otherwise specified.)

PARAMETER	SYM.	SPECIFICATION				TEST CONDITIONS
		MIN.	TYP.	MAX.	UNIT	
Operation voltage	V_{DD}	2.4		5.5	V	V_{DD} =2.4V ~ 5.5V up to 100 MHz
Power Ground	V_{SS} AV_{SS}	-0.3			V	
Analog Operating Voltage	AV_{DD}	0		V_{DD}	V	
DPWM Speaker Voltage	V_{DDSPK}	2.4		5.5	V	
Operating Current Normal Run Mode @ 49.152 MHz	I_{DD}				mA	V_{DD}= 5.5V, Enable all IP.
	I_{DD}				mA	V_{DD}=5.5V, disable all IP
	I_{DD}		21.1		mA	V_{DD} = 3V, enable all IP (analog and digital)
	I_{DD}		9.58		mA	V_{DD} = 3V, disable all IP
	I_{SLEEP}		4.9		mA	V_{DD} = 3V, disable all IP, M0 Sleep (WFI)
Operating Current Normal Run Mode @ 32.768Mhz	I_{DD9}				mA	V_{DD} = 5.5V, Enable all IP.
	I_{DD10}				mA	V_{DD} = 5.5V, Disable all IP.
	I_{DD11}				mA	V_{DD} = 3V, Enable all IP.
	I_{DD12}				mA	V_{DD} = 3V, Disable all IP.
Operating Current Normal Run Mode @ 72.728MHz	I_{DD}				mA	V_{DD}= 5.5V, Enable all IP.
	I_{DD}				mA	V_{DD}=5.5V, disable all IP
	I_{DD}		-		mA	V_{DD} = 3V, enable all IP (analog and digital)
	I_{DD}		-		mA	V_{DD} = 3V, disable all IP

PARAMETER	SYM.	SPECIFICATION				TEST CONDITIONS
		MIN.	TYP.	MAX.	UNIT	
	I_{SLEEP}		6		mA	V_{DD} = 3V, disable all IP, M0 Sleep (WFI)

Operating Current Normal Run Mode @ 98MHz	I_{DD}				mA	V_{DD}= 5.5V, Enable all IP.
	I_{DD}				mA	V_{DD}=5.5V, disable all IP
	I_{DD}		34.1		mA	V_{DD} = 3V, enable all IP (analog and digital)
	I_{DD}		17.6		mA	V_{DD} = 3V, disable all IP
	I_{SLEEP}		7.9		mA	V_{DD} = 3V, disable all IP, M0 Sleep (WFI)
Operating Current Sleep Mode	I_{IDLE1}				mA	V_{DD}= 5.5V
	I_{IDLE1}		12.5		mA	V_{DD}= 3.3V, enable IP: RTC, I2S, ANA, SBRAM, LDO, WDG, XTL32, OSC16KHz
Operating Current Deep Sleep Mode	I_{IDLE1}				mA	V_{DD}=5.5V,
	I_{IDLE1}		7.2		mA	V_{DD}= 3.3V, enable IP: RTC, I2S, ANA, SBRAM, LDO, WDG, XTL32, OSC16KHz
Operating Current Deep STOP Mode	I_{IDLE1}				uA	V_{DD}=5.5V
	I_{IDLE1}		0.06		mA	V_{DD}= 3.3V
Standby Power down mode(SPD)	I_{IDLE1}		5.7		uA	V_{DD}=3.3V 32K running with RTC
	I_{IDLE1}		2.5		uA	V_{DD}= 3.3V 16K running
Operating Current Deep Power down mode(DPD)	I_{IDLE1}		500		nA	V_{DD}=3.3V Wakeup with16K
	I_{IDLE1}		140		nA	V_{DD}= 3.3V wakeup with wakeup pin

Input Current PA, PB (Quasi-bidirectional mode)	I_{IN1}	-60	-	+15	µA	V_{DD} = 5.5V, V_{IN} = 0V or V_{IN}=V_{DD}
Input Current at /RESET [1]	I_{IN2}	-55	-45	-30	µA	V_{DD} = 3.3V, V_{IN} = 0.45V
Input Leakage Current PA, PB	I_{LK}	-2	-	+2	µA	V_{DD} = 5.5V, 0<V_{IN}<V_{DD}

Logic 1 to 0 Transition Current PA~PB(Quasi-bidirectional mode)	I_{TL} [3]	-650	-	-200	μA	V_{DD} = 5.5V, V_{IN}<2.0V
Input Low Voltage PA, PB(TTL input)	V_{IL1}	-0.3	-	0.8	V	V_{DD} = 4.5V
		-0.3	-	0.6		V_{DD} = 2.5V
Input High Voltage PA, PB (TTL input)	V_{IH1}	2.0	-	V_{DD} +0.2	V	V_{DD} = 5.5V
		1.51	-	V_{DD} +0.2		V_{DD} =3.0V
Input Low Voltage XT1[*2]	V_{IL3}	0	-	0.8	V	V_{DD} = 4.5V
		0	-	0.4		V_{DD} = 3.0V
Input High Voltage XT1[*2]	V_{IH3}	3.5	-	V_{DD} +0.2	V	V_{DD} = 5.5V
		2.4	-	V_{DD} +0.2		V_{DD} = 3.0V
Input Low Voltage X32I[*2]	V_{IL4}	0	-	0.4	V	
Input High Voltage X32I[*2]	V_{IH4}	1.7		2.5	V	
Negative going threshold (Schmitt input), /REST	V_{ILS}	-0.5	-	$0.3V_{DD}$	V	
Positive going threshold (Schmitt input), /REST	V_{IHS}	$0.7V_{DD}$	-	V_{DD}+0.5	V	
Hysteresis voltage of PA~PB(Schmitt input)	V_{HY}		$0.2V_{DD}$		V	

Source Current PA, PB Quasi-bidirectional Mode)	I_{SR11}	-300	-340	-450	μA	V_{DD} = 4.5V, V_S = 2.4V
	I_{SR12}	-50	-65	-90	μA	V_{DD} = 2.7V, V_S = 2.2V
	I_{SR12}	-40	-55	-80	μA	V_{DD} = 2.5V, V_S = 2.0V
Source Current PA, PB (Push-pull Mode)	I_{SR21}	-20	-22	-28	mA	V_{DD} = 4.5V, V_S = 2.4V
	I_{SR22}	-3.5	-6	-8	mA	V_{DD} = 2.7V, V_S = 2.2V
	I_{SR22}	-3	-5	-7	mA	V_{DD} = 2.5V, V_S = 2.0V
Sink Current PA, PB (Quasi-bidirectional and Push-pull Mode)	I_{SK1}	10	12	20	mA	V_{DD} = 4.5V, V_S = 0.45V
	I_{SK1}	7	9	13	mA	V_{DD} = 2.7V, V_S = 0.45V
	I_{SK1}	6	7	12	mA	V_{DD} = 2.5V, V_S = 0.45V

Notes:

1. /REST pin is a Schmitt trigger input.
2. Crystal Input is a CMOS input
3. Pins of P0, P1, P2, P3 and P4 can source a transition current when they are being externally driven from 1 to 0. In the condition of V_{DD}=5.5V, 5he transition current reaches its maximum value when Vin approximates to 2V.

Table 10-2 DC Electrical Characteristics

10.2.1 Special Characteristic

Isb test Conditions:

- VCCD = 3.3V, VCCA = 3.3V, T_A = +25°C,
- Running mode = SPD/DPD, clock source = internal 49.152 MHz oscillator OR internal low power 16KHz oscillator, RTC source = External 32KHZ crystal.

PARAMETER	SYM.	SPECIFICATION				TEST CONDITIONS
		MIN.	TYP.	MAX.	UNIT	
Deep Power Down (DPD) current	I_{DD}	-	0.14	-	uA	Wake up by Wakeup pin
	I_{DD}	-	7.40	-	uA	Wake up by RTC alarm*
	I_{DD}	-	-	-	uA	Wake up by OSC 16KHz
Standby Power Down (SPD) current	I_{DD}	-	0.71	-	uA	Wake up by GPIO pin
	I_{DD}	-	0.70	-	uA	Wake up by POR
	I_{DD}	-	0.72	-	uA	Wake up by RTC Alarm
	I_{DD}	-	-	-	uA	Wake up by RTC tick

* Not all IP disabled.

10.2.2 Operating Current Curve (Test condition: run NOP)

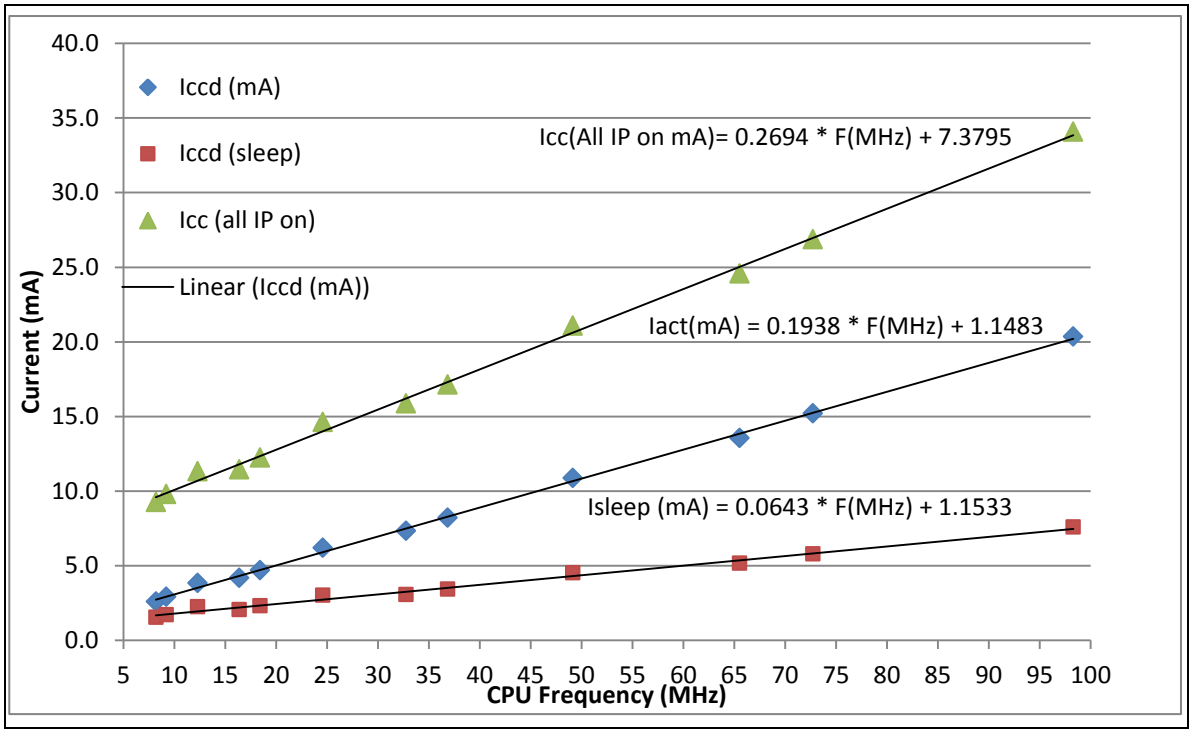


Figure 10-1 Operating Current Curve

10.3 AC Electrical Characteristics

10.3.1 External 32kHz XTAL Oscillator

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Input clock frequency	External crystal	-	32.768	-	kHz
Temperature	-	-40	-	85	℃
V_{DD}	-	2.4	-	5.5	V

Table 10-3 External 32kHz XTAL Oscillator

10.3.2 Internal 49.152MHz Oscillator

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Supply voltage	-	2.4	-	5.5	V
Center Frequency	OSCFsel=0 OSCFsel=1 OSCFsel=2	-	49.152 32.768 36.864		MHz
Calibrated Internal Oscillator Frequency	+25□C; V_{DD} =3V	-0.5	-	0.5	%
	-40□C~+85□C; V_{DD}=2.4V~5.5V	-2.0	-	1.0	%

Table 10-4 Internal 49.152MHz Oscillator

10.3.3 Internal 16kHz Oscillator

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Supply voltage	-	2.4	-	5.5	V
Center Frequency	-	-	16	-	kHz
Calibrated Internal Oscillator Frequency	+25□C; V_{DD} =3V	-10	-	10	%
	-40□C~+85□C; V_{DD}=2.4V~5.5V	-20	-	20	%

Table 10-5 Internal 16 KHz Oscillator

10.4 Analog Characteristics

10.4.1 Specification of ADC and Speaker Driver

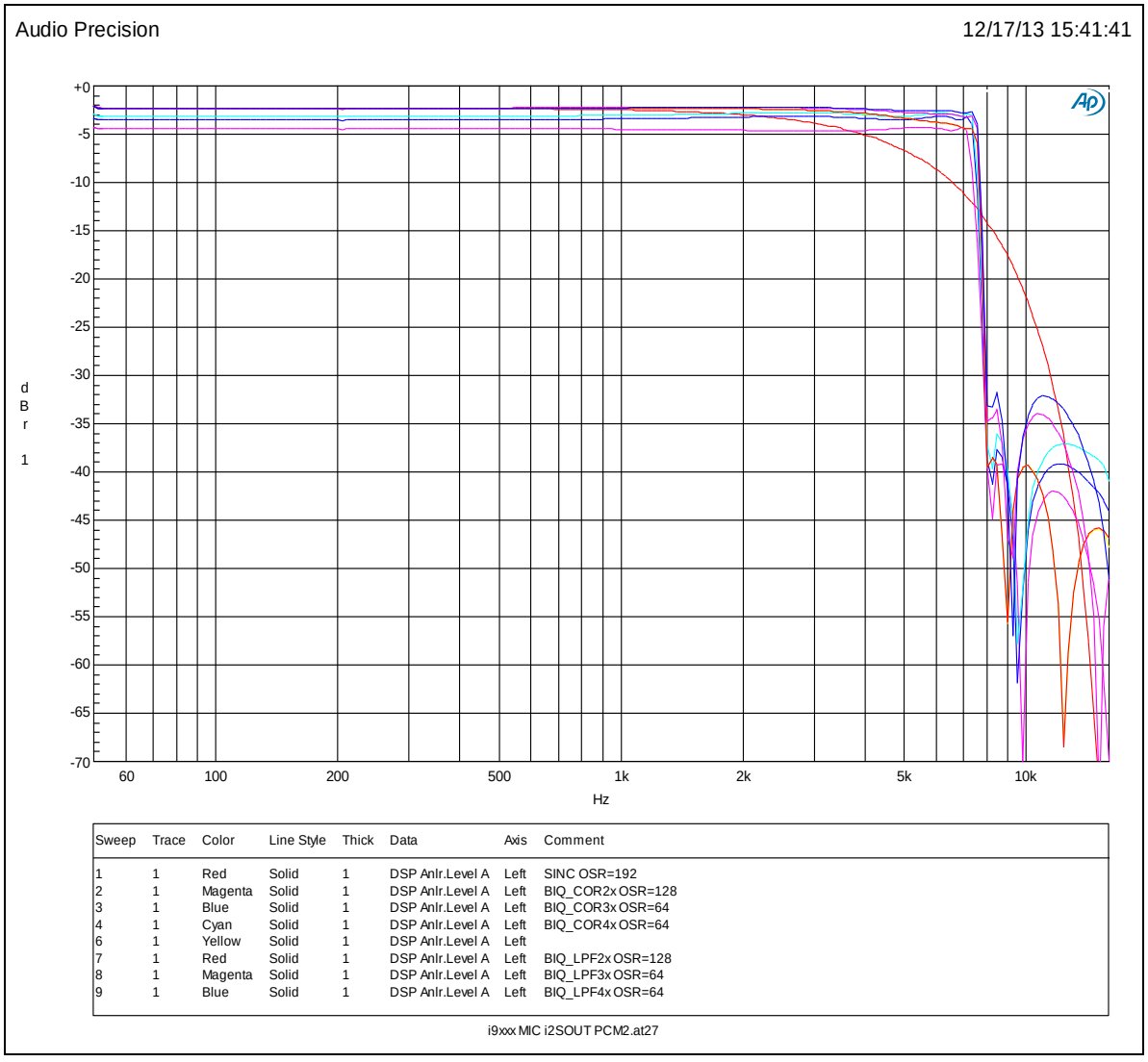
Conditions: VCCD = 3.3V, VCCA = 3.3V, T_A = +25°C, 1kHz signal, fs = 16kHz, 16-bit audio data, unless otherwise stated.

Parameter	Symbol	Comments/Conditions	Min	Typ	Max	Units
Analog to Digital Converter (ADC)						
Full scale input signal [1]	V_{INFS}	PGABST = 0dB PGAGAIN = 0dB		1.0 0		Vrms dBV
Signal-to-noise ratio	SNR	Gain = 0dB, A-weighted	tbd	90		dB
Total harmonic distortion [2]	THD+N	Input = -3dB FS input		-80	tbd	dB
PWM Speaker Output (8Ω bridge-tied-load)						
Full scale output [4]			VCCSPK / 3.3			V_{rms}
Total harmonic distortion [2]	THD+N	P_o= 200mW, VDDSPK=3.3V		*63		dB
		P_o= 320mW, VDDSPK = 3.3V		-64		dB
		P_o= 860mW, VDDSPK = 5V		-60		dB
		P_o= 1000mW, VDDSPK = 5V		-36		dB
Signal-to-noise ratio	SNR	VDDSPK = 3.3V		91		dB
		VDDSPK=5V		90		dB
Power supply rejection ratio (50Hz - 22kHz)	PSRR	VDDSPK = 3.3V				dB
		VDDSPK = 5V				dB

Table 10-6 Specification of ADC and Speaker Driver

10.4.2 ADC Filter Characteristics

Below are responses of ADC with and without various biquaddownsample filters for 16kHz sample rate. The biquad correction filters compensate for SINC filter droop with greater passband ripple while the LPF filters are maximally flat but roll off with SINC response as can be seen in the passband figure. Filter coefficients used for these results are recorded below.



Sweep	Trace	Color	Line Style	Thick	Data	Axis	Comment
1	1	Red	Solid	1	DSP Anlr.Level A	Left	SINC OSR=192
2	1	Magenta	Solid	1	DSP Anlr.Level A	Left	BIQ_COR2x OSR=128
3	1	Blue	Solid	1	DSP Anlr.Level A	Left	BIQ_COR3x OSR=64
4	1	Cyan	Solid	1	DSP Anlr.Level A	Left	BIQ_COR4x OSR=64
6	1	Yellow	Solid	1	DSP Anlr.Level A	Left	
7	1	Red	Solid	1	DSP Anlr.Level A	Left	BIQ_LPF2x OSR=128
8	1	Magenta	Solid	1	DSP Anlr.Level A	Left	BIQ_LPF3x OSR=64
9	1	Blue	Solid	1	DSP Anlr.Level A	Left	BIQ_LPF4x OSR=64

i9xxx MIC i2SOUT PCM2.at27

Figure 10-2 ADC Filter Characteristics

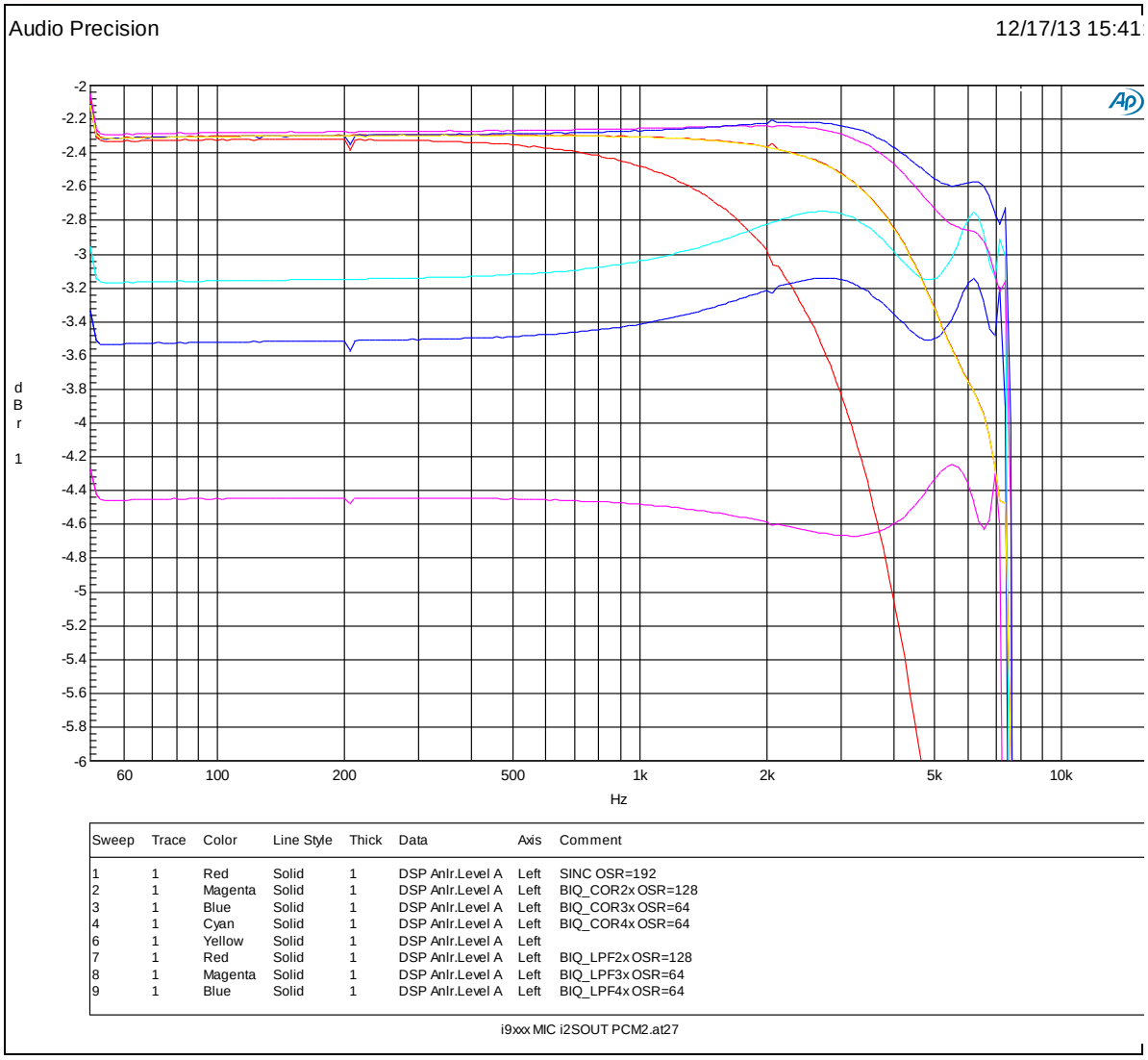


Figure 10-3 ADC Filter Characteristics Zoom In

Filter Coefficients

const int32_t biq_lpf_2x[15] = { 13084, 20028, 13084, -31664, 12324, 56921, 1628, 56921, -10493, 60919, 38661, 13816, 38661, -17191, 43987, };;

const int32_t biq_lpf_3x[15] = { 8635, 6870, 8635, -62680, 21284, 56738, -55661, 56738, -69684, 62157, 36743, -26956, 36743, -67027, 48671, };;

const int32_t biq_lpf_4x[15] = { 6957, 1062, 6957, -78840, 28280, 56826, -79674, 56826, -94372, 62919, 36484, -45099, 36484, -89271, 52002, };;

const int32_t biq_correction_2x[15] = {-19134, -13071, -19129, -31679, 35130, 30257, 1615, 30255, -18774, 58716, 42682, 41327, 250, -17473, 25, };;

const int32_t biq_correction_3x[15] = {7815, 7795, 125, -78206, 28409, 48850, -47621, 48849, -72457, 61930, 27970, -17354, 27959, -73295, 48957, };;

const int32_t biq_correction_4x[15] = {4742, 4643, 88, -91540, 35482, 15833, -18582, 15831, -95858, 62902, 91737, -128342, 91736, -93424, 52730, };;

10.4.3 Specification of PGA and BOOST

Conditions: VCCD = 3.3V, VCCA = 3.3V, T_A = +25°C, 1kHz signal, fs = 16kHz, 16-bit audio data, unless otherwise stated.

Parameter	Symbol	Comments/Conditions	Min	Typ	Max	Units
Microphone Inputs (MICP, MICN) and Programmable Gain Amplifier (PGA)						
Full scale input signal [1]		PGABST = 0dB PGAGAIN = 0dB		1.0 0		Vrms dBV
Programmable gain			-12		35.25	dB
Programmable gain step size		Guaranteed Monotonic		0.75		dB
Mute Attenuation				120		dB
Input resistance		Inverting Input PGA Gain = 35.25dB PGA Gain = 0dB PGA Gain = -12dB **Non-inverting Input**		 1.6 47 75 94		 kΩ kΩ kΩ kΩ
Input capacitance				10		pF
PGA equivalent input noise		0 to 20kHz, Gain set to 35.25dB		120		μV
Input Boost						
Gain boost		Boost disabled Boost enabled		0 26		dB dB

Table 10-7 Specification of PGA and BOOST

10.4.4 Specification of ALC an MICBIAS

Conditions: VCCD = 3.3V, VCCA = 3.3V, T_A = +25°C, 1kHz signal, fs = 16kHz, 16-bit audio data, unless otherwise stated.

Parameter	Symbol	Comments/Conditions	Min	Typ	Max	Units
Automatic Level Control (ALC) & Limiter:						
Target record level			-22.5		-1.5	dBFS
Programmable gain			-12		35.25	dB
Gain hold time [3]	t_{HOLD}	Doubles every gain step, with 16 steps total	0 / 2.67 / 5.33 / … / 43691			ms
Gain ramp-up (decay) [3]	t_{DCY}	ALC Mode ALC = 0	4 / 8 / 16 / … / 4096			ms
		Limiter Mode ALC = 1	1 / 2 / 4 / … / 1024			ms
Gain ramp-down (attack) [3]	t_{ATK}	ALC Mode ALC = 0	1 / 2 / 4 / … / 1024			ms
		Limiter Mode ALC = 1	0.25 / 0.5 / 1 / … / 128			ms
Mute Attenuation				120		dB
Microphone Bias						
Bias voltage	$V_{MICBIAS}$		0.90, 0.65 ,0.75, 0.50, 2.4, 1.7,2.0			VDDA V
Bias current source	$I_{MICBIAS}$			3		mA
Output noise voltage	V_n	1kHz to 20kHz		14		nV/√Hz

Notes

1. Full Scale is relative to the magnitude of VCCA and can be calculated as FS = VDDA/3.3.
2. Distortion is measured in the standard way as the combined quantity of distortion products plus noise. The signal level for distortion measurements is at 3dB below full scale, unless otherwise noted.
3. Time values scale proportionally with HCLK. Complete descriptions and definitions for these values are contained in the detailed descriptions of the ALC functionality.

Table 10-8 Specification of ALC an MICBIAS

10.4.5 Specification of LDO & Power management

PARAMETER	MIN	TYP	MAX	UNIT	NOTE
Input Voltage	2.4	5	5.5	V	V_{DD} input voltage
Output Voltage	-10%	1.8	+10%	V	V_{DD}> 1.8

Notes

1. It is recommended that a 10uF or higher capacitor and a 100nF bypass capacitor are connected between VCCD and the VSSD pin of the device.
2. To ensure regulator stability, a 1.0uF capacitor must be connected between LDO pin and the VSSD pin of the device. Also a 100nF bypass capacitor between LDO and VSSD will help suppress output noise.

Table 10-9 Specification of LDO & Power management

10.4.6 Specification of Brownout Detector

PARAMETER	CONDITION		MIN.	TYP.	MAX.	UNIT
Operation voltage	-		2.2	-	5.5	V
Quiescent current	AVDD=5.5V		-	-	125	μA
Temperature	-		-40	25	85	℃
Brown-out voltage	Range=0	BOV_VL[2:0]=000		2.1		V
		BOV_VL [2:0]=001		2.2		V
		BOV_VL [2:0]=010		2.4		V
		BOV_VL[2:0]=011		2.5		V
		BOV_VL [2:0]=100		2.65		V
		BOV_VL[2:0]=101		2.8		V
		BOV_VL [2:0]=110		3.0		V
		BOV_VL [2:0]=111		4.6		V
	Range=1	BOV_VL[2:0]=000		3.1		V
		BOV_VL [2:0]=001		3.2		V
		BOV_VL [2:0]=010		3.4		V
		BOV_VL[2:0]=011		3.6		V
		BOV_VL [2:0]=100		3.7		V
		BOV_VL[2:0]=101		3.8		V
		BOV_VL [2:0]=110		3.9		V
		BOV_VL [2:0]=111		4.2		V
Hysteresis				_		V

Table 10-10 Specification of Brownout Detector

10.4.7 Specification of Power-On Reset (VCCD)

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Temperature	-	-40	25	85	℃
Reset voltage	VCC ramping down	-	1.0	-	V
Reset Release voltage	VCC ramping up		1.5		V
Quiescent current	Vin>reset voltage	-	60	-	nA

Table 10-11 Specification of Power-On Reset (VCCD)

10.4.8 Specification of Temperature Sensor

PARAMETER	MIN	TYP	MAX	UNIT	CONDITIONS
Supply voltage[1]	2.4	-	5.5	V	
Temperature	-40	-	125	℃	
Current consumption				uA	
Gain				mV/℃	
Offset				mV	Temp=0 ℃

Notes

1. Internal operation voltage comes from LDO.

Table 10-12 Specification of Temperature Sensor

10.4.9 Specification of Comparator

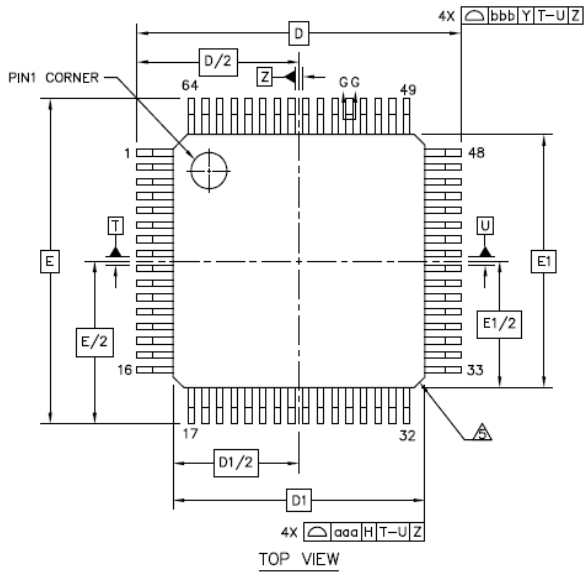
PARAMETER	MIN.	TYP.	MAX.	CONDITION
Temperature	-40℃	25 ℃	85℃	-
VCCA	2.4	3	5.5	-
VCCA current	-	20uA	40uA	20uA@VDD=3V
Input offset voltage	-	5mV	15mV	-
Input common mode range	0.1	-	VDD-1.2	-
DC gain	-	70dB	-	-
Propagation delay	-	200ns	-	@VCM=1.2V & VDIFF=0.1V

Comparison voltage	10mV	20mV	-	20mV@VCM=1V 50mV@VCM=0.1V 50mV@VCM=VDD-1.2 @10mV for non-hysteresis
Hysteresis	-	±10mV	-	One bit control W/O & W. hysteresis @VCM=0.4V ~ VDD-1.2V
Wake up time	-	-	2us	@CINP=1.3V CINN=1.2V

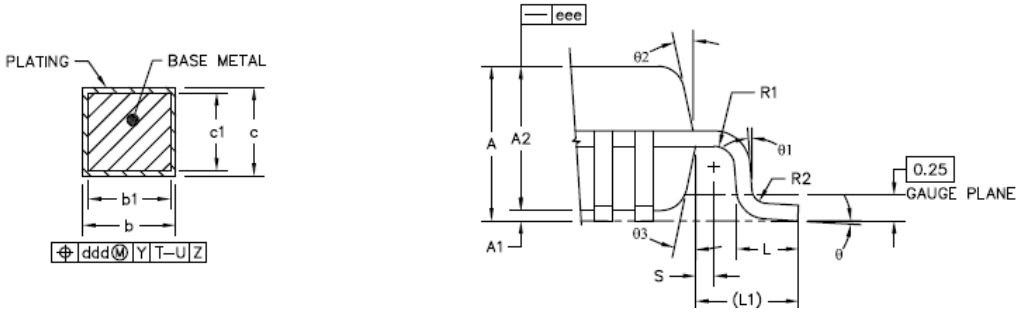
Table 10-13 Specification of Comparator

11 PACKAGE DIMENSIONS

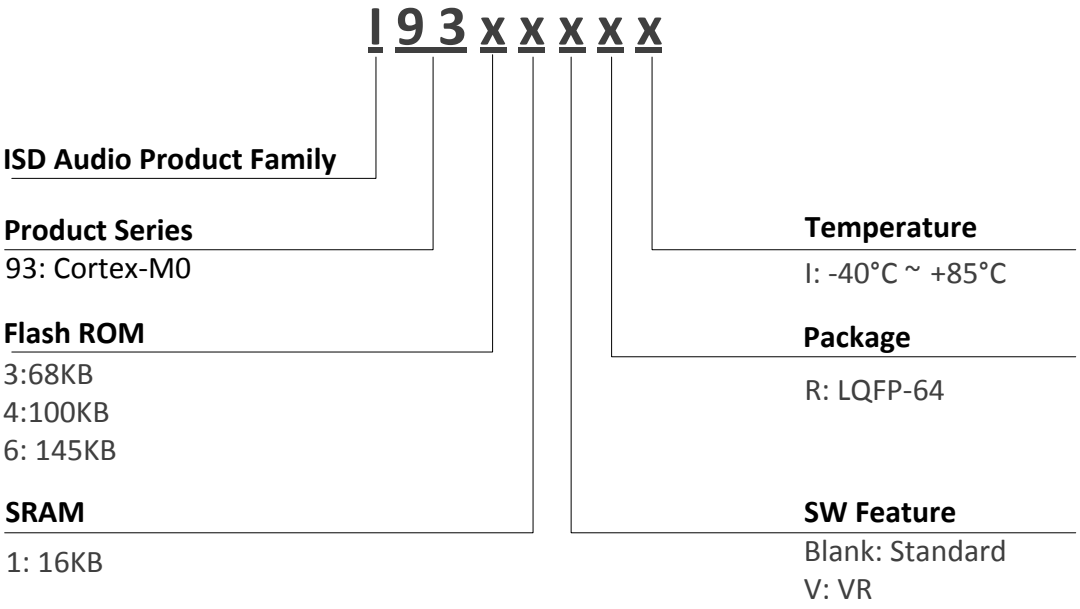
11.1 64L LQFP (7x7x1.4mm footprint 2.0mm)



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.13	0.18	0.23
LEAD WIDTH		b1	0.13	0.16	0.19
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
BODY SIZE	X	D	9 BSC		
	Y	E	9 BSC		
	X	D1	7 BSC		
	Y	E1	7 BSC		
LEAD PITCH		e	0.4 BSC		
		L	0.45	0.6	0.75
FOOTPRINT		L1	1 REF		
		θ	0°	3.5°	7°
		θ1	0°	---	---
		θ2	11°	12°	13°
		θ3	11°	12°	13°
		R1	0.08	---	---
		R2	0.08	---	0.2
		S	0.2	---	---
PACKAGE EDGE TOLERANCE		aaa	0.2		
LEAD EDGE TOLERANCE		bbb	0.2		
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.07		
MOLD FLATNESS		eee	0.05		



12 ORDERING INFORMATION



13 EVISION HISTORY

REVISION	DATE	DESCRIPTION
1.10	Feb 12, 2015	Preliminary version to normal version. Modify text, figure's registers renaming and pwm channel names. Added I9331 and I9341 into the series. Flash base address description update. BOD, ADC clock description update.
1.12	Mar 17, 2016	Remove PSW VR option for ordering.