ISD9000 Series Audio SoC

I91032

Technical Reference Manual

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

www.nuvoton.com

Table of Contents

List of Tables

List of Figures

1. General Description

The I91032 is a new member of Nuvoton ARM® Cortex™-M0 Audio SoC family. The 1.8V to 5.5V operating voltage makes it an ideal fit for battery powered consumer products. The highly architecture - 32-bit ARM® Cortex™-M0 processor, wide operating voltage, speaker driver, embedded flash memory and multi-function GPIO – is designed to provide fast Time-to-Market and cost effective solution, enable audio feature to wide range of industrial and consumer products such as voice recognition and audio/voice feedback for portable medical devices, security systems, public transit vehicles, home appliances, and novelty items.

Note1: Please refer to datasheet for electrical related spec and typical applications.

Note2: DAC (5.13) is grayed because some issues are still under clarification.

2. Features

- Core
  - ARM® Cortex™-M0 core runs up to 50MHz
  - One 24-bit System tick timer for operating system support
  - Single-cycle 32-bit hardware multiplier.
  - NVIC (Nested Vector Interrupt Controller) for interrupt inputs, each with 4-levels of priority.
  - Serial Wire Debug (SWD) supports with 2 watch-points/4 breakpoints.
- Power Management
  - Wide operating voltage range from 1.8V to 5.5V (not include ADC/PGC)
  - Power Management Unit (PMU) provides different levels of power control.
  - Deep Power Down (DPD) mode with one byte specific register retention is the lowest power state.
  - Wakeup from DPD via specific WAKEUP pin (GPA4) or LIRC timed operation.
  - Standby Power Down (SPD/STOP) mode is the lowest power state with RAM retention (typically < 5uA), with LXT or LIRC oscillation (typically < 7uA).
  - Wakeup from SPD can be from any GPIO, RTC, or WDT interrupts.

  Flash EEPROM Memory
  - 64KB Flash EPROM for program code and data storage
  - Additional 4KB of flash can be configured as boot sector for ISP loader
  - Support ISP and ICP code update
  - 512-Byte page erase, 4-Byte word programming.
  - Configurable boundary to delineate code and data flash
  - Support 2 wire In-circuit Programming (ICP) update from SWD ICE interface
- SRAM Memory
  - 6KB embedded SRAM.
- Clock Control
  - Built-in high speed (HIRC, factory trimmed within +/-1% to 49.152MHz) and low speed (LIRC, 10KHz) oscillators, no external components necessary.
  - External 32K Hz crystal/resonator input (LXT).
- GPIO
  - Max. 22 GPIOs (including SWD, LXT) can be configured individually for below four I/O modes:
    - Quasi bi-direction
    - Push-Pull output
    - Open-Drain output
    - Input with optional pull-up
  - Schmitt trigger and slew rate control selectable by quad
  - Each I/O pin can be configured as interrupt/wake-up source with edge/level setting.
- ADC
  - 10-bit SAR ADC could be up to 12-bit output via configurable decimation filter with over sampling.

- Programmable gain amplifier with 64 steps from -18dB to 45dB in 1dB step size.
- DMA support for minimal CPU intervention.

- Differential Audio PWM Output (DPWM)
  - Direct connection of speaker
  - 0.5W drive capability into 8Ω load @5V
  - Configurable up-sampling to support sample rates from 8-32k Hz.
  - DMA support for minimal CPU intervention.
- 13-bit DAC
  - An alternative to DPWM for driving external PA.
- PDMA
  - 2-channel DMAs support data transfer between SRAM and peripherals of ADC, DPWM, SPI0 or SPIM.
- Timers
  - 3 timers with 8-bit pre-scaler and 16-bit counter.
  - Counter auto reload.
  - Timer1 can be IR carrier generator.
  - One fixed frequency timer.
- Watch Dog Timer
  - Multiple clock sources
  - 8 selectable time-out periods 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
  - Clock from LIRC 10KHz or LXT 32KHz
  - Selectable interrupt frequencies (0.25, 0.5, 1, 2, 4, 8, 16, 32 Hz based on LXT 32KHz)
  - Support wake up function
- Two sets of PWM/Capture
  - Built-in one 16-bit timer and four 16-bit comparators for 4 PWM outputs as a set.
  - 2 sets support up to 8 individual PWM outputs or up to 4 complementary paired outputs.
  - The PWM generator equipped with a clock source selector, a clock divider, an 8-bit pre-scalar 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 with flow control (TX, RX, CTS and RTS)
- SPI0

  - Master up to 24.576Mbps / Slave up to 12.288Mbps
  - Support MICROWIRE/SPI master/slave mode (SSP).
  - Full duplex synchronous serial data transfer
  - Variable length of transfer data from 1 to 32 bits
  - MSB or LSB first data transfer
  - 2 slave/device select lines when used in master mode
  - Two 32-bit buffers or DMA support for burst transfers.

- SPIM
  - Supports general SPI master interface protocol
  - 8-, 16-, 24-, and 32-bit length of transaction
  - Supports standard (1-bit), dual (2-bit), and quad (4-bit) I/O transfer mode
  - DMA support for burst transfers

- Brown-Out Detector (BOD)
  - With 16 levels: 1.8/1.9/2.0/2.1/2.2/2.4/2.6/2.8/3.0/3.1/3.4/3.6/3.7/.3.9/4.2/4.6V
  - Supports Brown-out Interrupt and Reset option.

- Low Voltage Reset: 1.6V

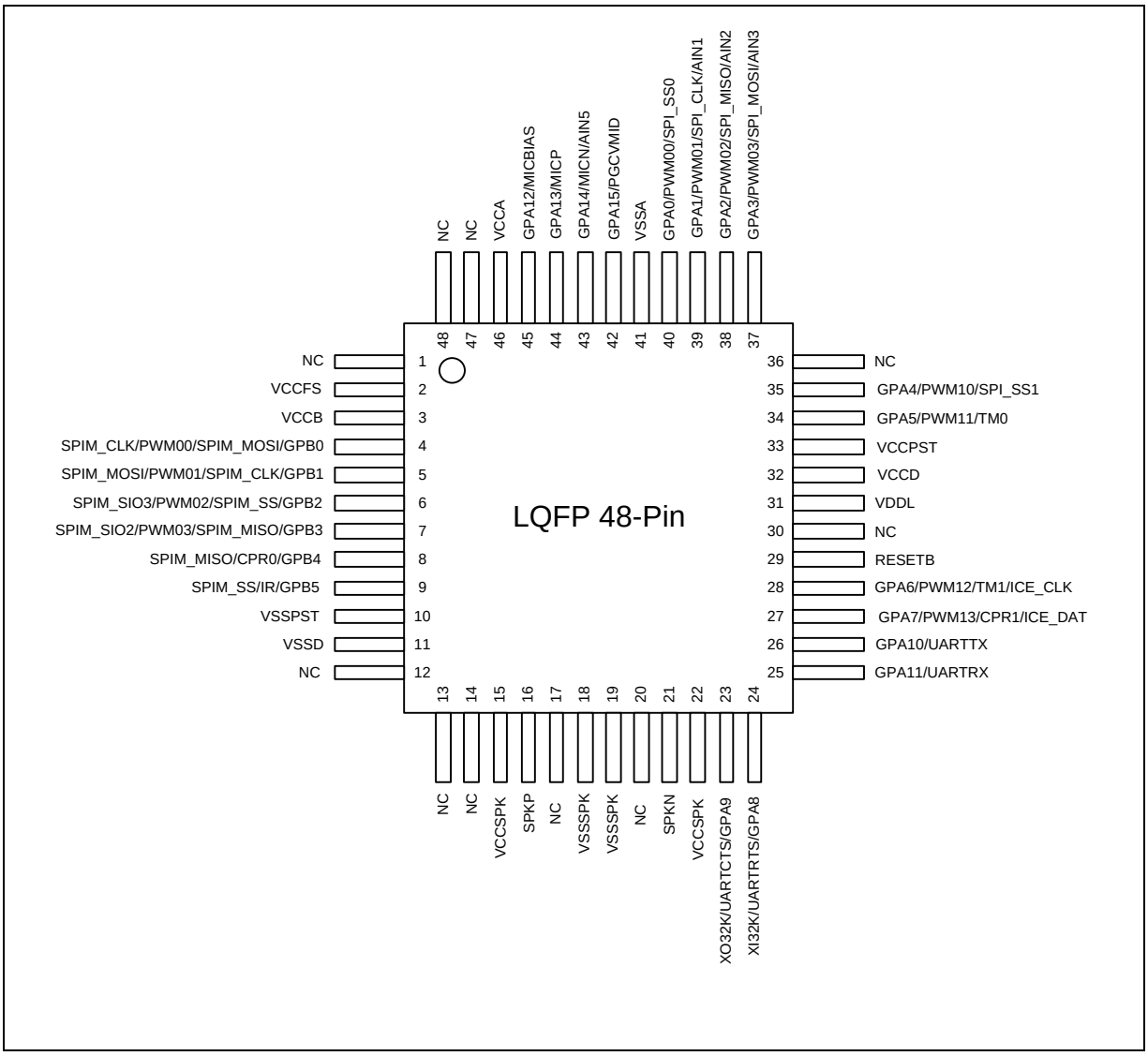
- Built-in Low Dropout Voltage Regulator (LDO)
  - Capable of delivering 25mA load current at 3.3V
  - Configurable output voltage: 8 options of 1.5V/1.7V/1.8V/2.4V/2.5V/2.7V/3V/3.3V
  - 1.5V LDO for core logic

- Operating Temperature: -40°C ~ 85°C

- Package:
  - Green package (RoHS)
  - LQFP 48-pin

3. Part Information and Pin Configuration

3.1 LQFP 48-Pin Diagram



3.2 Pin/Pad Description

LQFP48 Pin No.	Name	Type	Alt CFG	Description
1	NC			Remain unconnected
2	VCCFS	**P**		Power supply for the LDO of VCCB, normally connect to VCCD
3	VCCB	**P**		Power supply for GPB0~5 and driving external, could be VCCFS based LDO regulator output. If used, a 1µF capacitor must be placed to ground. If not used then tie to VCCD.
4	GPB0	**I/O**	0	General purpose input/output pin; port B, bit0
	SPIM_CLK	**O**	1	SPIM Serial Clock
	PWM00	**O**	2	PWM0 channel 0 output
	SPIM_MOSI	**O**	3	SPIM Master Out Slave In
5	GPB1	**I/O**	0	General purpose input/output pin; port B, bit1
	SPIM_MOSI	**I/O**	1	SPIM Master Out Slave In
	PWM01	**O**	2	PWM0 channel 1 output
	SPIM_CLK	**O**	3	SPIM Serial Clock
6	GPB2	**I/O**	0	General purpose input/output pin; port B, bit2
	SPIM_SIO3	**I/O**	1	SPIM Quad Mode Serial I/O 3
	PWM02	**O**	2	PWM0 channel 2 output
	SPIM_SS	**O**	3	SPIM Slave Select
7	GPB3	**I/O**	0	General purpose input/output pin, port B, bit3
	SPIM_SIO2	**I/O**	1	SPIM quad mode Serial I/O 2
	PWM03	**O**	2	PWM0 channel 3 output
	SPIM_MISO	**I**	3	SPIM Master In Slave Out
8	GPB4	**I/O**	0	General purpose input/output pin, port B, bit4
	SPIM_MISO	**I/O**	1	SPIM Master In Slave Out
	CPR0	**I**	2	Capture input based on PWM0 timer
9	GPB5	**I/O**	0	General purpose input/output pin, port B, bit5
	SPIM_SS	**O**	1	SPIM Slave Select
	IR	**O**	2	IR carrier generator based on Timer1 interval
10	VSSPST	**P**		Digital ground, connect to VSSD
11	VSSD	**P**		Digital ground

LQFP48 Pin No.	Name	Type	Alt CFG	Description
12	NC			Remain unconnected
13	NC			Remain unconnected
14	NC			Remain unconnected
15	VCCSPK	**P**		Power Supply for DPWM Speaker Driver
16	SPKP	**O**		Positive Speaker Driver Output
17	NC			Remain unconnected
18	VSSSPK	**P**		Ground for DPWM Speaker Driver
19				
20	NC			Remain unconnected
21	SPKN	**O**		Negative Speaker Driver Output
22	VCCSPK	**P**		Power Supply for DPWM Speaker Driver
23	GPA9	**I/O**	0	General purpose input/output pin; port A, bit9
	UARTCTS	**I**	2	UART Clear To Send Input
	XO32K	**O**	3	32K Crystal Oscillator Output
24	GPA8	**I/O**	0	General purpose input/output pin; port A, bit8
	UARTRTS	**O**	2	UART Request To Send Output
	XI32K	**I**	3	32K Crystal Oscillator Input
25	GPA11	**I/O**	0	General purpose input/output pin; port A, bit11
	UARTRX	**I**	2	UART Receiver Serial In
26	GPA10	**I/O**	0	General purpose input/output pin; port A, bit10
	UARTTX	**O**	2	UART Transmitter Serial Out
27	GPA7	**I/O**	0	General purpose input/output pin; port A, bit7
	PWM13	**O**	1	PWM1 channel 3 output
	CRP1	**I**	2	Capture input based on PWM1 timer
	ICE_DAT	**I/O**	X	SWD Interface, Serial Data
28	GPA6	**I/O**	0	General purpose input/output pin; port A, bit6
	PWM12	**O**	1	PWM1 channel 2 output
	TM1	**I**	2	Timer1 external clock input

LQFP48 Pin No.	Name	Type	Alt CFG	Description
	ICE_CLK	**I**	X	SWD Interface, Serial Clock
29	RESETB	**I**		Reset input, low active, internal pull-high
30	NC			Remain unconnected
31	VDDL	**P**		Regulator output decoupling pin for core logic. A 1uF cap returning to VSSD must be placed on it..
32	VCCD	**P**		Main Digital Power Supply
33	VCCPST	**P**		Digital power, connect to VCCD
34	GPA5	**I/O**	0	General purpose input/output pin; port A, bit5
	PWM11	**O**	1	PWM1 channel 1 output
	TM0	**I**	2	Timer0 external clock input
35	GPA4	**I/O**	0	General purpose input/output pin; port A, bit4
	PWM10	**O**	1	PWM1 channel 0 output
	SPI_SS1	**O**	2	SPI0 Slave Select 1
36	NC			Remain unconnected
37	GPA3 (AIN3)	**I/O**	0	General purpose input/output pin; port A, bit3 ADC input channel 3
	PWM03	**O**	1	PWM0 channel 3 output
	SPI_MOSI	**I/O**	2	SPI0 Master Out Slave In
38	GPA2 (AIN2)	**AIO**	0	General purpose input/output pin; port A, bit2 ADC input channel 2
	PWM02	**O**	1	PWM0 channel 2 output
	SPI_MISO	**I/O**	2	SPI0 Master In Slave Out
39	GPA1 (AIN1)	**I/O**	0	General purpose input/output pin; port A, bit1 ADC input channel 1
	PWM01	**O**	1	PWM0 channel 1 output
	SPI_CLK	**I/O**	2	SPI0 Serial Clock
40	GPA0	**I/O**	0	General purpose input/output pin; port A, bit0
	PWM00	**O**	1	PWM0 channel 0 output
	SPI_SS0	**I/O**	2	SPI0 Slave Select 0
41	VSSA	**AP**		Ground for Analog Circuitry
42	GPA15	**I/O**	0	General purpose input/output pin; port A, bit15

LQFP48 Pin No.	Name	Type	Alt CFG	Description
	PGCVMID	**AO**	1	Mid Rail Reference, Connect 4.7uF to VSSA
43	GPA14	**I/O**	0	General purpose input/output pin; port A, bit14
	MICN	**AI**	1	Negative Microphone Input
	AIN5	**AI**	2	ADC input channel 5
44	GPA13	**I/O**	0	General purpose input/output pin; port A, bit13
	MICP	**AI**	1	Positive Microphone Input
45	GPA12	**I/O**	0	General purpose input/output pin; port A, bit12
	MICBIAS	**AO**	1	Microphone Bias Output
46	VCCA	**AP**		Power Supply for Analog Circuitry
47	NC			Remain unconnected
48	NC			Remain unconnected

Pin/Pad Type: I=Digital Input, O=Digital Output, AI=Analog Input, AO=Analog Output, P=Power, AP=Analog Power

3.3 Alternative Function List of GPIO

GPIO	Power	ALT=1		ALT=2		ALT=3		Special
		Name	I/O type	Name	I/O type	Name	I/O type	
GPA0	VCCD	PWM00	O	SPI_SS0	I/O	-		
GPA1	VCCD	PWM01	O	SPI_CLK	I/O	-		AIN1
GPA2	VCCD	PWM02	O	SPI_MISO	I/O	-		AIN2
GPA3	VCCD	PWM03	O	SPI_MOSI	I/O	-		AIN3
GPA4	VCCD	PWM10	O	SPI_SS1	O	-		
GPA5	VCCD	PWM11	O	TM0	I	-		
GPA6	VCCD	PWM12	O	TM1	I	-		ICE_CLK
GPA7	VCCD	PWM13	O	CPR1	I	-		ICE_DAT
GPA8	VCCD	-		UARTRTS	O	XI32K	I	
GPA9	VCCD	-		UARTCTS	I	XO32K	O	
GPA10	VCCD	-		UARTTX	O	-		
GPA11	VCCD	-		UARTRX	I	-		
GPA12	VCCA	MICBIAS	AO	-		-		
GPA13	VCCA	MICP	AI	-		-		
GPA14	VCCA	MICN	AI	AIN5	AI	-		
GPA15	VCCA	PGCVMID	AO	-		-		
GPB0	VCCB	SPIM_CLK	O	PWM00	O	SPIM_MOSI	I/O	
GPB1	VCCB	SPIM_MOSI	I/O	PWM01	O	SPIM_CLK	O	
GPB2	VCCB	SPIM_SIO3	I/O	PWM02	O	SPIM_SS	O	
GPB3	VCCB	SPIM_SIO2	I/O	PWM03	O	SPIM_MISO	I/O	
GPB4	VCCB	SPIM_MISO	I/O	CPR0	I	-		
GPB5	VCCB	SPIM_SS	O	IR	O	-		

1. ICE_CLK and ICE_DAT are default setting and have specific control.
2. AIN1, AIN2 and AIN3 are ADC inputs, select GPIO with input mode and disable pull-up option.

4. Block Diagram

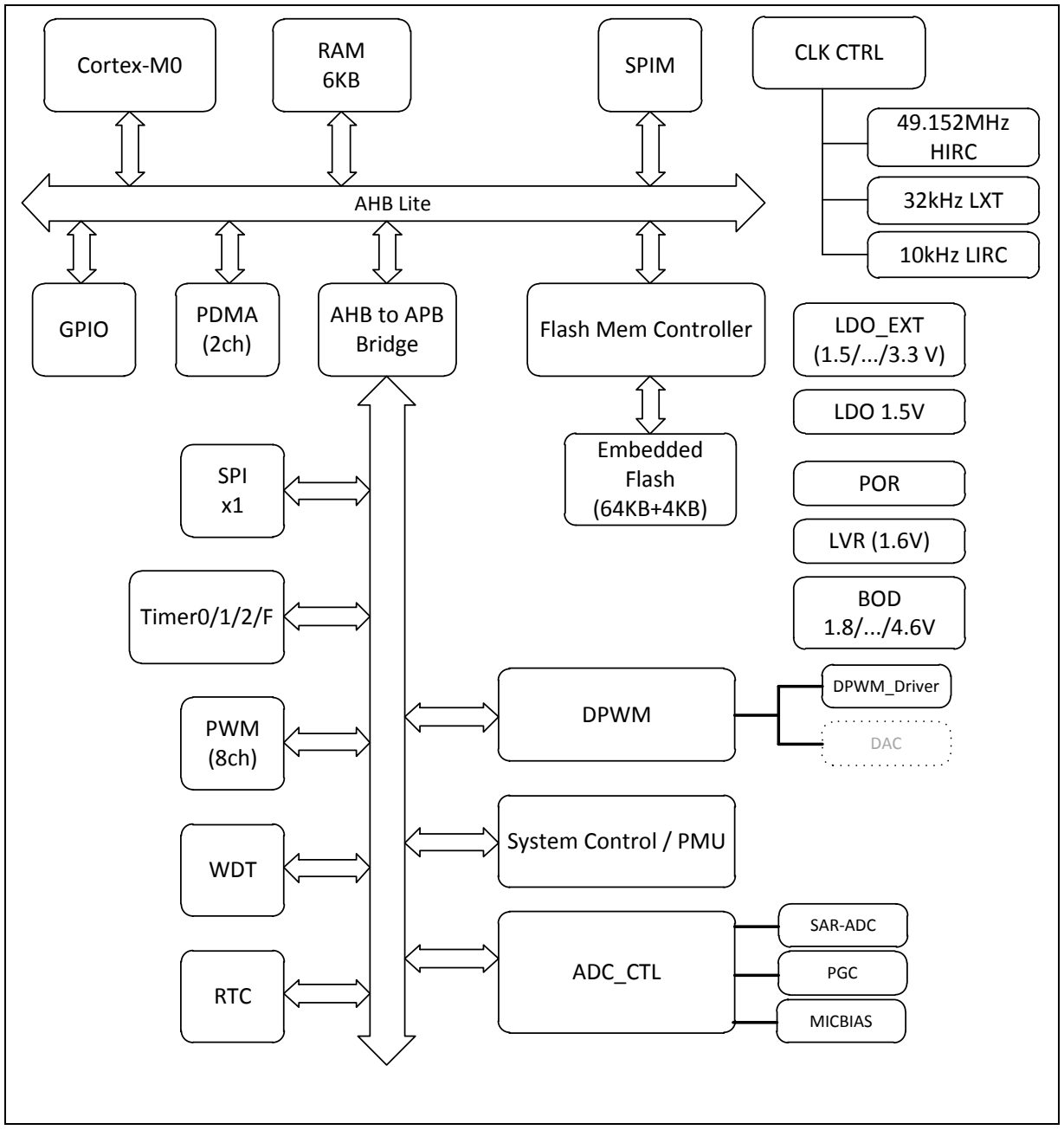


Figure 4-1 Functional Block Diagram

5. Functional Description

5.1 ARM® Cortex® -M0 Core

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

Figure 5-1 shows the functional blocks of processor.

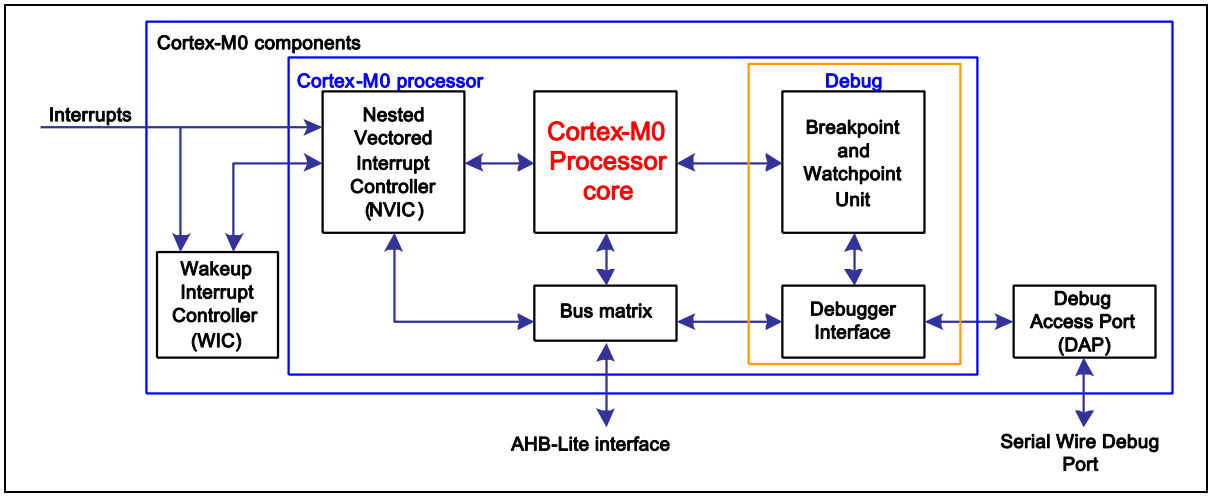


Figure 5-1 Functional Block Diagram

The implemented device provides:

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

- NVIC features
  - 32 external interrupt inputs, each with four levels of priority.
  - Dedicated non-Maskable Interrupt (NMI) input.
  - Support for both level-sensitive and pulse-sensitive interrupt lines
  - Wake-up Interrupt Controller (WIC), providing ultra-low power sleep mode support.
- Debug support
  - Four hardware breakpoints.
  - Two watchpoints.
  - Program Counter Sampling Register (PCSR) for non-intrusive code profiling.
  - Single step and vector catch capabilities.
- Bus interfaces
  - Single 32-bit AMBA-3 AHB-Lite system interface that provides simple integration to all system peripherals and memory.
  - Single 32-bit slave port that supports the DAP.

5.2 System Manager

5.2.1 Overview

The following functions are included in system manager section

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

5.2.2 System Memory Map

I91032 provides a 4G-byte address space for programmers. The memory locations assigned to each on-chip modules are shown in Table 5-1 Address Space Assignments for On-Chip Modules. The detailed register and memory addressing and programming will be described in the following sections for individual on-chip modules. I91032 series only supports little-endian data format.

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

Address Space	Token	Modules	Reference
Flash & SRAM Memory Space			
0x0000_0000 – 0x0000_FFFF	FLSAH_BA	Flash Memory Space (64KB)	
0x2000_0000 – 0x2000_17FF	SRAM_BA	SRAM Memory Space (6KB)	
AHB Modules Space (0x5000_0000 – 0x501F_FFFF)			
0x5000_0000 – 0x5000_01FF	SYS_BA	System Global Control Registers	5.2.3
0x5000_0200 – 0x5000_02FF	CLK_BA	Clock Control Registers	5.3.5
0x5000_0300 – 0x5000_03FF	INT_BA	Interrupt Multiplexer Control Registers	5.2.4.5
0x5000_4000 – 0x5000_4FFF	GPIO_BA	GPIO Control Registers	5.5.2
0x5000_7000 – 0x5000_7FFF	SPIM_BA	SPIM Control Registers	
0x5000_9000 – 0x5000_9FFF	PDMA_BA	PDMA Control Registers	
0x5000_C000 – 0x5000_CFFF	FMC_BA	Flash Memory Control Registers	
APB Modules Space (0x4000_0000 ~ 0x400F_FFFF)			
0x4000_4000 – 0x4000_7FFF	WDT_BA	Watch-Dog Timer Control Registers	5.11
0x4000_8000 – 0x4000_8FFF	RTC_BA	Real Time Clock (RTC) Control Registers	5.8.2
0x4001_0000 – 0x4001_0FFF	TMR_BA	Timer0/Timer1/Timer2/TimerF Control Registers	5.10.4
0x4003_0000 – 0x4003_0FFF	SPI0_BA	SPI0 Serial Interface Control Registers	5.9.7
0x4004_0000 – 0x4005_0FFF	PWM_BA	PWM Control Registers	5.6.11
0x4006_0000 – 0x4006_0FFF	UART_BA	UART Control Registers	
0x4007_0000 – 0x4007_0FFF	DPWM_BA	DPWM Control Registers	
0x400E_0000 – 0x400E_0FFF	ADC_BA	Analog-Digital-Converter (ADC) Control Registers	5.14.9
System Control Space (0xE000_E000 ~ 0xE000_EFFF)			
0xE000_E100 – 0xE000_ECFF	SCS_BA	NVIC Control Registers	5.2.4.4

5.2.3 System Manager Control Registers

Register	Offset	R/W	Description	Reset Value
SYS Base Address: SYS_BA = 0x5000_0000				
SYS_PDID	SYS_BA+0x00	R	Product Identifier Register	0xXXXX_XXXX
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Source Register	0x0000_0XXX
SYS_IPRST0	SYS_BA+0x08	R/W	IP Reset Control Resister0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	IP Reset Control Resister1	0x0000_0000
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-Out Detector Control Register	0x000X_00XX
SYS_GPA_MFP	SYS_BA+0x30	R/W	GPIO PA Multiple Alternate Functions and Input Type Control Register	0x5500_0000
SYS_GPB_MFP	SYS_BA+0x34	R/W	GPIO PB Multiple Alternate Functions and Input Type Control Register	0x0000_0000
SYS_ICE_MFP	SYS_BA+0x38	R/W	ICE Multi-Function-Pin Controller Register	0x0000_0001
SYS_GPIO_INTP	SYS_BA+0x40	R/W	GPIO input type and slew rate ontrol	0x0000_0FFF
SYS_GPA_PULL	SYS_BA+0x44	R/W	PA.15 ~ PA.0 Pull Resistance Control Register	0x0000_0000
SYS_GPA_IEN	SYS_BA+0x4C	R/W	PA.15 ~ PA.0 Digital and Analog Input Buffer Control Register	0x0000_F000
SYS_GPB_PULL	SYS_BA+0x54	R/W	PB.5 ~ PB.0 Pull Resistance Control Register	0x0000_0000
SYS_GPB_IEN	SYS_BA+0x5C	R/W	PB.5 ~ PB.0 Digital input buffer Control Register	0x0000_0000
SYS_IMGMAP3	SYS_BA+0xF0	R	MAP3 Data Image Register	0xXXXX_XXXX
SYS_DEVICEID	SYS_BA+0xF4	R	Device ID Register	0x0000_3571
SYS_IMGMAP0	SYS_BA+0xF8	R	MAP0 Data Image Register	0xXXXX_XXXX
SYS_IMGMAP1	SYS_BA+0xFC	R	MAP1 Data Image Register	0xXXXX_XXXX
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control Register	0x0000_0000
SYS_OSCTRIM	SYS_BA+0x110	R/W	Internal oscillator trim register	0xXXXX_XXXX
SYS_OSC10K	SYS_BA+0x114	R/W	10kHz Oscillator and Bias trim register	0xXXXX_XXXX
SYS_OSCTRIMn N=0,1,2	SYS_BA+0x118+(0x04*n)	R/W	Internal oscillator trim register	0xXXXX_XXXX

Product Identifier Register (SYS_PDID)

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

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

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

Bits	Description	
[15:0]	**IMG2**	**Product Identifier** Data in MAP2 of information block are copied to this register after power on. MAP2 is used to store part number defined by Nuvoton.

System Reset Source Register (SYS_RSTSTS)

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

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

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**POR_WK**	**Wakeup from DPD From POR** The device was woken from Deep Power Down by a Power On Reset. 0= No wakeup from POR. 1= The device was issued a wakeup from DPD by a POR.
[9]	**TIM_WK**	**Wakeup from DPD From TIMER** The device was woken from Deep Power Down by count of 10kHz timer. 0= No wakeup from TIMER. 1= The device was issued a wakeup from DPD by a TIMER event.
[8]	**PIN_WK**	**Wakeup from DPD From PIN** The device was woken from Deep Power Down by a low transition on the WAKEUP in or RESETB pin. 0= No wakeup from PIN. 1= The device was issued a wakeup from DPD by a pin transition. **Note:** Write 1 to this register to clear all wakeup flags.
[7]	**Reserved**	Reserved

[6]	**PMURSTF**	**Reset Source From PMU** The PMURSTF flag is set by the reset signal from the PMU module to indicate the previous reset source. 0= No reset from PMU. 1= The PMU has issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[5]	**Reserved**	Reserved.
[4]	**BOD**	**BOD Reset Flag** The BOD reset flag is set by the "Reset Signal" from the Brown Out Reset Controller to indicate the previous reset source. 0 = No reset from BOD. 1 = BOD controller had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[3]	**LVRF**	**LVR Reset Flag** The LVR reset flag is set by the "Reset Signal" from the Low Voltage Reset Controller to indicate the previous reset source. 0 = No reset from LVR. 1 = LVR controller had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[2]	**WDTRF**	**Reset Source From WDG** The WDTRF flag is set if pervious reset source originates from the Watch-Dog module. 0= No reset from Watch-Dog. 1= The Watch-Dog module issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[1]	**PINRF**	**RESETB Pin Reset Flag** The RESETB pin reset flag is set by the "Reset Signal" from the RESETB Pin to indicate the previous reset source. 0 = No reset from RESETB pin. 1 = Pin RESETB had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[0]	**PORF**	**POR Reset Flag** The POR reset flag is set by the "Reset Signal" from the Power-on Reset (POR) Controller to indicate the previous reset source. 0 = No reset from POR. 1 = Power-on Reset (POR) Controller had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.

IP Reset Control Register0(SYS_IPRST0)

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

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

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

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

IP Reset Control Register1 (SYS_IPRST1)

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

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

31	30	29	28	27	26	25	24
Reserved		**DPWMRST**	**ADCRST**	**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**PWM1RST**	**PWM0RST**	**Reserved**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**SPIMRST**	**SPI0RST**	**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDMARST	**TMRFRST**	**Reserved**	**TMR2RST**	**TMR1RST**	**TMR0RST**	**GPIORST**	**Reserved**

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**DPWMRST**	**DPWM Controller Reset** 0 = Normal Operation. 1 = Reset.
[28]	**ADCRST**	**ADC Controller Reset** 0 = Normal Operation. 1 = Reset.
[27:22]	**Reserved**	Reserved.
[21]	**PWM1RST**	**PWM1 Controller Reset** 0 = Normal Operation. 1 = Reset.
[20]	**PWM0RST**	**PWM0 Controller Reset** 0 = Normal Operation. 1 = Reset.
[19:14]	**Reserved**	Reserved.
[13]	**SPIMRST**	**SPIM Controller Reset** 0 = Normal Operation. 1 = Reset.

[12]	**SPI0RST**	**SPI0 Controller Reset** 0 = Normal Operation. 1 = Reset.
[11:8]	**Reserved**	Reserved.
[7]	**PDMARST**	**PDMA Controller Reset** 0 = Normal operation. 1 = Reset.
[6]	**TMRFRST**	**TimerF Controller Reset** 0 = Normal operation. 1 = Reset.
[5]	**Reserved**	Reserved
[4]	**TMR2RST**	**Timer2 Controller Reset** 0 = Normal operation. 1 = Reset.
[3]	**TMR1RST**	**Timer1 Controller Reset** 0 = Normal Operation. 1 = Reset.
[2]	**TMR0RST**	**Timer0 Controller Reset** 0 = Normal Operation. 1 = Reset.
[1]	**GPIORST**	**GPIO Controller Reset** 0 = Normal operation. 1 = Reset.
[0]	**Reserved**	Reserved.

Brown-Out Detector Control Register (SYS_BODCTL)

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

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

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

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18:17]	**LVR_FILTER**	00 = LVR output will be filtered by1 HCLK 01 = LVR output will be filtered by 2 HCLK 10 = LVR output will be filtered by 8 HCLK 11 = LVR output will be filtered by 15 HCLK Default value is 00
[16]	**LVR_EN**	**Low Voltage Reset (LVR) Enable (Initialized & Protected Bit)** The LVR function resets the chip when the input power voltage is lower than LVR trip point. Default value is set by flash controller as inverse of CLVR (config0 [27]). 0 = Disable LVR function. 1 = Enable LVR function.
[15:9]	**Reserved**	Reserved.
[8]	**BOD_INT**	**Brown-Out Detector Interrupt** 1 = indicates BOD_INT is active. Write 1 to clear.

<table>
<tr><td>[7]</td><td>BOD_OUT</td><td>Brown-Out Detector Output State
0 = Brown-out Detector status output is 0, the detected voltage is higher than BOD_VL setting.
1 = Brown-out Detector status output is 1, the detected voltage is lower than BOD_VL setting.</td></tr>
<tr><td>[6]</td><td>BOD_HYS</td><td>Brown-Out Detector Hysteresis (Initialized & Protected Bit)
The default value is set by flash controller user configuration CBOV[4] bit (config0 [26]).
0 = No hysteresis on BOD detection.
1 = BOD hysteresis enabled.</td></tr>
<tr><td>[5:2]</td><td>BOD_LVL[3:0]</td><td>Brown-Out Detector Threshold Voltage Selection (Initialized & Protected Bit)
The default value is set by flash controller user configuration CBOV bit (config0 [25:22]).
<table>
<tr><th>BOD_LVL[3:0]</th><th>Brown-out voltage</th><th>BOD_LVL[3:0]</th><th>Brown-out voltage</th></tr>
<tr><td>0111</td><td>2.8V</td><td>1111</td><td>4.6V</td></tr>
<tr><td>0110</td><td>2.6V</td><td>1110</td><td>4.2V</td></tr>
<tr><td>0101</td><td>2.4V</td><td>1101</td><td>3.9V</td></tr>
<tr><td>0100</td><td>2.2V</td><td>1100</td><td>3.7V</td></tr>
<tr><td>0011</td><td>2.1V</td><td>1011</td><td>3.6V</td></tr>
<tr><td>0010</td><td>2.0V</td><td>1010</td><td>3.4V</td></tr>
<tr><td>0001</td><td>1.9V</td><td>1001</td><td>3.1V</td></tr>
<tr><td>0000</td><td>1.80V</td><td>1000</td><td>3.0V</td></tr>
</table></td></tr>
<tr><td>[1]</td><td>BOD_RSTEN</td><td>Brown-Out Detector Reset or Interrupt Bit (Initialized & Protected Bit)
The default value is set by flash controller as inverse of user configuration CBORST bit (config0 [21]).
0 = Brown-Out Detector generate an interrupt
1 = Brown-Out Detector will reset chip
When the BOD is enabled and the interrupt is asserted, the interrupt will be kept till the BOD is disabled. The interrupt for CPU can be blocked either by disabling the interrupt in the NVIC or by disabling the interrupt source by disabling the BOD. BOD can then be re-enabled as required.</td></tr>
<tr><td>[0]</td><td>BOD_EN</td><td>Brown-Out Detector Enable (Initialized & Protected Bit)
The default value is set by flash controller as inverse of user configuration CBODEN bit (config0 [20]).
0 = Brown-Out Detector function is disabled
1 = Brown-Out Detector function enabled</td></tr>
</table>

GPIO PA Multiple Alternate Function Control Register (SYS_GPA_MFP)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP	SYS_BA+0x30	R/W	GPIO PA Multiple Alternate Functions Control Register	0x5500_0000

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

Bits	Description					
[31:30]	PA15MFP	PA.15 Multi-function Pin Selection				
		Pin Name	PA15MFP			
		PA.15	00	01	10	11
			PA.15	PGCVMID		
[29:28]	PA14MFP	PA.14 Multi-function Pin Selection				
		Pin Name	PA14MFP			
		PA.14	00	01	10	11
			PA.14	MICN	AIN5	
[27:26]	PA13MFP	PA.13 Multi-function Pin Selection				
		Pin Name	PA13MFP			
		PA.13	00	01	10	11
			PA.13	MICP		
[25:24]	PA12MFP	PA.12 Multi-function Pin Selection				
		Pin Name	PA12MFP			
		PA.12	00	01	10	11
			PA.12	MICBIAS		

[23:22]	**PA11MFP**	**PA.11 Multi-function Pin Selection**

Pin Name	PA11MFP			
PA.11	00	01	10	11
	PA.11			

[21:20]	**PA10MFP**	**PA.10 Multi-function Pin Selection**

Pin Name	PA10MFP			
PA.10	00	01	10	11
	PA.10			

[19:18]	**PA9MFP**	**PA.9 Multi-function Pin Selection**

Pin Name	PA9MFP			
PA.9	00	01	10	11
	PA.9			X32O

[17:16]	**PA8MFP**	**PA.8 Multi-function Pin Selection**

Pin Name	PA8MFP			
PA.8	00	01	10	11
	PA.8			X32I

[15:14]	**PA7MFP**	**PA.7 Multi-function Pin Selection**

Pin Name	PA7MFP			
PA.7	00	01	10	11
	PA.7	PWM13	CPR1	

[13:12]	**PA6MFP**	**PA.6 Multi-function Pin Selection**

Pin Name	PA6MFP			
PA.6	00	01	10	11
	PA.6	PWM12	TM1	

[11:10]	**PA5MFP**	**PA.5 Multi-function Pin Selection**

Pin Name	PA5MFP			
PA.5	00	01	10	11
	PA.5	PWM11	TM0	

[9:8]	**PA4MFP**	**PA.4 Multi-function Pin Selection**

Pin Name	PA4MFP			
PA.4	00	01	10	11
	PA.4	PWM10	SPI_SS1	

<table>
<tr><td rowspan="4">[7:6]</td><td rowspan="4">PA3MFP</td><td colspan="5">PA.3 Multi-function Pin Selection</td></tr>
<tr><td>Pin Name</td><td colspan="4">PA3MFP</td></tr>
<tr><td rowspan="2">PA.3</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.3</td><td>PWM03</td><td>SPI_MOSI</td><td></td></tr>
<tr><td rowspan="4">[5:4]</td><td rowspan="4">PA2MFP</td><td colspan="5">PA.2 Multi-function Pin Selection</td></tr>
<tr><td>Pin Name</td><td colspan="4">PA2MFP</td></tr>
<tr><td rowspan="2">PA.2</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.2</td><td>PWM02</td><td>SPI_ MISO</td><td></td></tr>
<tr><td rowspan="4">[3:2]</td><td rowspan="4">PA1MFP</td><td colspan="5">PA.1 Multi-function Pin Selection</td></tr>
<tr><td>Pin Name</td><td colspan="4">PA1MFP</td></tr>
<tr><td rowspan="2">PA.1</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.1</td><td>PWM01</td><td>SPI_CLK</td><td></td></tr>
<tr><td rowspan="4">1:0]</td><td rowspan="4">PA0MFP</td><td colspan="5">PA.0 Multi-function Pin Selection</td></tr>
<tr><td>Pin Name</td><td colspan="4">PA0MFP</td></tr>
<tr><td rowspan="2">PA.0</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PA.0</td><td>PWM00</td><td>SPI_SS0</td><td></td></tr>
</table>

GPIO PB Multiple Alternate Function Control Register (SYS_GPB_MFP)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFP	SYS_BA+0x34	R/W	GPIO PB Multiple Alternate Functions 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				PB5MFP		PB4MFP	
7	6	5	4	3	2	1	0
PB3MFP		PB2MFP		PB1MFP		PB0MFP	

Bits	Description	
[31:12]	**Reserved**	Reserved
[11:10]	**PB5MFP**	**PB.5 Multi-function Pin Selection**

Pin Name	PB5MFP			
PB.5	00	01	10	11
	PB.5	SPIM_SS	IR	

Bits	Description	
[9:8]	**PB4MFP**	**PB.4 Multi-function Pin Selection**

Pin Name	PB4MFP			
PB.4	00	01	10	11
	PB.4	SPIM_MISO	CPR0	

Bits	Description	
[7:6]	**PB3MFP**	**PB.3 Multi-function Pin Selection**

Pin Name	PB3MFP			
PB.3	00	01	10	11
	PB.3	SPIM_SIO2	PWM03	SPIM_MISO

<table>
<tr><td rowspan="4">[5:4]</td><td rowspan="4">PB2MFP</td><td colspan="5">PB.2 Multi-function Pin Selection</td></tr>
<tr><td>Pin Name</td><td colspan="4">PB2MFP</td></tr>
<tr><td rowspan="2">PB.2</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PB.2</td><td>SPIM_SIO3</td><td>PWM02</td><td>SPIM_SS</td></tr>
<tr><td rowspan="4">[3:2]</td><td rowspan="4">PB1MFP</td><td colspan="5">PB.1 Multi-function Pin Selection</td></tr>
<tr><td>Pin Name</td><td colspan="4">PB1MFP</td></tr>
<tr><td rowspan="2">PB.1</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PB.1</td><td>SPIM_MOSI</td><td>PWM01</td><td>SPIM_CLK</td></tr>
<tr><td rowspan="4">[1:0]</td><td rowspan="4">PB0MFP</td><td colspan="5">PB.0 Multi-function Pin Selection</td></tr>
<tr><td>Pin Name</td><td colspan="4">PB0MFP</td></tr>
<tr><td rowspan="2">PB.0</td><td>00</td><td>01</td><td>10</td><td>11</td></tr>
<tr><td>PB.0</td><td>SPIM_CLK</td><td>PWM00</td><td>SPIM_MOSI</td></tr>
</table>

ICE Multi-Function-Pin (SYS_ICE_MFP)

Register	Offset	R/W	Description	Reset Value
SYS_ICE_MFP	SYS_BA+0x38	R/W	ICE Multi-Function-Pin Controller Register	0x0000_0001

This register is a protected register. To program this needs an open lock sequence, write "59h", "16h", "88h" to address 0x5000_0100 to un-lock this bit. Refer to the register SYS_REGLCTL at address SYS_BA+0x100.

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**ICE_EN**	This bit will set ICE_CLK & ICE_DAT pins to be serial debug wires or PA.6/7 0 = ICE_CLK and ICE_DAT will be assigned as PA.6 and PA.7, for general IO purpose 1 = ICE_CLK and ICE_DAT will be set as ICE CLOCK/ ICE DIO, only for debugging purpose

GPIO Input type control (SYS_GPIO_INTP)

Register	Offset	R/W	Description	Reset Value
SYS_GPIO_INTP	SYS_BA+0x40	R/W	GPIO Input Type and Slew Rate Control Register	0x0000_0FFF

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

Bits	Description	
[31:6]	Reserved	Reserved
	GPx[m:n]SS **GPx[m:n]HS**	. This register controls whether the GPIO input buffer Schmitt trigger is enabled and whether high or low slew rate is selected for output driver. Each bit controls a group of four GPIO pins **GPx[m:n]SS = 1 =** input Schmitt Trigger enabled. **GPx[m:n]SS = 0 =** input CMOS enabled. **GPx[m:n]HS = 1 =** Output high slew rate. **GPx[m:n]HS = 0 =** Output low slew rate.

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

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**PU_EN[n]**	**PA.n Pull Control Register. n = 15~0** 1 = Pull-Up function Enable 0 = Pull-Up function Disable. This function only for the GPIO pin as an INPUT mode.

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

Register	Offset	R/W	Description	Reset Value
SYS_GPA_IEN	SYS_BA+0x4C	R/W	PA.15 ~ PA.0 Digital Input Buffer Control Register	0x0000_F000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] N = 0,1,…15	**IEN[n]**	**PA.n Digital Input Buffer Control Register. n = 15~0** 0 = Input buffer Enabled. 1 = Input buffer disabled, and input signal always equals to 0.

PB.5~PB.0 Pull Resistance Control Register (SYS_GPB_PULL)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_PULL	SYS_BA+0x54	R/W	PB.5 ~ PB.0 Pull Resistance Control Register	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**PU_EN[n]**	**PB.n Pull Control Register. n = 5~0** 1 = Pull-Up function Enable 0 = Pull-Up function Disable. This function only for the GPIO pin as an INPUT mode.

PB.5~PB.0 Digital Input Buffer Control Register (SYS_GPB_IEN)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_IEN	SYS_BA+0x5C	R/W	PB.5 ~ PB.0 Digital Input Buffer Control Register	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] N=0,1,…5	**IEN[n]**	**PB.n Digital Input Buffer Control Register. n = 5~0** 0 = Input buffer Enabled. 1 = Input buffer disabled, and input signal always equals to 0.

MAP3 Data Image Register (SYS_ IMGMAP3)

Register	Offset	R/W	Description	Reset Value
SYS_IMGMAP3	SYS_BA+0xF0	R	MAP3 Data Image Register	0xXXXX_XXXX

This register provides specific read-only information for software to check the content of MAP3.

31	30	29	28	27	26	25	24
IMG3[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
IMG3[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IMG3[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IMG3[7:0]							

Bits	Description	
[31:0]	**IMG3**	**Data Image of MAP3** Data in MAP3 of information block are copied to this register after power on.

Device ID Register (SYS_DEVICEID)

Register	Offset	R/W	Description	Reset Value
SYS_DEVICEID	SYS_BA+0xF4	R	Device ID Register	0x0000_3571

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DEVICEID**	**Device ID Data** This register provides specific read-only information for the Device ID

MAP0 Data Image Register (SYS_ IMGMAP0)

Register	Offset	R/W	Description	Reset Value
SYS_IMGMAP0	SYS_BA+0xF8	R	MAP0 Data Image Register	0xXXXX_XXXX

This register provides specific read-only information for software to check the content of MAP0.

31	30	29	28	27	26	25	24
IMG0[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
IMG0[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IMG0[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IMG0[7:0]							

Bits	Description	
[31:0]	**IMG0**	**Data Image of MAP0** Data in MAP0 of information block are copied to this register after power on.

MAP1 Data Image Register (SYS_ IMGMAP1)

Register	Offset	R/W	Description	Reset Value
SYS_IMGMAP1	SYS_BA+0xFC	R	MAP1 Data Image Register	0xXXXX_XXXX

This register provides specific read-only information for software to check the content of MAP1.

31	30	29	28	27	26	25	24
IMG1[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**

IMG1[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IMG1[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IMG1[7:0]							

Bits	Description	
[31:0]	**IMG1**	**Data Image of MAP1** Data in MAP1 of information block are copied to this register after power on.

Register Lock Control Register (SYS_REGLCTL)

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

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

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

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**SYS_REGLCTL[7:0]**	**Register Lock Control Code (Write Only)** Some registers have write-protection function. Writing these registers have to disable the protected function by writing the sequence value "59h", "16h", "88h" to this field. After this sequence is completed, the REGLCTL bit will be set to 1 and write-protection registers can be normal write.

<table>
<tr>
<td>[0]</td>
<td>REGLCTL</td>
<td>
Protected Register Lock/Unlock Index (Read Only)
0 = Protected registers are locked. Any write to the target register is ignored.
1 = Protected registers are unlocked.

The protected registers are:
<table>
<tr><th>Registers</th><th>Address</th><th>Note</th></tr>
<tr><td>SYS_IPRST0</td><td>0x5000_0008</td><td></td></tr>
<tr><td>SYS_BODCTL</td><td>0x5000_0018</td><td></td></tr>
<tr><td>SYS_ICE_MFP</td><td>0x5000_0038</td><td></td></tr>
<tr><td>CLK_PWRCTL</td><td>0x5000_0200</td><td>Bit[6] is not protected for pow wake-up interrupt clear.</td></tr>
<tr><td>CLK_APBCLK bit[0]</td><td>0x5000_0208</td><td>Bit[0] is watchdog clock enable.</td></tr>
<tr><td>CLK_CLKSEL 0</td><td>0x5000_0210</td><td>HCLK and CPU STCLK clo source select.</td></tr>
<tr><td>CLK_CLKSEL 1 bit[1:0]</td><td>0x5000_0214</td><td>Watchdog clock source select.</td></tr>
<tr><td>NMI_SEL bit[7]</td><td>0x5000_0380</td><td>Interrupt test mode</td></tr>
<tr><td>NMI_SEL bit[8]</td><td>0x5000_0380</td><td>NMI interrupt enable.</td></tr>
<tr><td>ISPCON</td><td>0x5000_C000</td><td>Flash ISP Control.</td></tr>
<tr><td>ISPTRG</td><td>0x5000_C010</td><td>ISP Trigger Control.</td></tr>
<tr><td>WDT_CTL</td><td>0x4000_4000</td><td>Watchdog Timer Control.</td></tr>
</table>
</td>
</tr>
</table>

Oscillator Trim Control Register (OSCTRIM)

The master oscillator is adjustable and is controlled by the OSCTRIM and OSC_TRIM[n] registers. There are factory trimmed settings available for the oscillator. The OSCTRIM register accesses the current active oscillator trim OSC_TRIM[0].

This register is a protected register. To program this needs an open lock sequence, write "59h", "16h", "88h" to address 0x5000_0100 to un-lock this bit. Refer to the register SYS_REGLCTL at address SYS_BA+0x100.

Register	Offset	R/W	Description	Reset Value
SYS_OSCTRIM	SYS_BA+0x110	R/W	Internal Oscillator Trim Register	0xXXXX_XXXX

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

Bits	Description	
[15]	**EN2MHZ**	1: Low Frequency mode of oscillator active (2MHz). 0: High frequency mode (20-50MHz)
[9:0]	**TRIM**	10 bit trim for oscillator.

10kHz Oscillator Trim Register (SYS_OSC10KTRIM)

Register	Offset	R/W	Description	Reset Value
SYS_OSC10KTRIM	SYS_BA+0x114	R/W	10kHz Oscillator and Bias Trim Register	0xXXXX_XXXX

31	30	29	28	27	26	25	24

TRM_CLK	Reserved						
23	22	21	20	19	18	17	16
Reserved	OSC10K_TRIM[22:16]						
15	14	13	12	11	10	9	8
OSC10K_TRIM [15:8]							
7	6	5	4	3	2	1	0
OSC10K_TRIM [7:0]							

Bits	Description	
[31]	**TRM_CLK**	Must be toggled to load a new OSC10K_TRIM
[22:0]	**OSC10K _TRIM**	23bit trim for 10kHz oscillator.

Internal Oscillator Trim Register (SYS_OSCTRIMx)

Register	Offset	R/W	Description	Reset Value
SYS_OSCTRIMn **n=0,1,.2**	SYS_BA+0x118+(0x04*n)	R/W	Internal Oscillator Trim Register	0xXXXX_XXXX

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

Bits	Description

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

5.2.4 Nested Vectored Interrupt Controller (NVIC)

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

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

The NVIC prioritizes and handles all supported exceptions. All exceptions are handled in "Handler Mode". This NVIC architecture supports 32 (IRQ[31:0]) discrete interrupts with 4 levels of priority. All of the interrupts and most of the system exceptions can be configured to different priority levels. When an interrupt occurs, the NVIC will compare the priority of the new interrupt to the current running one's priority. If the priority of the new interrupt is higher than the current one, the new interrupt handler will override the current handler.

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

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

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

5.2.4.1 Exception Model and System Interrupt Map

The following table lists the exception model supported by I91032. Software can set four levels of priority on certain exceptions as well as on all interrupts. The highest user-configurable priority is denoted as "0" and the lowest priority is denoted as "3". The default priority of all the user-configurable interrupts is "0". Note that priority "0" is treated as the fourth priority on the system, after three system exceptions "Reset", "NMI" and "Hard Fault".

Table 5-1 Exception Model

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

Table 5-2 System Interrupt Map

Vector Number	Interrupt Number (Bit In Interrupt Registers)	Interrupt Name	Source IP	Interrupt Description
0 ~ 15	-	-	-	System exceptions
16	0	**WDT_INT**	WDT	Watch Dog Timer interrupt
17	1	**DPWM_INT**	DPWM	DPWM threshold interrupt
18	2	**ADC_INT**	ADC	Interrupt from ADC
19	3	**Reserved**		
20	4	**SPIM_INT**	SPIM	Interrupt from SPIM
21	5	**TMR0_INT**	Timer0	Timer0
22	6	**TMR1_INT**	Timer1	Timer1
23	7	**TMR2_INT**	Timer2	Timer2
24	8	**GPAB_INT**	GPIO A/B	Port interrupt from GPIO port
25	9	**SPI0_INT**	SPI0	Interrupt from SPI0
26	10	**PWM0_INT**	PWM0 Timer	PWM Timer interrupt
27	11	**PDMA**	PDMA	Interrupt from PDMA
28	12	**TMRF_INT**	TimerF	Fixed frequency TimerF

29	13	**RTC_INT**	RTC	Real time clock interrupt
30	14	**Reserved**		
31	15	**PWM1_INT**	PWM1 Timer	PWM Timer interrupt
32	16	**Reserved**		
33	17	**URT0_INT**	UART	UART interrupt
34	18	**BOD_INT**	BOD	Brown-out detector interrupt.
>35	>19	**Reserved**	-	-

5.2.4.2 Vector Table

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

Table 5-3 Vector Table Format

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

5.2.4.3 Operation Description

NVIC interrupts can be enabled and disabled by writing to their corresponding Interrupt Set-Enable or Interrupt Clear-Enable register bit-field. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current enabled state of the corresponding interrupts. When an interrupt is disabled, interrupt assertion will cause the interrupt to become Pending, however, the interrupt will not activate. If an interrupt is Active when it is disabled, it remains in its Active state until cleared by reset or an exception return. Clearing the enable bit prevents new activations of the associated interrupt.

NVIC interrupts can be pended/un-pended using a complementary pair of registers to those used to enable/disable the interrupts, named the Set-Pending Register and Clear-Pending Register respectively. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current pended state of the corresponding interrupts. The Clear-Pending Register has no effect on the execution status of an Active interrupt.

NVIC interrupts are prioritized by updating an 8-bit field within a 32-bit register (each register supporting four interrupts).

The general registers associated with the NVIC are all accessible from a block of memory in the System Control Space and will be described in next section.

5.2.4.4 NVIC Control Registers

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

Register	Offset	R/W	Description	Reset Value
SCS Base Address: SCS_BA = 0xE000_E000				
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ18 Set-Enable Control Register	0x0000_0000
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ18 Clear-Enable Control Register	0x0000_0000
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ18 Set-Pending Control Register	0x0000_0000
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ18 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 ~ IRQ18 Priority Control Register	0x0000_0000

IRQ0 ~ IRQ15 Set-Enable Control Register （NVIC_ISER）

Register	Offset	R/W	Description	Reset Value
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ18 Set-Enable Control Register	0x0000_0000

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**SETENA**	**Interrupt Set-Enable Bit** The NVIC_ISER register enables interrupts, and shows what interrupts are enabled. Each bit represents an interrupt number from IRQ0 ~ IRQ18 (Vector number from 16 ~ 34). Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ15 Clear-Enable Control Register (NVIC_ICER）

Register	Offset	R/W	Description	Reset Value
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ18 Clear-Enable Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CLRENA**	**Interrupt Clear-Enable Bit** The NVIC_ICER register disables interrupts, and shows what interrupts are enabled. Each bit represents an interrupt number from IRQ0 ~ IRQ18 (Vector number from 16 ~ 34). Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ15 Set-Pending Control Register （NVIC_ISPR）

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ18 Set-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**SETPEND**	**Interrupt Set-Pending Bit** The NVIC_ISPR register forces interrupts into the pending state, and shows what interrupts are pending. Each bit represents an interrupt number from IRQ0 ~ IRQ18 (Vector number from 16 ~ 34). Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ15 Clear-Pending Control Register （NVIC_ICPR）

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ18 Clear-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CLRPEND**	**Interrupt Clear-Pending Bit** The NVIC_ICPR register removes the pending state of associated interrupts, and shows what interrupts are pending. Each bit represents an interrupt number from IRQ0 ~ IRQ18 (Vector number from 16 ~ 34). Write Operation: 0 = No effect. 1 = Removes pending state of an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ3 Interrupt Priority Register （NVIC_IPR0）

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

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

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

IRQ4 ~ IRQ7 Interrupt Priority Register （NVIC_IPR1）

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

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

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

IRQ8 ~ IRQ11 Interrupt Priority Register （NVIC_IPR2）

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

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

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

IRQ12 ~ IRQ15 Interrupt Priority Register （NVIC_IPR3）

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

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

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

IRQ16 ~ IRQ18 Interrupt Priority Register （NVIC_IPR4）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ18 Priority Control Register	0x0000_0000

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

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

5.2.4.5 Interrupt Source Control Registers

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

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

Register	Offset	R/W	Description	Reset Value
INT Base Address: INT_BA = 0x5000_0300				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (WDT) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (DPWM) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (ADC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	Reserved	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (SPIM) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (Timer0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (Timer1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (Timer2) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (GPA/B) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (SPI0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (PWM0) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (PDMA) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (TimerF) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (RTC) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	Reserved	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (PWM1) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	Reserved	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (UART) Interrupt Source Identity Register	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (BOD) Interrupt Source Identity Register	0xXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_001F
MCU_IRQ	INT_BA+0x84	R/W	MCU IRQ Number Identify Register	0x0000_0000

IRQ0 (WDT) Interrupt Source Identify Register (IRQ0_SRC)

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

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

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

IRQ1 (DPWM) Interrupt Source Identify Register (IRQ1_SRC)

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

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

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

IRQ2 (ADC) Interrupt Source Identify Register (IRQ2_SRC)

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

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

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

IRQ4 (SPIM) Interrupt Source Identify Register (IRQ4_SRC)

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

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

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

IRQ5 (Timer0) Interrupt Source Identify Register (IRQ5_SRC)

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: Timer0_INT

IRQ6 (Timer1) Interrupt Source Identify Register (IRQ6_SRC)

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: Timer1_INT

IRQ7(Timer2) Interrupt Source Identify Register (IRQ7_SRC)

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: Timer2_INT

IRQ8 (GPA/B) Interrupt Source Identify Register (IRQ8_SRC)

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: GPB_INT Bit0: GPA_INT

IRQ9 (SPI0) Interrupt Source Identify Register (IRQ9_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (SPI0) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: SPI0_INT

IRQ10 (PWM0) Interrupt Source Identify Register (IRQ10_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (PWM0) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: PWM0_INT

IRQ11 (PDMA) Interrupt Source Identify Register (IRQ11_SRC)

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

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

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

IRQ12 (TimerF) Interrupt Source Identify Register (IRQ12_SRC)

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

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

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

IRQ13 (RTC) Interrupt Source Identify Register (IRQ13_SRC)

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

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

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

IRQ15 (PWM1) Interrupt Source Identify Register (IRQ15_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (PWM1) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: PWM1_INT

IRQ17 (UART0) Interrupt Source Identify Register (IRQ17_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (UART0) Interrupt Source Identity Register	0xXXXX_XXXX

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**INT_SRC**	**Interrupt Source Identity** Bit2: 0 Bit1: 0 Bit0: UART0_INT

IRQ18 (BOD) Interrupt Source Identify Register (IRQ18_SRC)

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

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

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

NMI Interrupt Source Select Control Register (NMI_SEL)

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**IRQ_TM**	**IRQ Test Mode** This bit is the protected bit. To program this bit needs an open lock sequence, write “59h”, “16h”, “88h” to register SYS_REGLCTL to un-lock this bit. Refer to the register SYS_REGLCTL at address SYS_BA+0x100. 0 = The interrupt register MCU_IRQ operates in normal mode. The MCU_IRQ collects all the interrupts from the peripheral and generates interrupt to MCU. 1 = All the interrupts from peripheral to MCU are blocked. The peripheral IRQ signals (0-15) are replaced by the value in the MCU_IRQ register.
[6:5]	**Reserved**	Reserved.
[4:0]	**NMI_SEL**	**NMI Source Interrupt Select** The NMI interrupt to Cortex-M0 can be selected from one of the interrupt [18:0]. The NMI_SEL bit is used to select the NMI interrupt source. **Note:** IRQ3, IRQ14 and IRQ16 are reserved.

MCU Interrupt Request Source Test Mode Register (MCU_IRQ)

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
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	
[31:16]	**Reserved**	Reserved.
[15:0]	**MCU_IRQ**	**MCU IRQ Source Test Mode** The MCU_IRQ collects all the interrupts from the peripherals and generates the synchronous interrupt to MCU Cortex-M0. There are two modes to generate interrupt to MCU Cortex-M0, the normal mode and test mode. In Normal mode (NMI_SEL register bit [7] = 0) The MCU_IRQ collects all interrupts from each peripheral and synchronizes them to interrupt the Cortex-M0. In Test mode (NMI_SEL register bit [7] = 1), the interrupts from peripherals are blocked, and the interrupts are replaces by MCU_IRQ[18:0]. When MCU_IRQ[n] is “0” : Writing MCU_IRQ[n] “1” will generate an interrupt to Cortex_M0 IRQ[n]. When MCU_IRQ[n] is “1” (meaning an interrupt is asserted): Writing MCU_IRQ[n] “1” will clear the interrupt; writing MCU_IRQ[n] “0”: has no effect. **Note:** IRQ3, IRQ14 and IRQ16 are reserved.

5.3 Clock Controller

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

5.3.1 Clock Generator

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

- One external 32.768k Hz crystal (Low speed external crystal, LXT).
- One internal 10k Hz RC oscillator (Low speed internal oscillator, LIRC).
- One internal 49.152MHz RC oscillator (High speed internal Oscillator, HIRC).

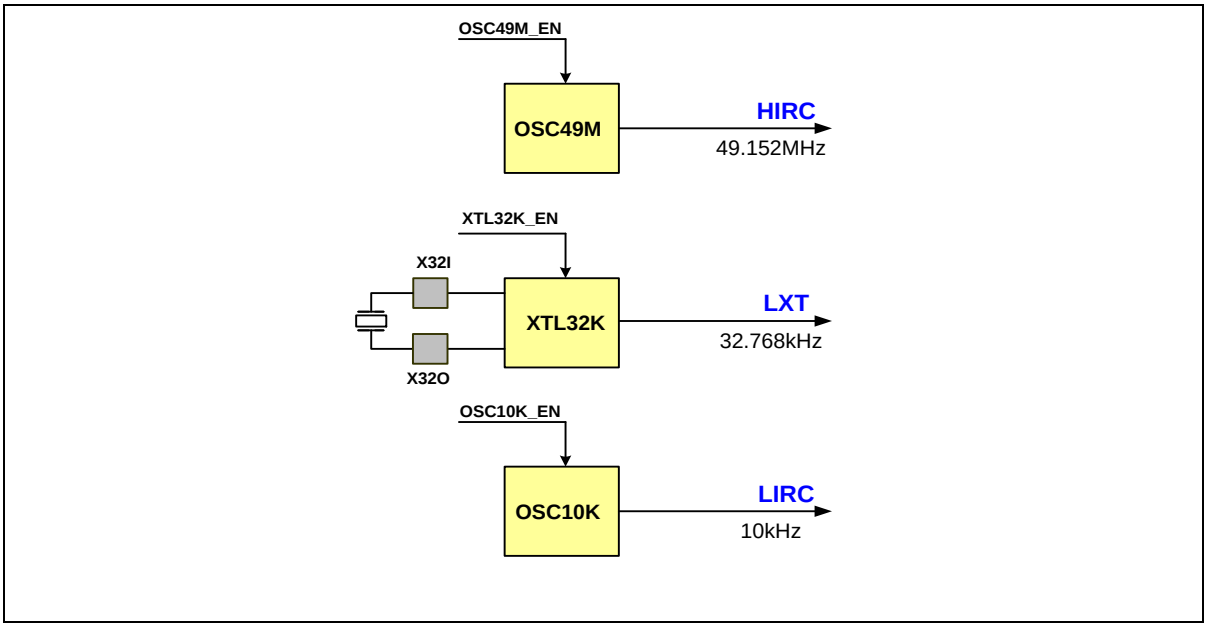


Figure 5-2 Clock generator block diagram

5.3.2 System Clock

The system clock has 3 clock sources which were generated from clock generator block. The clock source switch depends on the register HCLKSEL(CLK_CLKSEL0[2:0]). The block diagram is listed below.

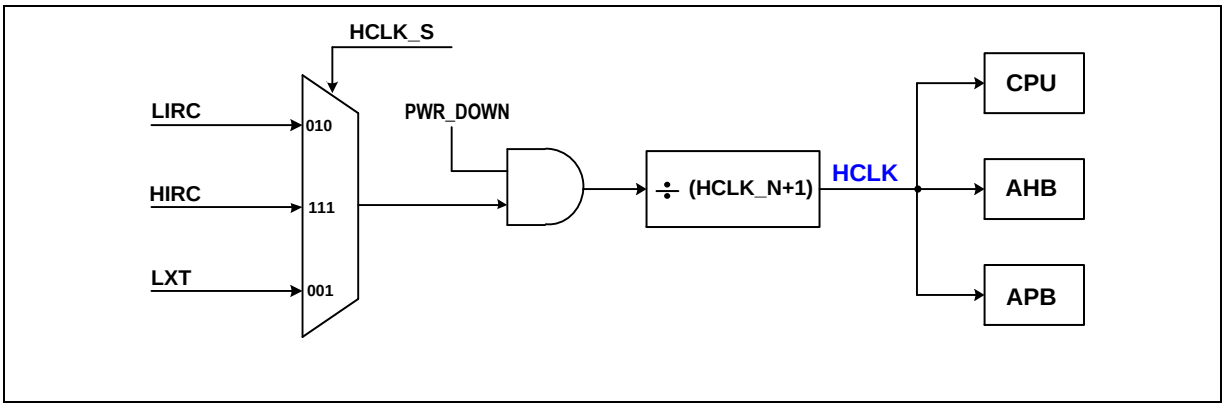


Figure 5-3 System Clock Block Diagram

5.3.3 Peripheral Clock

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

5.3.4 Power Management

The I91032 is equipped with a Power Management Unit (PMU) that implements a variety of power saving modes. There are four levels of power control with increasing functionality (and power consumption):

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

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

5.3.4.1 Level0: Deep Power Down (DPD)

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

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

1. A high to low transition on the WAKEUP (GPA4) pin.
2. A timed wakeup where the LIRC oscillator is configured active and reaches a certain count.
3. A power cycle of main chip supply triggering a POR event.

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

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

- SYSCLK-> PWRCON.PIN_ENB: If set to '1' then the WAKEUP pin is disabled and will not wake up the chip.
- SYSCLK->PWRCON.DPD_10K: If set to '1' then the 10kHz oscillator will power down in DPD. No timed wakeup is possible.
- SYSCLK->PWRCON.TIMER_SEL: Each bit in this register will trigger a wakeup event after a certain number of OSC10K 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 SYSCLK->PWRCON.PIN_WU, SYSCLK->PWRCON.TIMER_WU and SYSCLK->PWRCON.POI_WU.

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

5.3.4.2 **Level1: Standby Power Down (STOP) mode**

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

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

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

5.3.4.3 **Level2: Deep Sleep mode**

The Deep Sleep mode is the lowest power state where the Cortex-M0 and all logic state are preserved. In Deep Sleep mode the CLK48M oscillator is shut down and a low speed oscillator is selected, if CLK32K is active this source is selected, if not then CLK10K is enabled and selected. All clocks to the Cortex-M0 core are gated eliminating dynamic power in the core. HCLK is operating at a low frequency and CLK48M is not available. Deep Sleep mode is entered by setting System Control register bit 2: SCB->SCR |= (1UL << 2) and executing a WFI/WFE instruction. Software can determine whether the device woke up from Deep Sleep by interrogating the register bit SYSCLK->PFLAGCON.DS_FLAG.

5.3.4.4 **Level3: Sleep mode**

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

5.3.5 Clock Control Register Map

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

Register	Offset	R/W	Description	Reset Value
CLK Base Address: CLK_BA = 0x5000_0200				
CLK_PWRCTL	CLK_BA + 0x00	R/W	System Power Control Register	0x0C00_1004
CLK_AHBCLK	CLK_BA + 0x04	R/W	AHB Device Clock Enable Control Register	0x0000_0005
CLK_APBCLK	CLK_BA + 0x08	R/W	APB Device Clock Enable Control Register	0x0000_0000
CLK_DPDFLR	CLK_BA + 0x0C	R/W	Deep Power Down State and Flash Regulator	0x8000_XXXX
CLK_CLKSEL0	CLK_BA + 0x10	R/W	Clock Source Select Control Register 0	0x000X_0007
CLK_CLKSEL1	CLK_BA + 0x14	R/W	Clock Source Select Control Register 1	0x0070_0004
CLK_CLKDIV	CLK_BA + 0x18	R/W	Clock Divider Number Register	0x0018_0000
CLK_PFLAG	CLK_BA + 0x24	R/W	Power down Flag Register	0x0000_0000

5.3.6 Clock Control Register Description

System Power Control Register (CLK_PWRCTL)

Except the BIT[6], all the other bits are protected. To program these bits needs an open lock sequence, write “59h”, “16h”, “88h” to register SYS_REGLCTL to un-lock these bits. Refer to the register SYS_REGLCTL at address SYS_BA + 0x100.

Register	Offset	R/W	Description	Reset Value
CLK_PWRCTL	CLK_BA + 0x00	R/W	System Power Control Register	0x0C00_1004

31	30	29	28	27	26	25	24
TIMER_SEL_RD[3:0]				**WK_TRI**	**POI_WU**	**TIMER_WU**	**PIN_WU**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TIMER_SEL[3:0]				**FLASH_PWR**		**DPD_10K**	**WK_ENB**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VSET[2:0]			**OSC10K_EN**	**Reserved**	**DEEP_PD**	**STOP**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**XTL32K_FILTER**	**OSC49M_EN**	**XTL32K_EN**	**Reserved**

Bits	Description	
[31:28]	TIMER_SEL_RD	Read-Only. Read back of the current WAKEUP timer setting. This value is updated with TIMER_SEL upon entering DPD mode.
[27]	**WK_TRI**	1: Tristate the wakeup pin (GPIOA[4]) in DPD. 0: Add pull-up to wakeup pin in DPD.
[26]	**POI_WU**	Read Only. This flag indicates that wakeup of device was requested with a power-on reset. Flag is cleared when DPD mode is entered or any of the DPD bits of RSTSRC register (RSTSRC[10:8]) are cleared.
[25]	**TIMER_WU**	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 or any of the DPD bits of RSTSRC register (RSTSRC[10:8]) are cleared.
[24]	**PIN_WU**	Read Only. This flag indicates that wakeup of device was requested with a high to low transition of the WAKEUP pin. Flag is cleared when DPD mode is entered or any of the DPD bits of RSTSRC register (RSTSRC[10:8]) are cleared.

[23:20]	**TIMER_SEL**	Select WAKEUP Timer: TIMER_SEL[0]=1: WAKEUP after 128 OSC10K clocks (12.8 ms) TIMER_SEL[1]=1: WAKEUP after 256 OSC10K clocks (25.6 ms) TIMER_SEL[2]=1: WAKEUP after 512 OSC10K clocks (51.2 ms) TIMER_SEL[3]=1: WAKEUP after 1024 OSC10K clocks (102.4ms)
[19:18]	**FLASH_PWR**	Determine whether FLASH memory enters deep power down. FLASH_PWR[0]: 1: flash enters deep power down upon DEEP_SLEEP FLASH_PWR[1]: 1: flash enters deep power down upon STOP mode. If FLASH_PWR is selected for a power state mode, current consumption is reduced, but a 10us wakeup time must be added to the wakeup sequence. Trade-off is wakeup time for standby power.
[17]	**DPD_10K**	Determines whether OSC10K is enabled in DPD mode. 0=enabled, 1=disabled in DPD. If OSC10K is disabled, device cannot wake from DPD with TIMER_SEL delay.
[16]	**WK_ENB**	Determines whether WAKEUP pin is enabled in DPD mode. 0=enabled, 1=disabled.
[15:13]	**VSET[2:0]**	Adjusts the digital supply voltage. Should be left as default 0.
[12]	**OSC10K_EN**	**Internal 10kHz Oscillator Control** After reset, this bit is "0". 0 = Internal 10KHz oscillator Disabled. 1 = Internal 10KHz oscillator Enabled.
[11]	**Reserved**	Reserved
[10]	**DEEP_PD**	Deep Power Down (DPD) bit. Set to '1' and issue WFI/WFE instruction to enter DPD mode.
[9]	**STOP**	STOP mode bit. Set to '1' and issue WFI/WFE instruction to enter STOP mode.
[8:4]	**Reserved**	Reserved
[3]	**XTL32K_FILTER**	**Filter the XTL32K output clock** 0 = Disable, XTL32K output clock without filter. 1 = Enable, XTL32K output clock will be filtered to avoid glitches. **Note 1**: High level of XTL32K must keep 112 HCLK for recognition valid, when this bit is enabled. **Note 2**: Should be disable when enter power down
[2]	**OSC49M_EN**	**Internal 49.152MHz RC Oscillator Control** After reset, this bit is "1". 0 = 49.152MHz oscillation Disabled. 1 = 49.152MHz oscillation Enabled.

[1]	**XTL32K_EN**	**External 32.768kHz Crystal Control** After reset, this bit is “0”. 0 = External 32.768kHz crystal Disabled. 1 = External 32.768kHz crystal Enabled.
[0]	**Reserved**	Reserved

AHB Device Clock Enable Control Register (CLK_AHBCLK)

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

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[3]	**UART_EN**	**UART Clock Enable Control (It works on APB)** 0 = UART engine clock Disabled. 1 = UART engine clock Enabled.
[2]	**ISPCKEN**	**Flash ISP Controller Clock Enable Control.** The Flash ISP engine clock always is from HIRC oscillator. 0 = Disable the Flash ISP engine clock. 1 = Enable the Flash ISP engine clock.
[1]	**SPIMCKEN**	**SPIM Clock Enable Control** 0 = SPIM engine clock Disabled. 1 = SPIM engine clock Enabled.
[0]	**PDMACKEN**	**PDMA Clock Enable Control** 0 = PDMA engine clock Disabled. 1 = PDMA engine clock Enabled.

APB Device Clock Enable Control Register (CLK_APBCLK)

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

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

31	30	29	28	27	26	25	24
Reserved	DAC_EN	DPWM_EN	ADC_EN	Reserved			
23	22	21	20	19	18	17	16
Reserved		PWM1_EN	PWM0_EN	Reserved			
15	14	13	12	11	10	9	8
Reserved			SPI0_EN	Reserved			
7	6	5	4	3	2	1	0
Reserved		TMRF_EN	TMR2_EN	TMR1_EN	TMR0_EN	RTC_EN	WDT_EN

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**DAC_EN**	**DAC clock enable control** 0 = DAC clock Disabled 1 = DAC clock Enabled
[29]	**DPWM_EN**	**DPWM Clock Enable Control** 0 = DPWM clock Disabled. 1 = DPWM clock Enabled.
[28]	**ADC_EN**	**Audio Analog-Digital-Converter (ADC) Clock Enable Control** 0=Disable. 1=Enable.
[27:22]	**Reserved**	Reserved.
[21]	**PWM1_EN**	**PWM1 Block Clock Enable Control** 0=Disable. 1=Enable.
[20]	**PWM0_EN**	**PWM0 Block Clock Enable Control** 0=Disable. 1=Enable.

[19:13]	**Reserved**	Reserved.
[12]	**SPI0_EN**	**SPI0 Clock Enable Control** 0=Disable. 1=Enable.
[11:6]	**Reserved**	Reserved.
[5]	**TMRF_EN**	**TimerF Clock Enable Control** 0=Disable. 1=Enable.
[4]	**TMR2_EN**	**Timer2 Clock Enable Control** 0=Disable. 1=Enable.
[3]	**TMR1_EN**	**Timer1 Clock Enable Control** 0=Disable. 1=Enable.
[2]	**TMR0_EN**	**Timer0 Clock Enable Control** 0=Disable. 1=Enable.
[1]	**RTC_EN**	**Real-Time-Clock APB Interface Clock Control** This bit is used to control the RTC APB clock only. The RTC engine clock source is from the 32.768KHz crystal. 0=Disable. 1=Enable.
[0]	**WDT_EN**	**Watchdog Clock Enable Control** This bit is the protected bit. To program this bit needs an open lock sequence, write “59h”, “16h”, “88h” to register SYS_REGLCTL to un-lock this bit. Refer to the register SYS_REGLCTL at address SYS_BA+0x100. 0=Disable. 1=Enable.

DPD State Register and Flash Regulator Control (CLK_DPDFLR)

Register	Offset	R/W	Description	Reset Value
CLK_DPDFLR	CLK_BA + 0x0C	R/W	DPD State Register and Flash Regulator Control	0x8000_XXXX

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

Bits	Description	
[31]	**PD**	1: Power Down Flash LDO regulator. 0: Power Up Flash LDO regulator.
[30:27]	**Reserved**	Reserved
[26:24]	**VSET**	Flash Regulator Voltage control. 0:1.8V, 1::2.4V, 2:2.5V, 3:2.7V, 4:3V, 5:3.3V, 6:1.5V, 7:1.7V
[23:16]	**Reserved**	Reserved
[15:8]	**PD_STATE_RB**	Current value of PD_STATE register.
[7:0]	**PD_STATE**	An 8bit register that is preserved when DPD (Deep Power Down) state is entered and after wakeup is available by reading PD_STATE_RB.

Clock Source Select Control Register 0 (CLK_CLKSEL0)

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

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL0	CLK_BA + 0x10	R/W	Clock Source Select Control Register 0	0x0001_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			STICKSEL		HCLKSEL		

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:3]	**STICKSEL**	**SYS_TICK Clock Source Select** 00 = clock source from HIRC 01 = clock source from LIRC 10 = clock source from LXT 11 = clock source from HCLK **Note:** 1. When power on, HIRC is selected as HCLK clock source. 2. Before clock switch, the related clock sources (pre-select and new-select) must be turned on.
[2:0]	**HCLKSEL**	**HCLK Clock Source Select** 000 = clock source from HIRC 001 = clock source from LXT 010 = clock source from LIRC 111 = clock source from HIRC Others = Reserved and equivalent with “111”. **Note:** 1. When power on, HIRC is selected as HCLK clock source. 2. Before clock switch, the related clock sources (pre-select and new-select) must be turned on.

Clock Source Select Control Register 1 (CLK_CLKSEL1)

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL1	CLK_BA + 0x14	R/W	Clock Source Select Control Register 1	0x0070_0004

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

Bits	Description	
[31:30]	**PWM1SEL**	**PWM Timer Clock Source Select** 00 = Clock source from HCLK. 01 = Clock source from LXT. 10 = Clock source from LIRC. 11 = Clock source from HIRC.
[29:28]	**PWM0SEL**	**PWM Timer Clock Source Select** 00 = Clock source from HCLK. 01 = Clock source from LXT. 10 = Clock source from LIRC. 11 = Clock source from HIRC.
[27:25]	**Reserved**	Reserved.
[24]	**RTCSEL**	**RTC Clock Source Select** 0 = Clock source from LIRC. 1 = Clock source from LXT.
[23]	**Reserved**	Reserved.

[22:20]	**TMRFSEL**	**TimerF Clock Source Select** 000 = Clock source from external LXT / 32, 001 = Clock source from external LXT / (4x32). 010 = Reserved, 011 = Reserved. 100 = Clock source from external LXT / 33, 101 = Clock source from external LXT / 33, 110 = Clock source from HIRC / 32768. 111 = Clock source from HIRC / (4x32768).
[19]	**Reserved**	Reserved.
[18:16]	**TMR2SEL**	**Timer2 Clock Source Select** 000 = Clock source from HCLK. 001 = Clock source from LXT. 010 = Clock source from LIRC. 011 = Equivalent with “000” 1xx = Clock source from HIRC.
[15]	**Reserved**	Reserved.
[14:12]	**TMR1SEL**	**Timer1 Clock Source Select** 000 = Clock source from HCLK. 001 = Clock source from LXT. 010 = Clock source from LIRC. 011 = Clock source from External Trigger. 1XX = Clock source from HIRC.
[11]	**Reserved**	Reserved.
[10:8]	**TMR0SEL**	**Timer0 Clock Source Select** 000 = Clock source from HCLK. 001 = Clock source from LXT. 010 = Clock source from LIRC. 011 = Clock source from External Trigger. 1xx = Clock source from HIRC.
[7:4]	**Reserved**	Reserved.
[3:2]	**ADCSEL**	**ADC Clock Source Select** 00 = Clock source from HCLK. 01 = Clock source from HCLK. 1x = Clock source from HIRC.

[1:0]	**WDTSEL**	**Watchdog Timer Clock Source Selection (Write Protect)** These bits are protected bits. To program these bits needs an open lock sequence, write “59h”, “16h”, “88h” to SYS_REGLCTL to un-lock these bits. Refer to the register SYS_REGLCTL at address SYS_BA+0x100.. 10 = Clock source from HCLK/2048. 01 = Clock source from LXT. 00= Clock source from LIRC. 11 = Clock source from HIRC.

Clock Divider Register (CLK_CLKDIV)

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

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

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22:16]	**ADCDIV**	**ADC Clock Divide Number From ADC Clock Source** The ADC clock frequency ADCLK = (ADC clock source frequency) / (ADCDIV + 1). The ADC engine clock must meet the constraint: ADCLK ≤ HCKL/2.
[15:4]	**Reserved**	Reserved.
[3:0]	**HCLKDIV**	**HCLK Clock Divide Number From HCLK Clock Source** The HCLK clock frequency = (HCLK clock source frequency) / (HCLKDIV + 1).

Power Down Flag Register (CLK_PFLAG)

Register	Offset	R/W	Description	Reset Value
CLK_PFLAG	CLK_BA + 0x24	R/W	Power Down Flag Register	0x0000_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						**STOP_FLAG**	**DS_FLAG**

Bits	Descriptions	
[31:2]	**Reserved**	Reserved
[1]	STOP_FLAG	**Device has been in STOP mode – write '1' to clear**.
[0]	DS_FLAG	**Device has been in DEEP_SLEEP mode – write '1' to clear**

5.4 SRAM

5.4.1 Overview

I91032 equips with 6KB SRAM for program code and data storage. It's an AHB slave interface. It supports byte, half, and word read, and write access.

5.4.2 Block Diagram

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

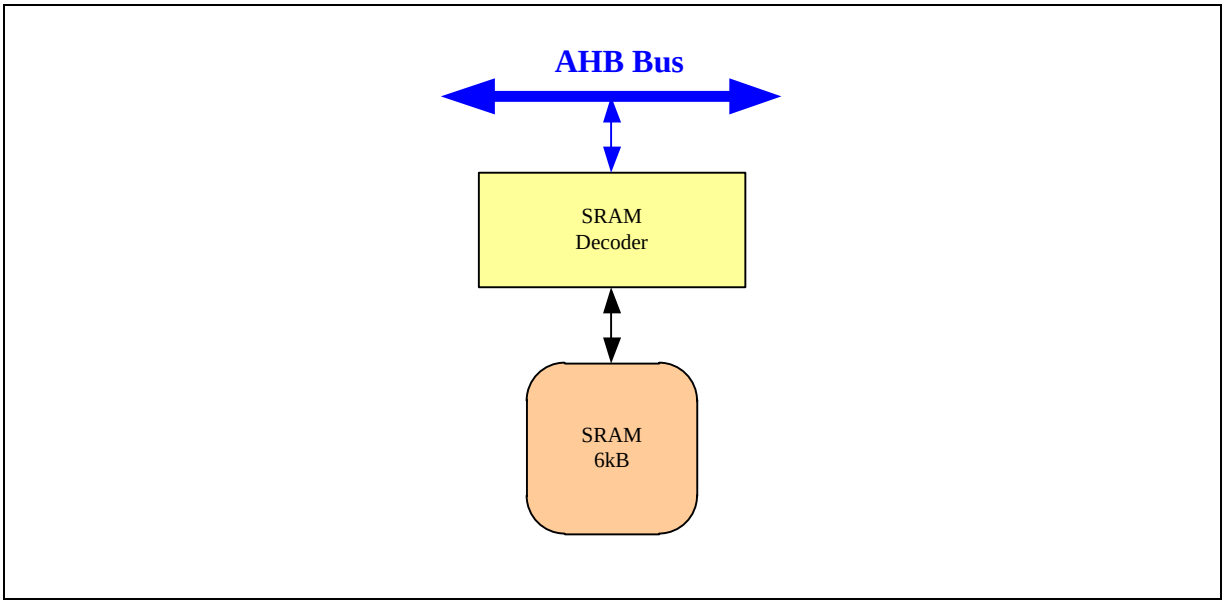


Figure 5-4 SRAM Controller Block Diagram

5.5 General Purpose I/O

5.5.1 Overview and Features

There are 22 pins of General Purpose I/O shared with special feature functions. These pins are arranged in 2 groups, PA and PB. Each pin of the 28 pins is independent and has the corresponding register bits to control the pin mode function and data.

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

In input mode, the I/O pin could be tri-state (high impedance) or with pull-up optional.

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

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

In quasi-mode, the I/O pin supports digital output and input function at the same time but the source current is only up to hundreds μA. Each I/O pin equips an individual pull-up resistor which is enabled by software setting. The pull-up resistor is about 110KΩ~300KΩ for V_{DD} is from 5.0V to 1.8V.

5.5.1.1 Open-Drain mode explanation

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

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

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

.

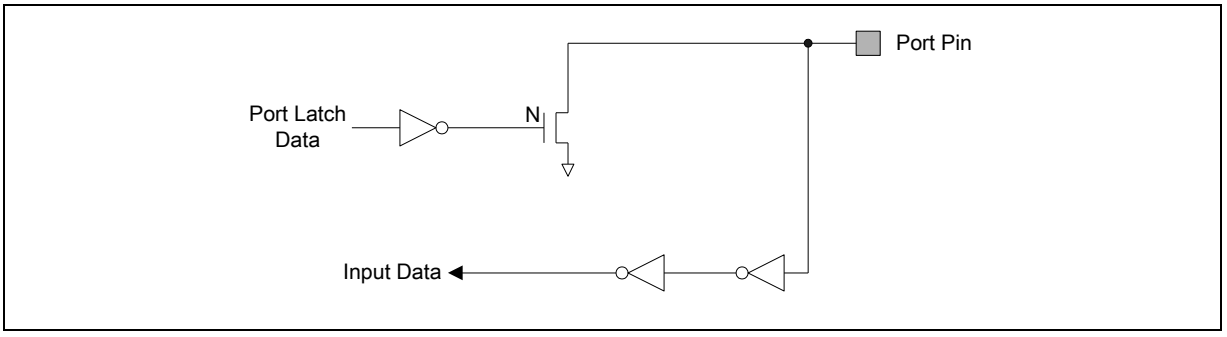


Figure 5-5 Open-Drain Output

5.5.1.2 Quasi-bidirectional Mode Explanation

For Px_MODE*n* = 11b the GPIO Px[*n*] pin is in Quasi-bidirectional mode.

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

If the bit value in the corresponding bit Px_DOUT[n] is "1", the pin will check the pin value. If pin value is high, no action takes. If pin state is low, it will drive 2 clock cycles "high" and then disable the output drive and then the pin status is controlled by internal pull-high MOS.

5.5.2 GPIO Control Register Map

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

Register	Offset	R/W	Description	Reset Value
GPIO Base Address: GPIO_BA = 0x5000_4000				
PA_MODE	GPIO_BA+0x000	R/W	GPIO PA Pin I/O Mode Control	0x0000_0000
PA_DOUT	GPIO_BA+0x008	R/W	GPIO PA Data Output Value	0x0000_FFFF
PA_PIN	GPIO_BA+0x010	R	GPIO PA Pin Value	0x0000_XXXX
PA_INTTYPE	GPIO_BA+0x018	R/W	GPIO PA Interrupt Trigger Type	0x0000_0000
PA_INTEN	GPIO_BA+0x01C	R/W	GPIO PA Interrupt Enable	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	GPIO PA Interrupt Source Flag	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	GPIO PB Pin I/O Mode Control	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	GPIO PB Data Output Value	0x0000_003F
PB_PIN	GPIO_BA+0x050	R	GPIO PB Pin Value	0x0000_00XX
PB_INTTYPE	GPIO_BA+0x058	R/W	GPIO PB Interrupt Trigger Type	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	GPIO PB Interrupt Enable	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	GPIO PB Interrupt Source Flag	0x0000_0000
PAn_PDIO n=0,1..15	GPIO_BA+0x800 +(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO n=0,1..5	GPIO_BA+0x840 +(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X

5.5.3 GPIO Control Register Description

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

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

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

Bits	Description	
[2n+1 :2n] n=0,1..15	**MODEn**	**Port [A/B] Pin[N] I/O Mode Control** Each GPIO Px pin has four modes: 00 = GPIO Px[n] pin is in INPUT mode. 01 = GPIO Px[n] pin is in OUTPUT mode. 10 = GPIO Px[n] pin is in Open-Drain mode. 11 = GPIO Px[n] pin is in Quasi-bidirectional mode. **Note**: PB_MODE[15:6] are reserved to 0.

Note: It's better not to set PA.13-PA.15 as input mode with pull-high because share design with MIC function. May impact each other when pull-high enable.

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

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	GPIO PA Data Output Value	0x0000_FFFF
PB_DOUT	GPIO_BA+0x048	R/W	GPIO PB Data Output Value	0x0000_003F

31	30	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
[15:0]	**DOUT**	**Port [A/B] Pin[N] Output Value** Each of these bits controls the status of a GPIO pin when the GPIO pin is configured as output, open-drain or quasi-bidirectional mode. 0 = GPIO port [A/B] Pin[n] will drive Low if the corresponding output mode bit is set. 1 = GPIO port [A/B] Pin[n] will drive High if the corresponding output mode bit is set. **Note**: PB_DOUT[15:6] are reserved to 0.

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

Register	Offset	R/W	Description	Reset Value
PA_PIN	GPIO_BA+0x010	R	GPIO PA Pin Value	0x0000_XXXX
PB_PIN	GPIO_BA+0x050	R	GPIO PB Pin Value	0x0000_00XX

31	30	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
[15:0]	**PIN[n]**	**Port [A/B] Pin[N] Pin Values** Each bit of the register reflects the actual status of the respective Px.n pin. If the bit is 1, it indicates the corresponding pin status is high; else the pin status is low. **Note**: PB_PIN[15:6] are reserved to 0.

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

Register	Offset	R/W	Description	Reset Value
PA_INTTYPE	GPIO_BA+0x018	R/W	GPIO PA Interrupt Trigger Type	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	GPIO PB Interrupt Trigger Type	0x0000_0000

31	30	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
[15:0]	**TYPE[n]**	**Port [A/B] Pin[N] Edge Or Level Detection Interrupt Trigger Type Control** TYPE[n] is used to control whether the interrupt mode is level triggered or edge triggered. If the interrupt mode is level triggered, the input source is sampled by one HCLK clock to generate the interrupt 0 = Edge triggered interrupt. 1 = Level triggered interrupt. **Note:** If level triggered interrupt is selected, then only one level can be selected in the Px_INTEN register. If both levels are set, the setting is ignored and no interrupt will occur

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

Register	Offset	R/W	Description	Reset Value
PA_INTEN	GPIO_BA+0x01C	R/W	GPIO PA Interrupt Enable	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	GPIO PB Interrupt Enable	0x0000_0000

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-level interrupt for each of the corresponding input Px.n pin. To set "1" also enables the pin wake-up function. When setting the RHIEN[n] (Px_INTEN[n+16]) bit to 1 : If the interrupt is configured as level trigger mode (TYPE[n] is set to 1), one interrupt will occur while the input Px.n state is at high level. If the interrupt is configured as edge trigger mode (TYPE[n] is set to 0), one interrupt will occur while he input Px.n state changes from low to high. 0 = Disable Px.n for low-to-high or level-high interrupt. 1 = Enable Px.n for low-to-high or level-high interrupt.

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

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

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

31	30	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
[15:0]	**INTSRC[n]**	**Port [A/B] Interrupt Source Flag** Read operation: 0 = No interrupt from Px.n. 1 = Px.n generated an interrupt. Write operation: 0 = No action. 1 = Clear the corresponding pending interrupt.

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

Register	Offset	R/W	Description	Reset Value
PAn_PDIO n=0,1..15	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO n=0,1..5	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PDIO**	**GPIO Px.n Pin Data Input/Output** Writing this bit can control one GPIO pin output value. 0 = Corresponding GPIO pin set to low. 1 = Corresponding GPIO pin set to high. Read this register to get GPIO pin status. For example, writing PA0_PDIO will reflect the written value to bit DOUT (Px_DOUT[0]), reading PA0_PDIO will return the value of PIN (PA_PIN[0]). **Note1:** The writing operation will not be affected by register DATMSK (Px_DATMSK[n]). **Note2:** Max. n=15 for port A. Max. n=5 for port B.

5.6 PWM Generator and Capture Timer

5.6.1 Introduction

I91032 has 2 sets of PWM timers. The PWM timer has 1 prescaler, 1 clock divider, 1 clock selector with 5 input sources, one 16-bit counter, four 16-bit comparators, and two Dead-Zone generators.

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

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

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

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

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

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

5.6.2 Features

- One 8-bit prescaler and clock divider
- Five clock inputs selector
- One 16-bit counter and four 16-bit comparators
- Two Dead-Zone generators
- Four PWM outputs, one capture function

5.6.3 PWM Generator Architecture

The following figures illustrate the architecture of the PWM generator.

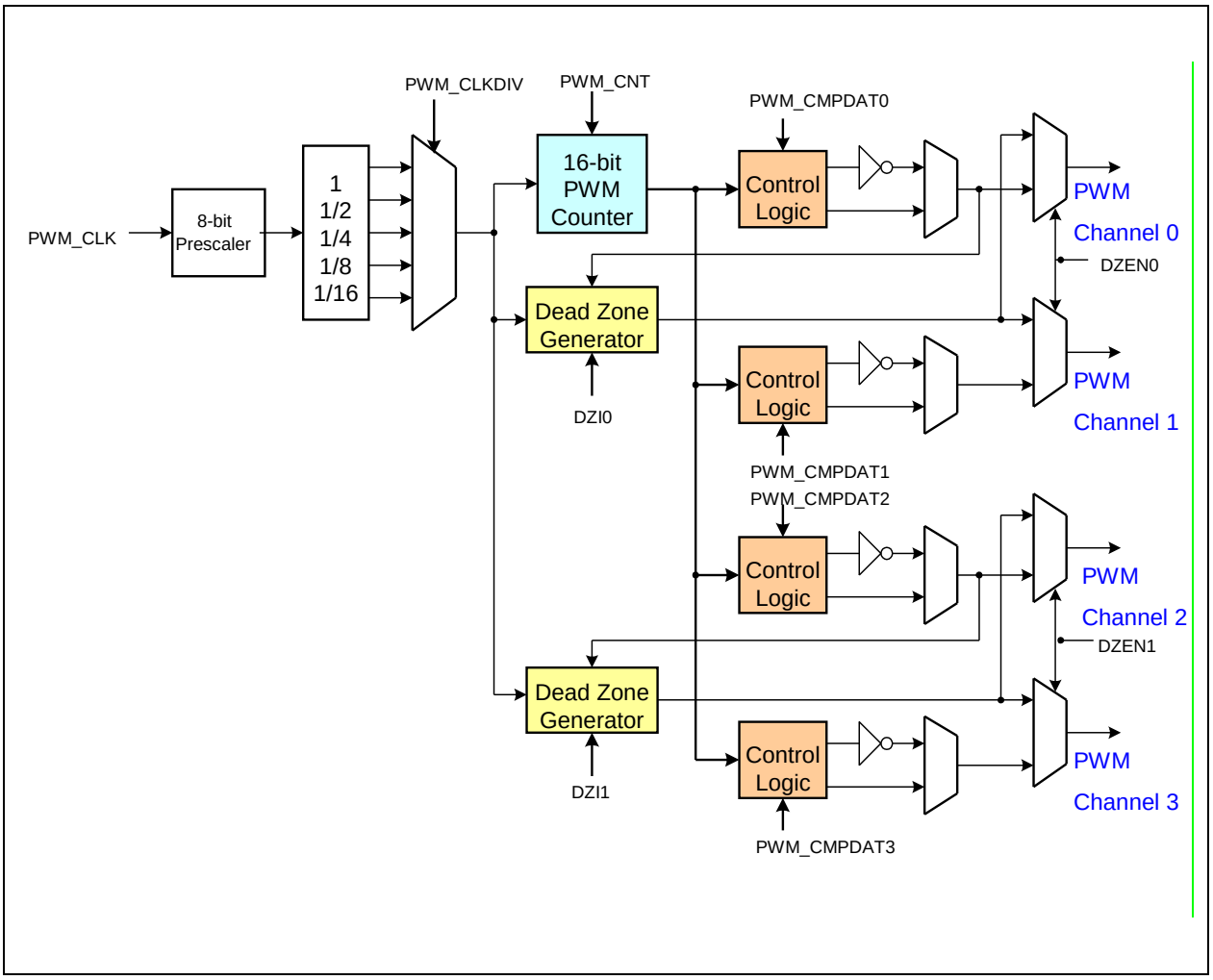


Figure 5-6 PWM Generator Architecture Diagram

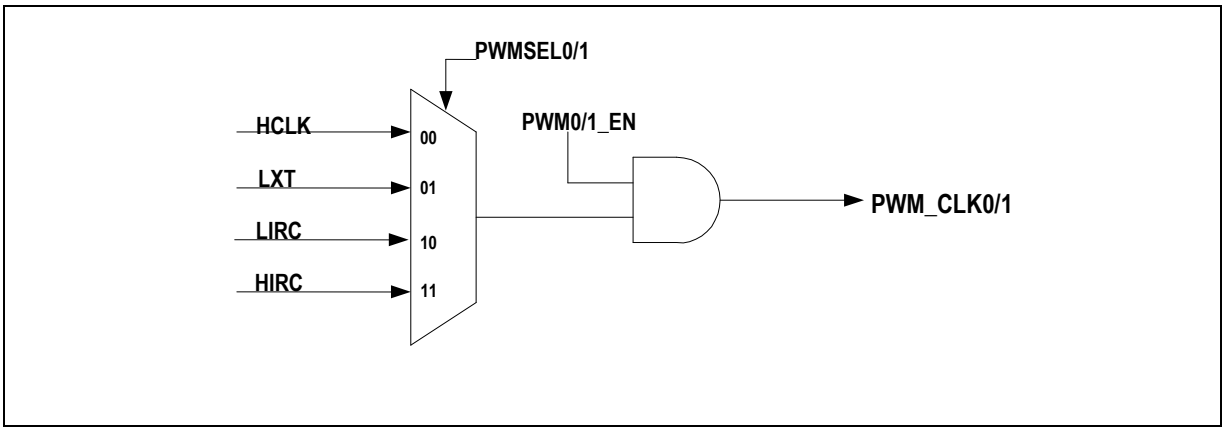


Figure 5-6 PWM Generator Clock Source Control

5.6.4 PWM-Timer Operation

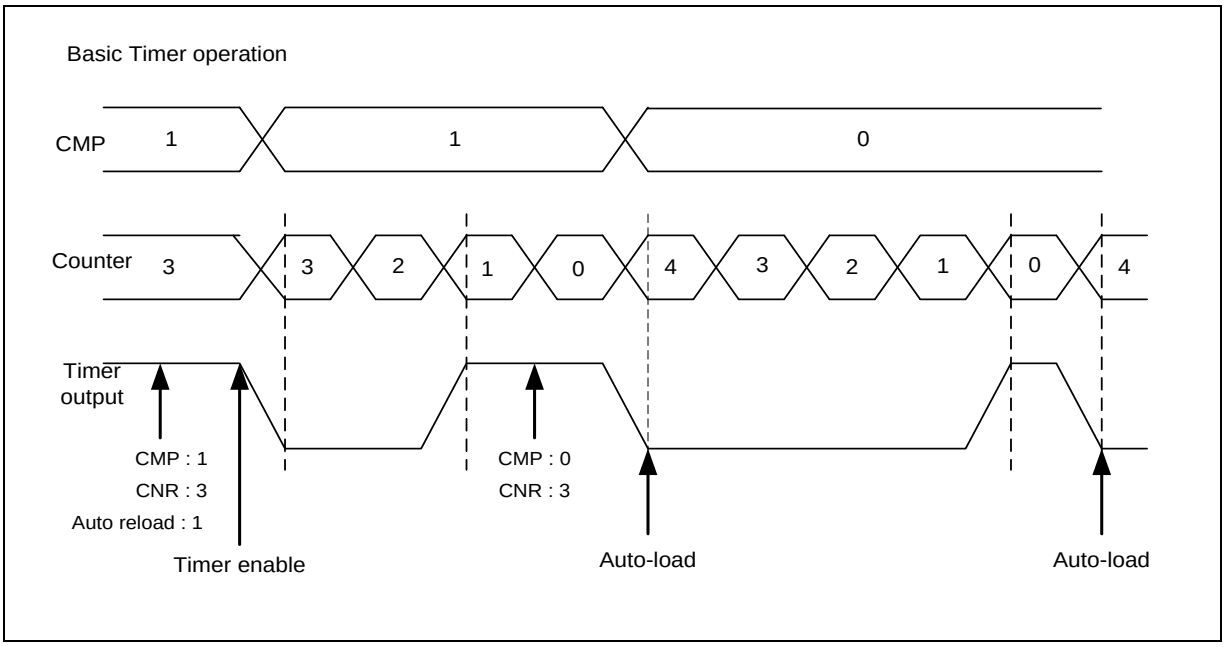


Figure 5-6 PWM Timer Operation Timing

5.6.5 PWM Auto-reload

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

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

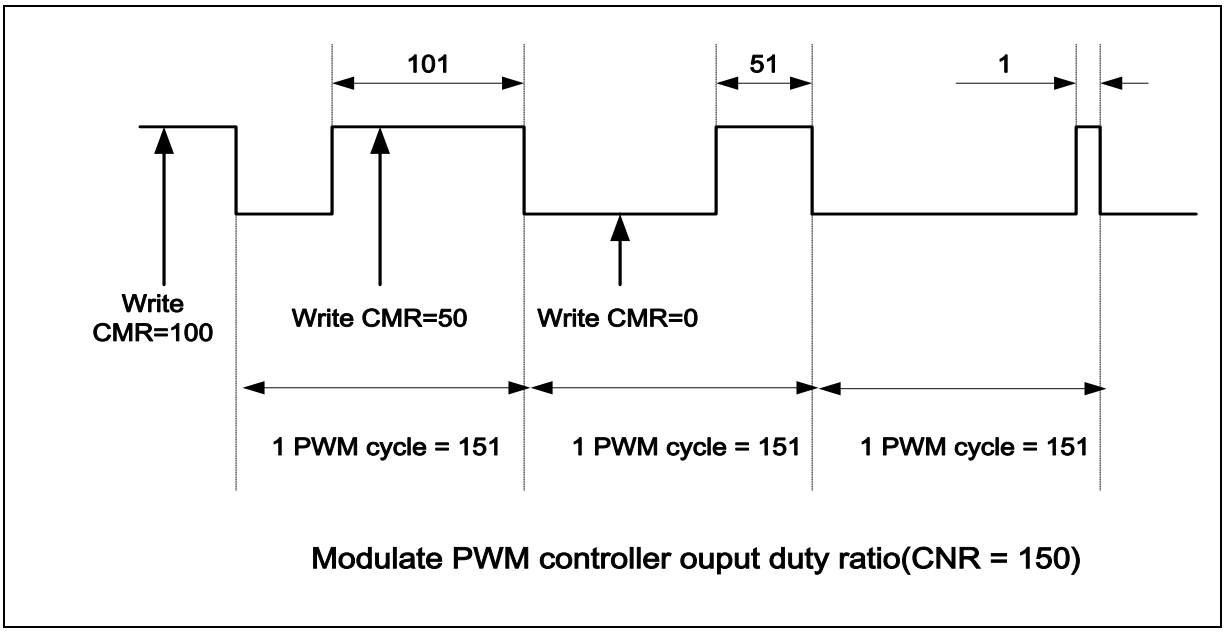


Figure 5-7 PWM Controller Output Duty Ratio.

5.6.6 Dead-Zone Generator

I91032 PWM is implemented with Dead Zone generator. They are built for power device protection. This function enables generation of a programmable time gap at the rising of PWM output waveform. User can program PWM_CLKPSC[31:24] and PWM_CLKPSC[23:16] to determine the two Dead Zone interval respectively.

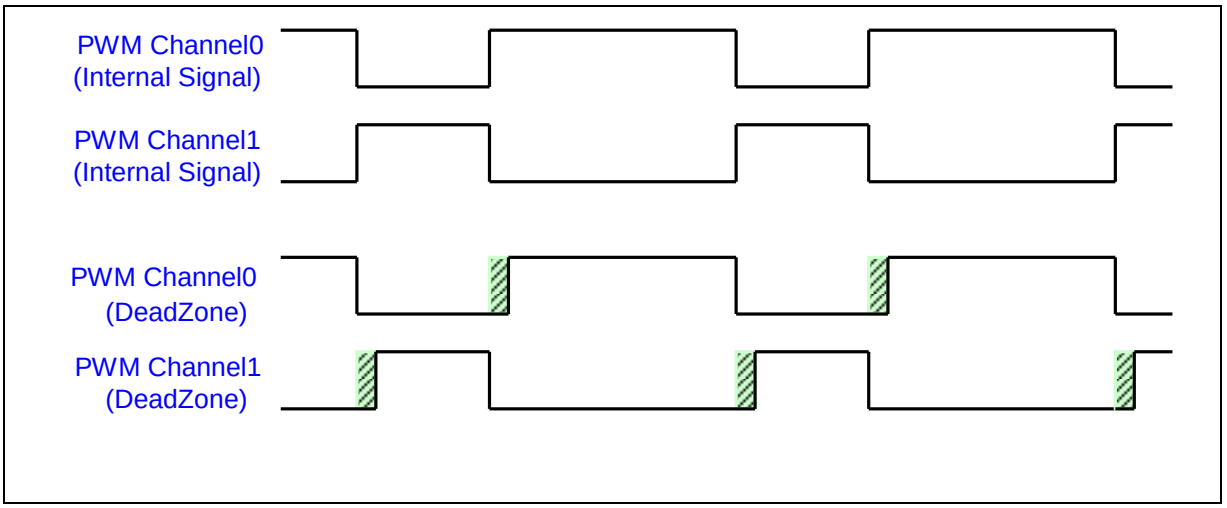


Figure 5-8 Dead Zone Generation Operation

5.6.7 PWM-Timer Start Procedure

The following procedure is recommended for starting a PWM generator.

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

5.6.8 PWM-Timer Stop Procedure

Method 1:

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

Method 2:

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

Method 3:

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

5.6.9 Capture Start Procedure

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

5.6.10 Capture Timer Operation

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

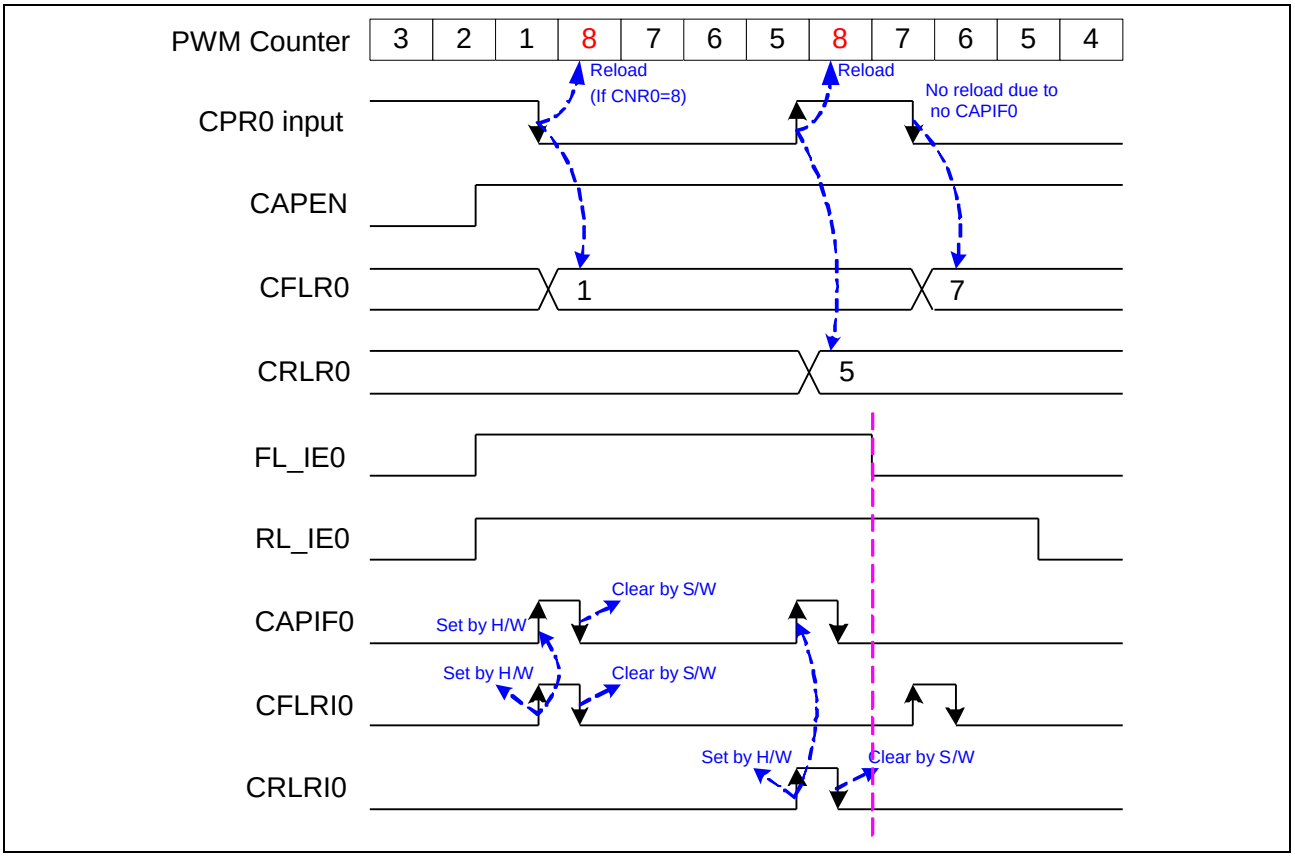


Figure 5-9 Capture Operation Timing

Figure 5-9 demonstrates the case where PERIOD = 8:

1. The PWM counter will be reloaded with PERIOD=8 at the time of interrupt occurrence when both falling and rising interrupt enabled.
2. The channel low pulse width is given by (PERIOD - RCAPDAT).
3. The channel high pulse width is given by (PERIOD - FCAPDAT).

5.6.11 Register Map

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

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

5.6.12 Register Description

PWM Pre-Scale Register (PWM_CLKPSC)

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

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

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

PWM Clock Select Register (PWM_CLKDIV)

Register	Offset	R/W	Description	Reset Value
PWM_CLKDIV	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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**CLKDIV**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**CLKDIV**	**PWM Timer Clock Source Selection** Value : Input clock divided by 000 : 2 001 : 4 010 : 8 011 : 16 1xx : 1

PWM Control Register (PWM_CTL)

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

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**DTEN1**	**Dead-Zone 1 Generator Enable/Disable** 0 = Disable. 1 = Enable.
[4]	**DTEN0**	**Dead-Zone 0 Generator Enable/Disable** 0 = Disable. 1 = Enable.
[3]	**CNTMODE**	**PWM-Timer Auto-Reload/One-Shot Mode** 0 = One-Shot Mode. 1 = Auto-reload Mode.
[2]	**PINV**	**PWM-Timer Output Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON.
[0]	**CNTEN**	**PWM-Timer Enable** 0 = Stop PWM-Timer Running. 1 = Enable PWM-Timer.

PWM Period Register (PWM_PERIOD)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**PWM Counter/Timer Reload Value** PERIOD determines the PWM period. **Note:** One PWM cycle width = (PERIOD + 1).

PWM Comparator Register 3-0 (PWM_CMPDAT3-0)

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

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

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

PWM Counter Register (PWM_CNT)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNT**	**PWM Counter Register** Reports the current value of the 16-bit down counter.

PWM Interrupt Enable Register (PWM_INTEN)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN	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							**PIEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PIEN**	**PWM Timer Interrupt Enable** 0 = Disable. 1 = Enable.

PWM Interrupt Flag Register (PWM_INTSTS)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS	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							**PIF**

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

Capture Control Register (PWM_CAPCTL)

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CFLIF**	**PWM_FCAPDAT Latched Indicator Bit** When input channel has a **falling** transition, PWM_FCAPDAT was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[6]	**CRLIF**	**PWM_RCAPDAT Latched Indicator Bit** When input channel has a **rising** transition, PWM_RCAPDAT was latched with the value of PWM down-counter and this bit is set by hardware, software can clear this bit by writing a zero to it.
[5]	**Reserved**	Reserved.
[4]	**CAPIF**	**Capture Interrupt Indication Flag** When capture input has a falling/rising transition and falling/rising latch interrupt is enabled (CFLIEN = 1/CRLIEN = 1), CAPIF0 is set 1 by hardware. Software can clear this bit by writing a one to it. **Note:** If this bit is "1", PWM counter will not be reloaded when next capture interrupt occurs.
[3]	**CAPEN**	**Capture Channel Input Transition Enable/Disable** 0 = Disable capture function. 1 = Enable capture function. When enabled, Capture function latches the PMW-counter to RCAPDAT (Rising latch) and FCAPDAT (Falling latch) registers on input edge transition. When disabled, Capture function is inactive as is interrupt.

[2]	**CFLIEN**	**Falling Latch Interrupt Enable ON/OFF** 0 = Disable falling latch interrupt. 1 = Enable falling latch interrupt. When enabled, capture block generates an interrupt on falling edge of input.
[1]	**CRLIEN**	**Rising Latch Interrupt Enable ON/OFF** 0 = Disable rising latch interrupt. 1 = Enable rising latch interrupt. When enabled, capture block generates an interrupt on rising edge of input.
[0]	**CAPINV**	**Inverter ON/OFF** 0 = Inverter OFF. 1 = Inverter ON. Reverse the input signal from GPIO before fed to Capture timer

Capture Rising Latch Register (PWM_RCAPDAT)

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

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

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

Capture Falling Latch Register (PWM_FCAPDAT)

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

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

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

PWM Output and Capture Input Enable Register (PWM_PCEN)

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

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CAPINEN**	**Capture Input Enable Register** 0 = OFF (PA.7 or PB.4 pin input disconnected from Capture block). 1 = ON (PA.7 or PB.4 pin, if in PWM alternative function, will be configured as an input and fed to capture function).
[7:4]	**Reserved**	Reserved.
[3]	**POEN3**	**PWM3 Output Enable Register** 0 = Disable PWM3 output to pin. 1 = Enable PWM3 output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPB_MFP)
[2]	**POEN2**	**PWM2 Output Enable Register** 0 = Disable PWM2 output to pin. 1 = Enable PWM2 output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)
[1]	**POEN1**	**PWM1 Output Enable Register** 0 = Disable PWM1 output to pin. 1 = Enable PWM1 output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)

[0]	**POEN0**	**PWM0 Output Enable Register** 0 = Disable PWM0 output to pin. 1 = Enable PWM0 output to pin. **Note:** The corresponding GPIO pin also must be switched to PWM function (refer to SYS_GPA_MFP)

5.7 Real Time Clock (RTC)

5.7.1 Overview

This RTC is a counter and the clock source can only be LXT (32 KHz crystal oscillator) or LIRC (internal 10KHz RC oscillator). The RTC time-out frequency is programmable. During normal operation, the RTC time-out can generate an interrupt at a pre-programmed frequency. During power down mode, the RTC time-out could be one of wake-up source.

5.7.2 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_CTL	RTC_BA+0x000	R/W	RTC Control Register	0x0000_0000

5.7.3 Register Description

RTC Control Register (RTC_CTL)

Register	Offset	R/W	Description	Reset Value
RTC_CTL	RTC_BA+0x000	R/W	RTC 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		**RTIS**			**RTCE**	**RTIE**	**RTIF**

Bits	Description	
[31:6]	**Reserved**	Reserved.

[5:3]	**RTIS**	**RTC Timer Interval Select** These two bits select the timeout interval for the RTC. 000 = Time-out frequency is 0.25Hz,. 001 = Time-out frequency is 0.5Hz,. 010 = Time-out frequency is 1Hz,. 011 = Time-out frequency is 2Hz. 100 = Time-out frequency is 4Hz,. 101 = Time-out frequency is 8Hz,. 110 = Time-out frequency is 16Hz,. 111 = Time-out frequency is 32Hz.
[2]	**RTCE**	**RTC Enable** 0 = Disable RTC function. 1 = Enable RTC function.
[1]	**RTIE**	**RTC Interrupt Enable** 0 = Disable the RTC interrupt. 1 = Enable the RTC interrupt.
[0]	**RTIF**	**RTC Interrupt Flag** If the RTC interrupt is enabled, then the hardware will set this bit to indicate that the RTC interrupt has occurred. If the RTC interrupt is not enabled, then this bit indicates that a timeout period has elapsed. 0 = RTC interrupt does not occur. 1 = RTC interrupt occurs. **Note:** This bit is cleared by writing 1 to this bit.

5.8 Oscillator Frequency Measurement

5.8.1 Overview

The Oscillator Frequency Measurement block allows user to measure the frequency of the internal RC oscillators by comparing to an accurate oscillator such as the 32kHz crystal oscillator. This is simply a special purpose timer/counter as shown in Figure 5-10.

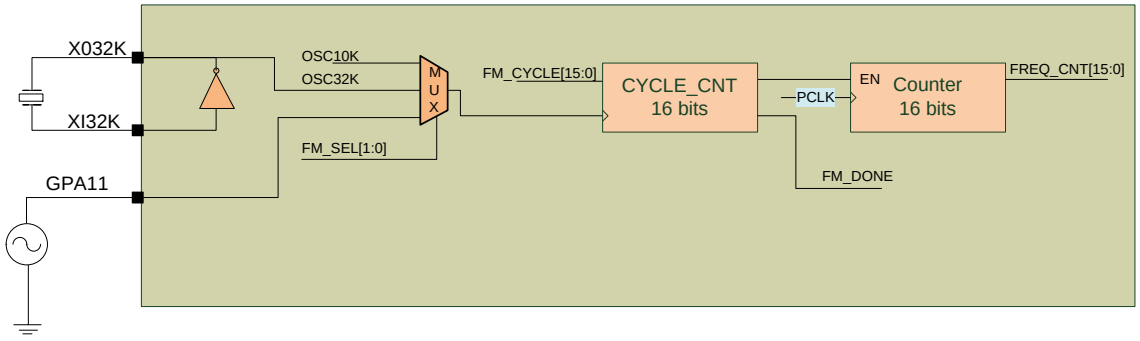


Figure 5-10 Oscillator Frequency Measurement Block Diagram

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 GPA[11] 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 LIRC 10kHz oscillator frequency relative to the internal master oscillator.

An example of use would be to measure the internal oscillator with reference to the 32768Hz crystal. To do this:

```
CLK->APBCLK.RTC_EN   = 1;                  // Turn on peripheral clock
OSCFM->OSCFM_CTL.CLK_FM_SEL = 1;   // Select reference source as 32kHz XTAL input
OSCFM->OSCFM_CYCLE = DRVOSC_NUM_CYCLES-1;
OSCFM->OSCFM_CTL.FM_GO = TRUE;
while ( (OSCFM->OSCFM_CTL.FM_DONE != 1) && (Timeout++ < 0x100000));
   if(   Timeout >= 0x100000)
         return(E_DRVOSC_MEAS_TIMEOUT);
Freq = OSCFM->OSCFM_CNT;
OSCFM->OSCFM_CTL.FM_GO = FALSE;
Freq = Freq*32768 /DRVOSC_NUM_CYCLES;
```

5.8.2 Register Map

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

Register	Offset	R/W	Description	Reset Value
OSCFM Base Address: OSCFM_BA = 0x4000_8004				
OSCFM_CTL	OSCFM_BA+0x000	R/W	Frequency Measurement Control Register	0x0000_0000
OSCFM_CNT	OSCFM_BA+0x004	R	Frequency Measurement Counter Register	0x0000_0000
OSCFM_CYCLE	OSCFM_BA+0x008	R/W	Frequency Measurement Cycle Register	0x0000_0000

5.8.3 Register Description

Frequency Measurement Control Register (OSCFM_CTL)

Register	Offset	R/W	Description	Reset Value
OSCFM_CTL	OSCFM_BA+0x004	R/W	Frequency Measurement Control Register	0x0000_0000

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

Bits	Description	
[31]	**FM_GO**	**FM GO BUSY** 0 --- no action 1 --- start to frequency measurement
[30:24]	**Reserved**	Reserved.
[23:16]	**FM_CYC[7:0]**	**FM Cycle number** Note: OSCFM_CYC[7:0] also can write this resigster
[15:3]	**Reserved**	**Reserved**
[2]	**FM_DONE**	**FM DONE FLAG** 0 = not done or no action. 1 = testing done, If FM_GO is “0”, clear the FM_DONE.
[0]	**CLK_FM_SEL**	**CLK FM SELETION** 2’b00 = source clock from CLK_10K. 2’b01 = source clock from CLK_32K. 2’b1x = source clock from GPA[11]. **Note:** This bit is cleared by writing 1 to this bit.

Frequency Measurement Counter Register (OSCFM_CNT)

Register	Offset	R/W	Description	Reset Value
OSCFM_CNT	OSCFM_BA+0x008	R	Frequency Measurement Counter Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**OSCFM_CNT**	**FM Counter Number** Report the counter

Frequency Measurement Cycle Register (OSCFM_CYCLE)

Register	Offset	R/W	Description	Reset Value
OSCFM_CYCLE	OSCFM_BA+0x00C	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**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FM_CYC[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FM_CYC[7:0]							

Bits	Description	
[30:16]	**Reserved**	Reserved.
[15:0]	**FM_CYC**	**FM iteration cycle number** **Note:** FM_CYC[7:0] can be overwritten by OSCFM_CTL[23:16]

5.9 Serial Peripheral Interface (SPI0) Controller

5.9.1 Overview

The Serial Peripheral Interface (SPI0) is a synchronous serial data communication protocol which operates in full duplex mode. Devices communicate in master/slave mode with 4-wire bi-directional interface. I91032 contains an SPI controller performing a serial-to-parallel conversion of data received from an external device, and a parallel-to-serial conversion of data transmitted to an external device. The SPI controller can be set as a master with up to 2 slave select (SSB) address lines to access two slave devices; it also can be set as a slave controlled by an off-chip master device.

5.9.2 Features

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

5.9.3 SPI0 Block Diagram

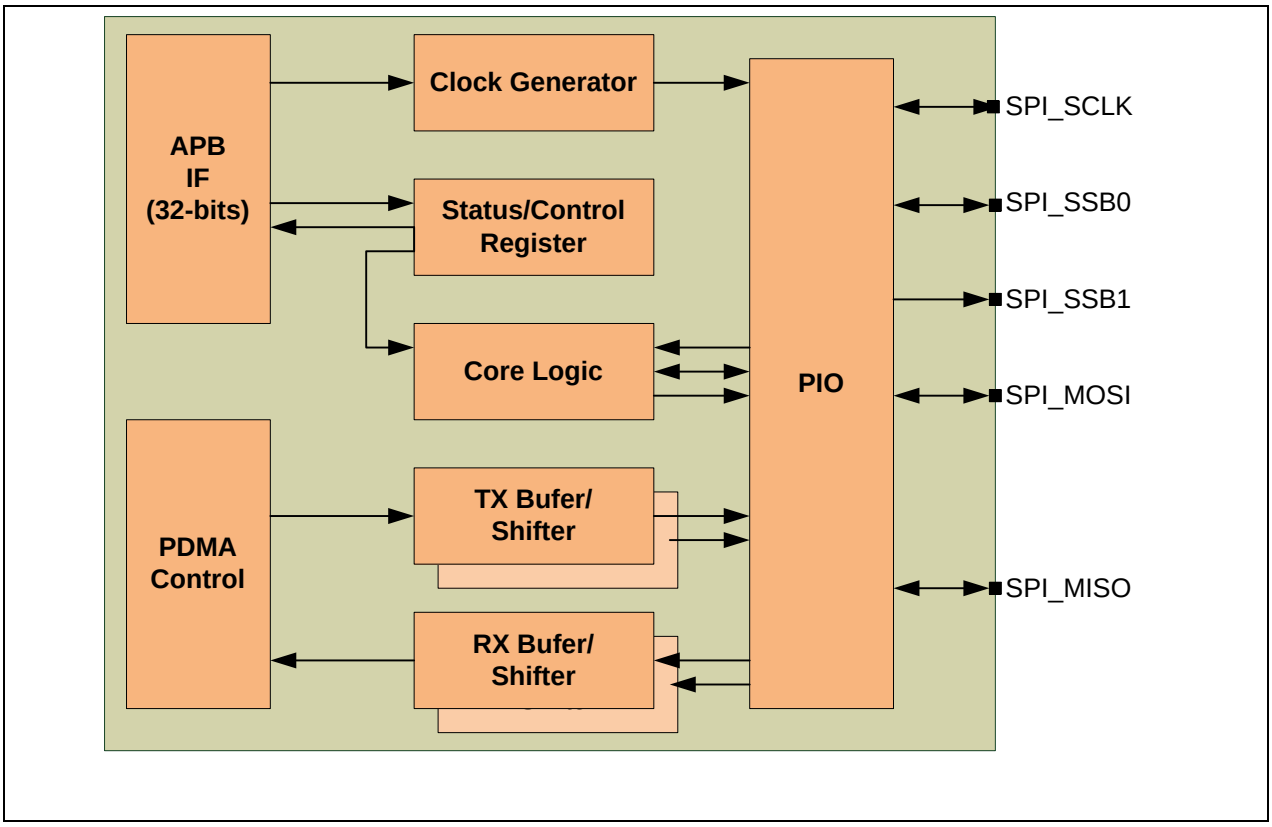


Figure 5-11 SPI Block Diagram

5.9.4 SPI0 Function Descriptions

- **Master/Slave Mode**

This SPI0 controller can be configured as in master or slave mode by setting the SLAVE bit (SPI->CNTRL.SLAVE). In master mode, it generates SCLK and SSB signals to access one or more slave devices. In slave mode, it monitors SCLK and SSB signals to respond to data transactions from an off-chip master. The signal directions are summarized in the application block diagrams of Figure 5-12 and Figure 5-13.

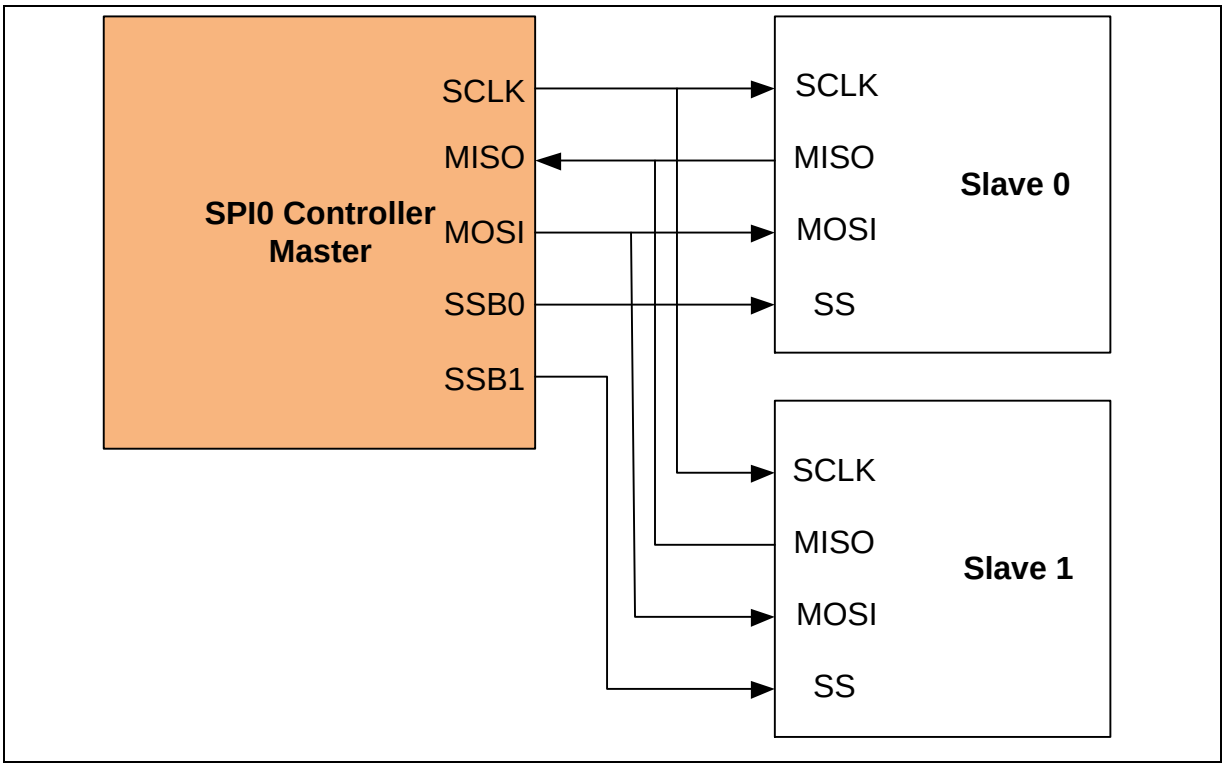


Figure 5-12 SPI0 Master Mode Application Block Diagram

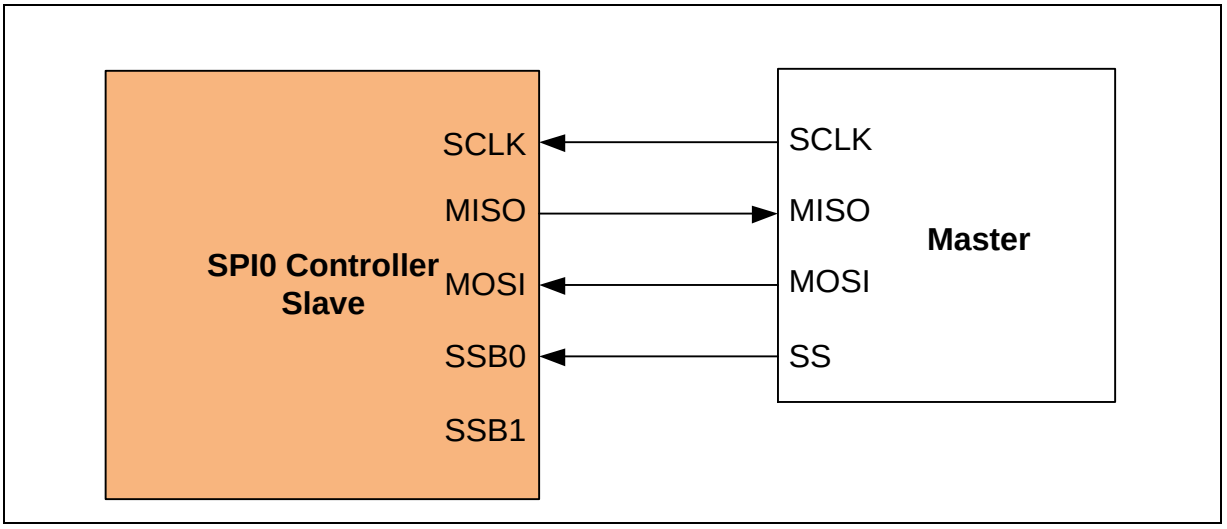


Figure 5-13 SPI0 Slave Mode Application Block Diagram

- **Slave Select**

In master mode, the SPI0 controller can address up to two off-chip slave devices through the slave select output pins SPI_SSB0 and SPI_SSB1. Only one slave can be addressed at any one time. If more slave address lines are required, GPIO pins can be manually configured to provide additional SSB lines. In slave mode, the off-chip master device drives the slave select signal SPI_SSB0 to address the SPI0 controller. The slave select signal can be programmed to be active low or active high via the SPI->SSR.SS_LVL bit. In addition the SPI->SSR.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->SSR.ASS is set, the slave select signals will be generated automatically and output to SPI_SSB0 and SPI_SSB1 pins according to registers SPI->SSR.SSR[0] and SPI->SSR.SSR[1]. In this mode, SPI controller will assert SSB when transaction is triggered and de-assert when data transfer is finished. If the SPI->SSR.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->SSR.SSR[1:0] register. The active level of the slave select output signals is specified by the SPI->SSR. SS_LVL bit.

- **Serial Clock**

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

- **Clock Polarity**

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

- **Transmit/Receive Bit Length**

The bit length of a transfer word is defined in SPI->CNTRL.TX_BIT_LEN bit field. It is set to define the length of a transfer word and can be up to 32 bits in length. TX_BIT_LEN=0x0 enables 32bit word length.

- **Burst Mode**

The SPI0 controller has a burst mode controlled by the SPI->CNTRL.TX_NUM field. If set to 0x01, SPI controller will burst two transactions from the TX[0] and TX[1] registers as shown in the waveform below:

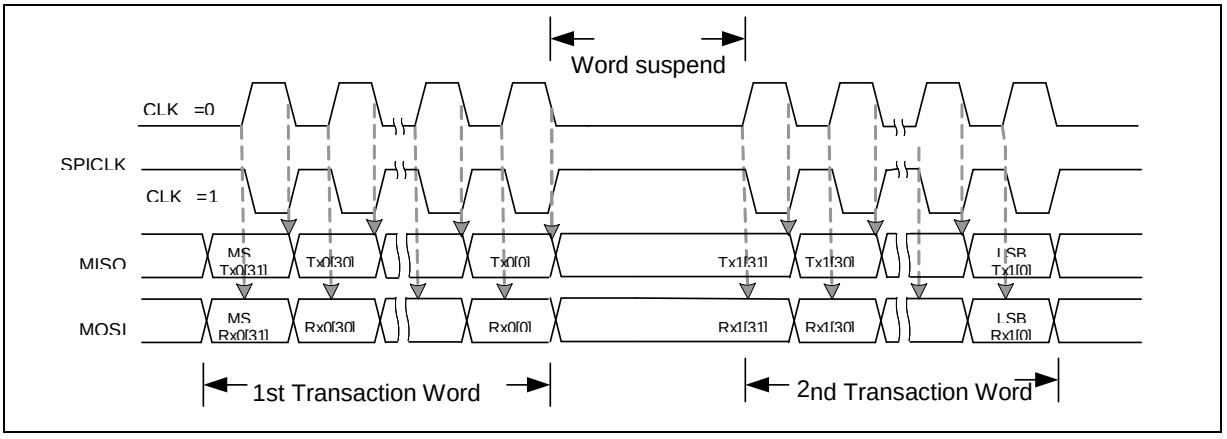


Figure 5-14 Two Transactions in One Transfer (Burst Mode)

- **LSB First**

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

- **Transmit Edge**

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

- **Receive Edge**

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

- **Word Sleep Suspend**

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

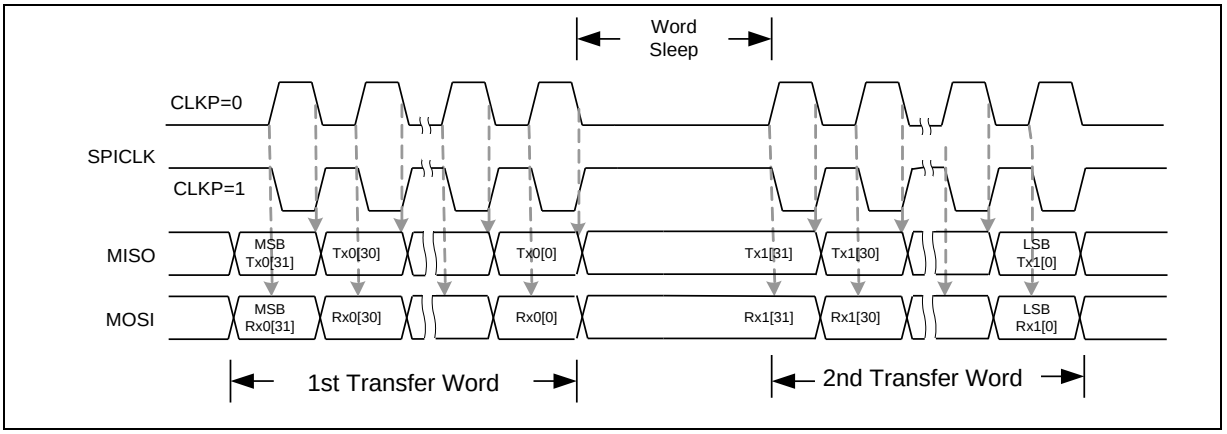


Figure 5-15 Word Sleep Suspend Mode

- **Byte Endian**

APB access to the SPI0 controller is via the 32 bit wide TX and RX registers. When the transfer is set as MSB first (SPI->CNTRL.LSB = 0) and the SPI->CNTRL.BYTE_ENDIAN bit is set, the data stored in the TX buffer and RX buffer will be rearranged such that the least significant **physical byte** is processed first. For TX_BIT_LEN =0 (32 bits transfer), the sequence of transmitted/received data will be BYTE0, BYTE1, BYTE2, and then BYTE3. If TX_BIT_LEN is set to 24-bits, the sequence will be BYTE0, BYTE1, and BYTE2. The rule of 16-bits mode is the same as above, see Figure 5-16.

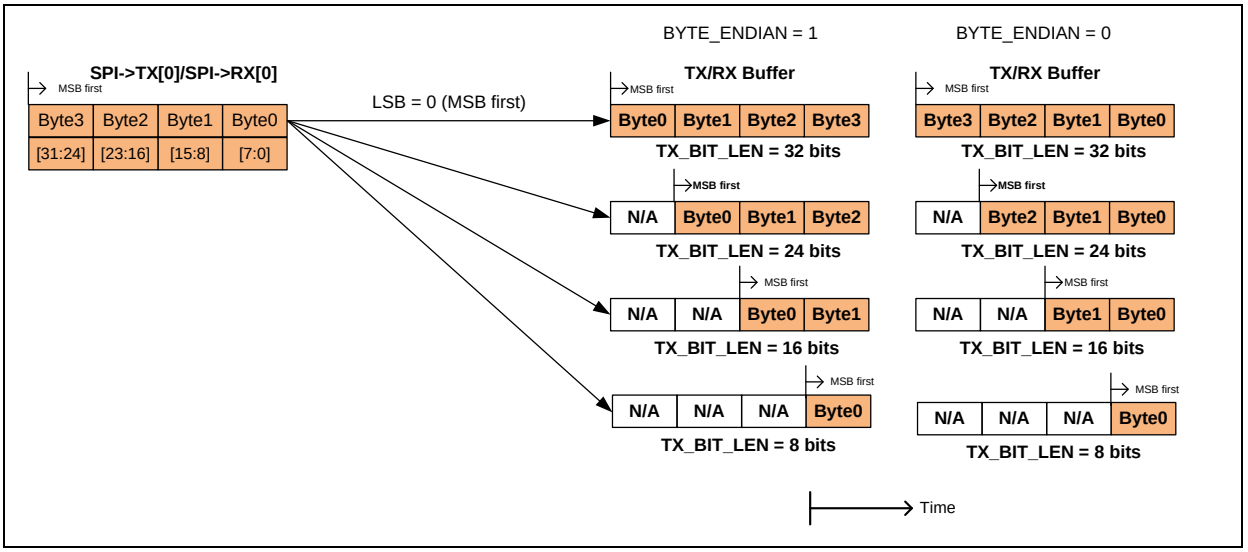


Figure 5-16 Byte Re-Ordering Transfer

Byte ordering can be a confusing issue when converting from arrays of data processed by the CPU for transmission out the SPI0 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 Cortex M0 stores the following arrays in memory:

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

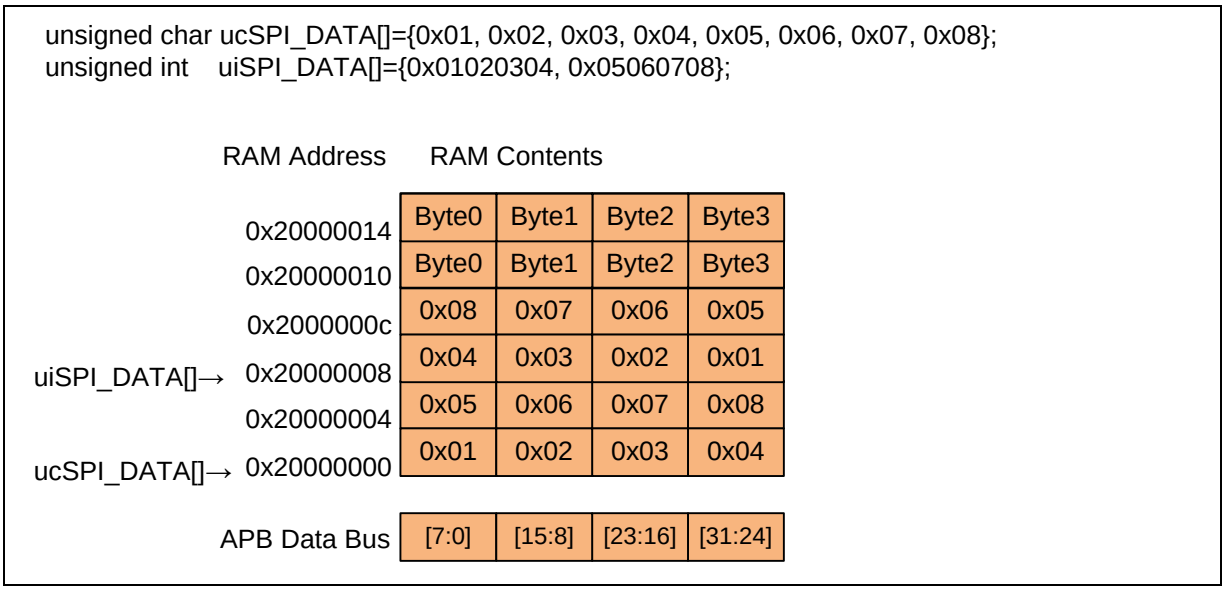


Figure 5-17 Byte Order in Memory

It can be seen from Figure 5-17 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 TX_BIT_LEN=8 and send data byte-by-byte SPI->TX[0] = ucSPI_DATA[i++]
2. Set TX_BIT_LEN=32 and send word-by-word SPI->TX[0] = uiSPI_DATA[i++]

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

It would be common that a byte array of data is constructed but user, for efficiency, wishes to transfer data to SPI0 via word transfers. Consider the situation of Figure 5-18 where a int pointer points to the byte data array.

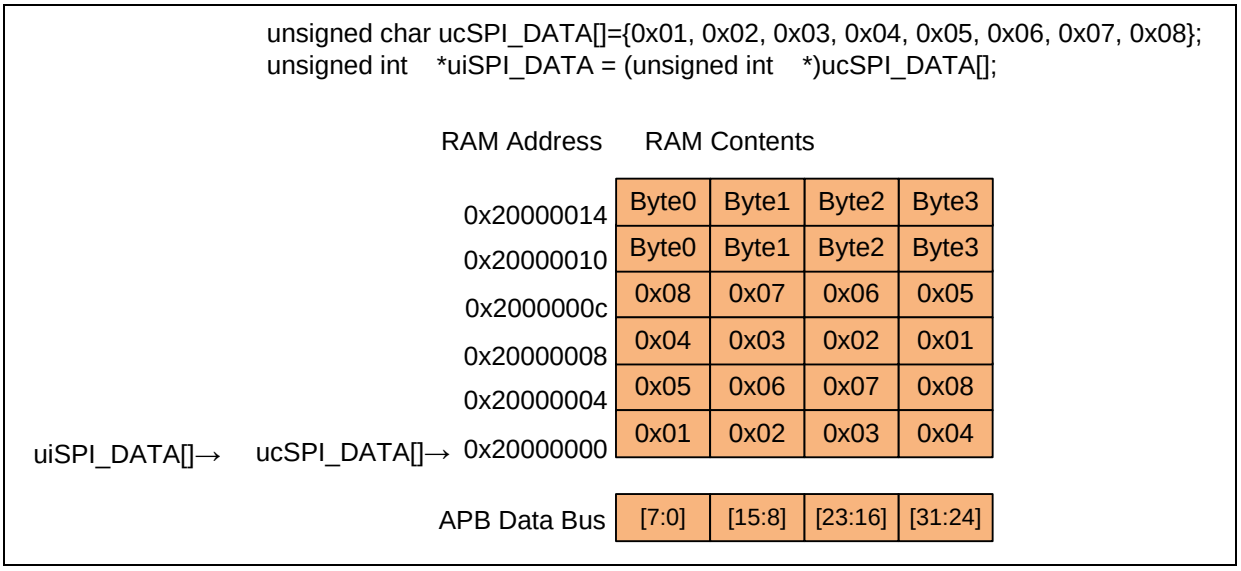


Figure 5-18 Byte Order in Memory

Now if we set TX_BIT_LEN=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 BYTE_ENDIAN=1, we would reverse this order to the desired stream: {0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08}.

- **Byte Sleep Suspend**

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

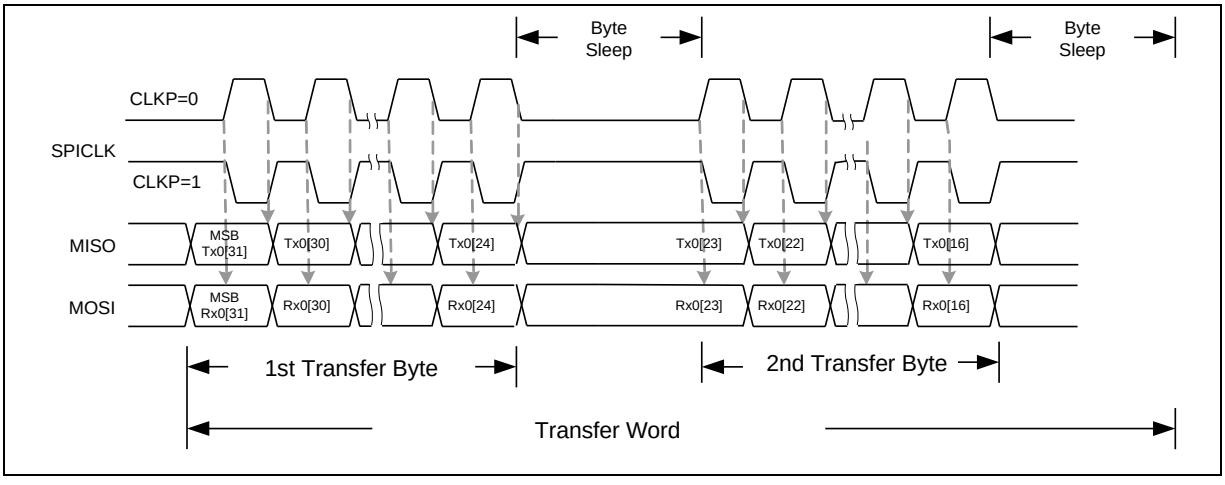


Figure 5-19 Byte Suspend Mode

- **Interrupt**

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

- **FIFO Mode**

The SPI0 controller supports a dual buffer mode when SPI->CNTRL.FIFO is set as 1. In normal mode, software can only update the transmitted data when the current transmission is done. In FIFO mode, the next transmitted data can be written into the SPI_TX buffer at any time when in master mode or the GO_BUSY bit is set in slave mode. This data will load into the transmit buffer when the current transmission done.

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

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

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

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

- **Variable Serial Clock Frequency**

In master mode 16 bit transfers, the output of serial clock can be programmed as variable frequency pattern if the Variable Clock Enable bit SPI->CNTRL.VARCLK_EN is enabled. The frequency pattern format is defined in SPI->VARCLK register. If the bit content of VARCLK is '0' the output period for that bit is determined by setting of SPI->DIVIDER.DIVIDER, if the bit content of VARCLK is '1', the output period for that bit is determined by the SPI->DIVIDER.DIVIDER2 register. The following figure shows the timing relationships of serial clock (SCLK), to the VARCLK, the DIVIDER and the DIVIDER2 registers. A two-bit combination in the VARCLK defines one clock cycle. The bit field VARCLK[31:30] defines the first clock cycle of SCLK. The bit field VARCLK[29:28] defines the second clock cycle of SCLK and so on. The clock source selections are defined in VARCLK and must be set 1 cycle before the next clock option. For example, if there are 5 CLK1 cycle in SPICLK, the VARCLK shall set 9 '0' in the MSB of VARCLK. The 10th shall be set as '1' in order to switch the next clock source is CLK2. Note that when VARCLK_EN bit is set, the setting of TX_BIT_LEN must be programmed as 0x10 (16 bits

mode only).

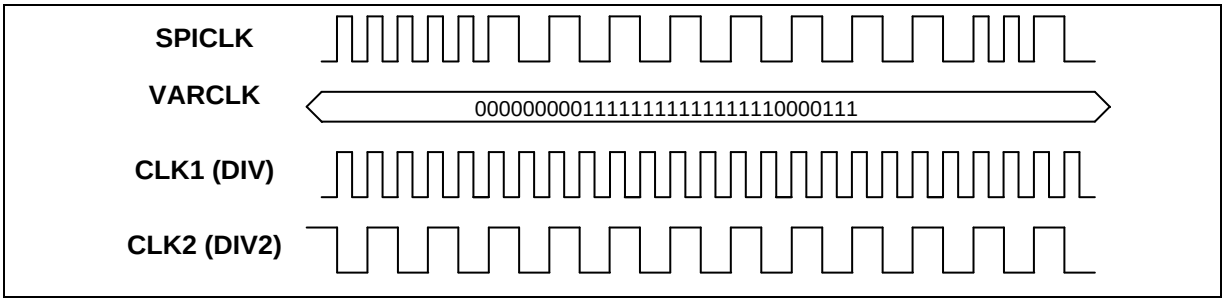


Figure 5-20 Variable Serial Clock Frequency

5.9.5 SPI0 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->SSR.SS_LVL bit. In slave mode, the SPI-SSR.SS_LTRIG will determine whether the slave select signal is treated as a level triggered or edge triggered signal.

The serial clock phase and polarity is controlled by CLKP, RX_NEG and TX_NEG bits. The bit length of a transfer word is configured by the TX_BIT_LEN parameter. Whether data transmission is MSB first or LSB first is controlled by the SPI->CNTRL.LSB bit. Four examples of SPI0 timing diagrams for master/slave operations and the related settings are shown as below.

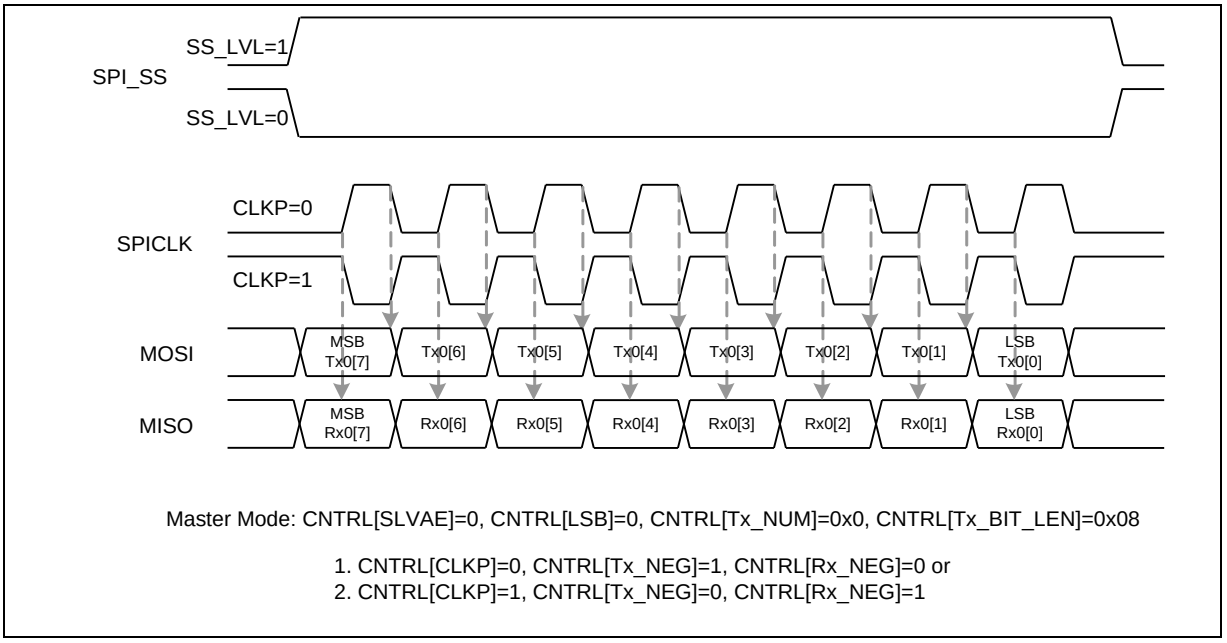


Figure 5-21 SPI0 Timing in Master Mode

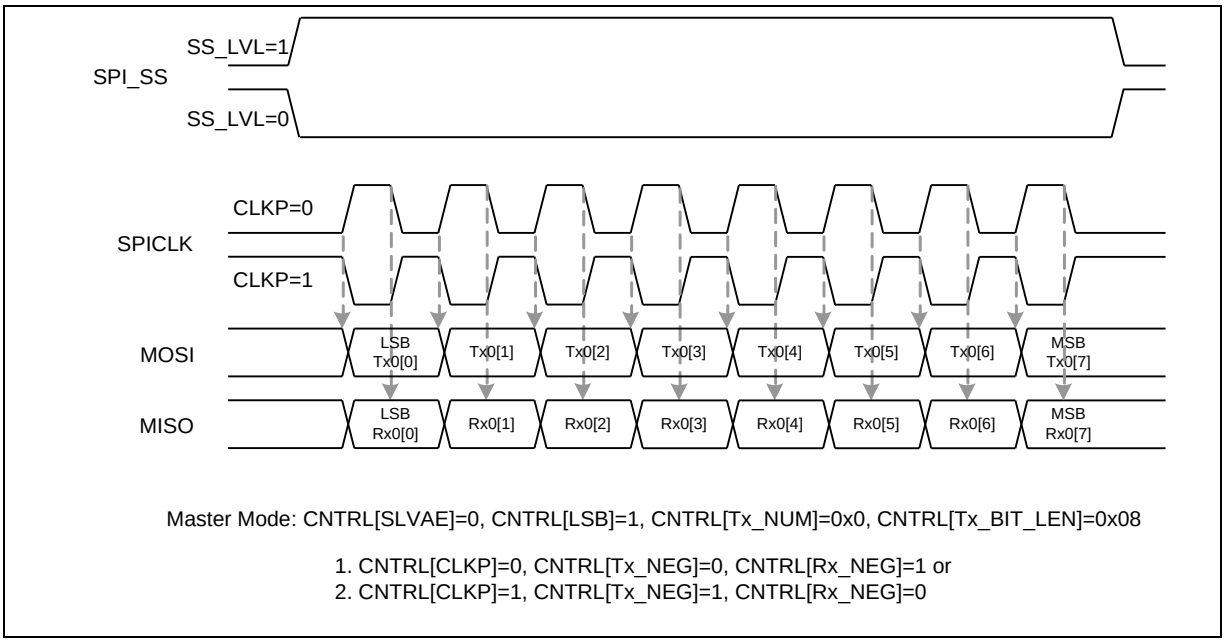


Figure 5-22 SPI0 Timing in Master Mode (Alternate Phase of SPICLK)

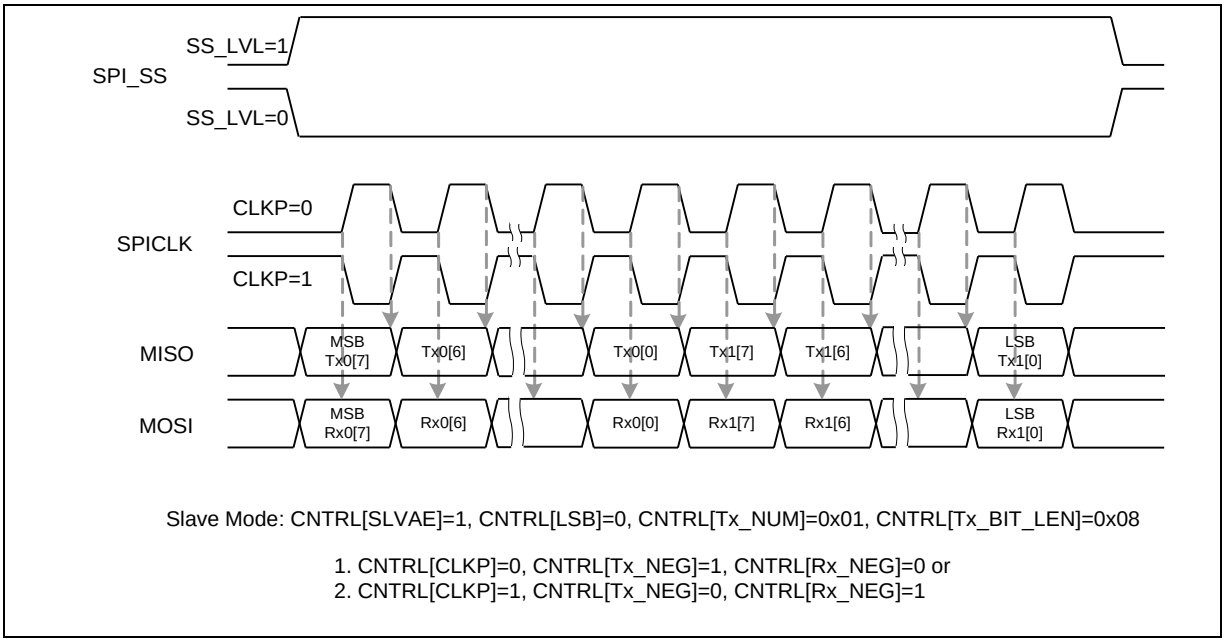


Figure 5-23 SPI0 Timing in Slave Mode

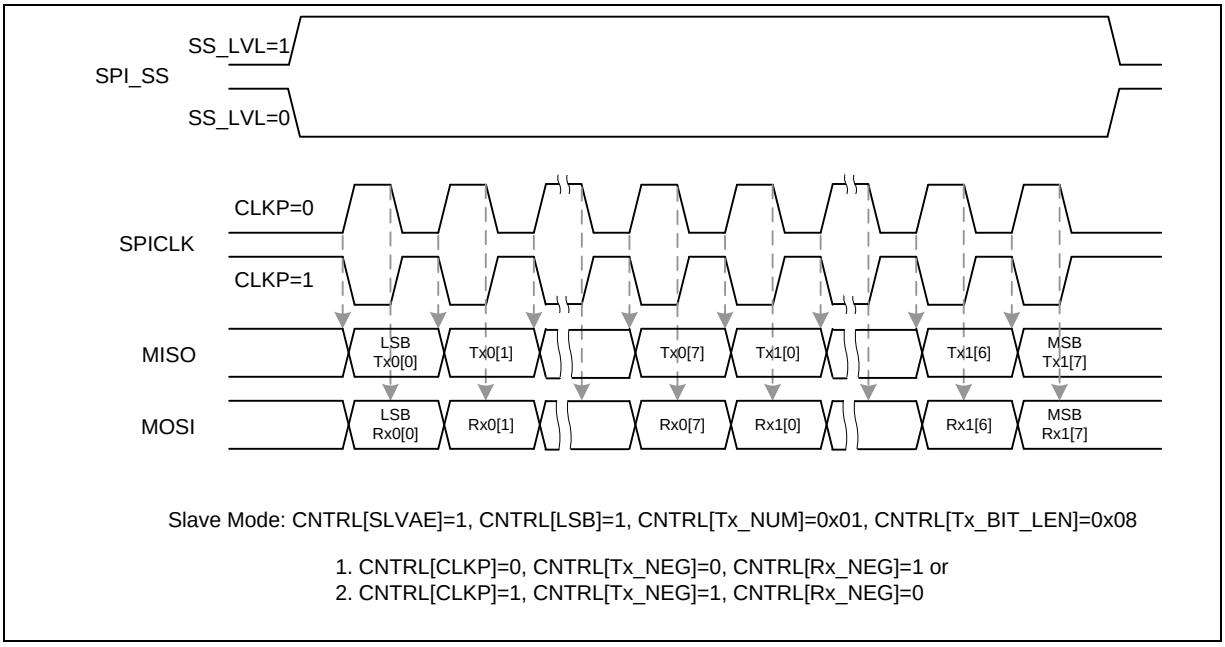


Figure 5-24 SPI0 Timing in Slave Mode (Alternate Phase of SPICLK)

5.9.6 SPI0 Configuration Examples

- Example 1, SPI0 controller is set as a master to access an off-chip slave device with following specifications:
  - Data bit latched on positive edge of serial clock
  - Data bit driven on negative edge of serial clock
  - Data be transferred from MSB first
  - SCLK low in idle state
  - Only one byte data be transmitted/received in a transfer
  - Slave select signal is active low
  - SCLK frequency is 10MHz

To configure the SPI0 interface to the above specifications perform the following steps:

1) Write a divisor into the SPI->DIVIDER 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->SSR register to address device. For example to manually address, set SPI-SSR.ASS=0, SPI->SSR.SSR_LVL=0 for active low SS. When software wishes to address device it will set SPI->SSR.SSR=1 to output an active SS on SPI_SSB0 pin.
3) Configure the SPI->CNTRL register. Set SPI->CNTRL.SLAVE=0 for master mode, set SPI->CNTRL.CLKP=0 for SCLK polarity normally low, set SPI->CNTRL.TX_NEG=1 so that data changes on falling edge of SCLK, set SPI->CNTRL.RX_NEG=0 so that data is latched into device on positive edge of SCLK, set SPI->CNTRL.TX_BIT_LEN=8 and SPI->CNTRL.TX_NUM=0 for a single byte transfer and finally set SPI->CNTRL.LSB=0 for MSB first transfer.
4) If manually selecting slave device set SPI->SSR.SSR=1.
5) To transmit one byte of data, write data to SPI->TX[0] register. If only doing a receive, write a dummy byte to SPI->TX[0] register.
6) Enable the SPI->CNTRL.GO_BUSY bit to start the data transfer over the SPI0 interface.

-- Wait for SPI0 transfer to finish. Can be interrupt driven (if the interrupt enable SPI->CNTRL.IE bit is set) or by polling the GO_BUSY bit which will be cleared to 0 by hardware automatically at end of transmission. --

7) Read out the received one byte data from SPI-> RX[0]
8) Go to 5) to continue another data transfer or set SPI->SSR.SSR=0 to deactivate the off-chip slave devices.

- Example 2, SPI0 controller is set as a slave device that controlled by an off-chip master device with the following characteristics:
  - Data bit latched on positive edge of serial clock
  - Data bit driven on negative edge of serial clock
  - Data be transferred from LSB first
  - SCLK high in idle state
  - Only one byte data be transmitted/received in a transfer
  - Slave select signal is active high level trigger

To configure the SPI0 interface to the above specifications perform the following steps:

1) Configure the SPI->SSR register. SPI->SSR.SS_LVL=1 for active high slave select, SPI->SSR.SSR.SS_LTRIG=1 for level sensitive trigger.
2) Configure the SPI->CNTRL register. Set SPI->CNTRL.SLAVE=1 for slave mode, set SPI->CNTRL.CLKP=1 for SCLK polarity idle high, set SPI->CNTRL.TX_NEG=1 so that data changes on falling edge of SCLK, set SPI->CNTRL.RX_NEG=0 so that data is latched into device on positive edge of SCLK, set SPI->CNTRL.TX_BIT_LEN=8 and SPI->CNTRL.TX_NUM=0 for a single byte transfer and finally set SPI->CNTRL.LSB=1 for LSB first transfer.
3) If SPI0 slave is to transmit one byte of data to the off-chip master device, write first byte to TX[0] register. If no data to be transmitted write a dummy byte.
4) Enable the GO_BUSY bit to wait for the slave select trigger input and serial clock input from the off-chip master device to start the data transfer at the SPI0 interface.

-- -- Wait for SPI0 transfer to finish. Can be interrupt driven (if the interrupt enable SPI->CNTRL.IE bit is set) or by polling the GO_BUSY bit which will be cleared to 0 by hardware automatically at end of transmission. --

5) Read out the received data from RX[0] register.
6) Go to 3) to continue another data transfer or disable the GO_BUSY bit to stop data transfer.

5.9.7 SPI0 Control Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI Base Address: SPI0_BA = 0x4003_0000				
SPI0_CTL	SPI0_BA+0x00	R/W	Control and Status Register	0x0000_0004
SPI0_CLKDIV	SPI0_BA+0x04	R/W	Clock Divider Register (Master Only)	0x0000_0000
SPI0_SSCTL	SPI0_BA+0x08	R/W	Slave Select Register	0x0000_0000
SPI0_RX0	SPI0_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPI0_RX1	SPI0_BA+0x14	R	Data Receive Register 1	0x0000_0000
SPI0_TX0	SPI0_BA+0x20	W	Data Transmit Register 0	0x0000_0000
SPI0_TX1	SPI0_BA+0x24	W	Data Transmit Register 1	0x0000_0000
SPI_VARCLK	SPI0_BA + 0x34	R/W	Variable Clock Pattern Register	0x007F_FF87
SPI_PDMACTL	SPI0_BA + 0x38	R/W	SPI PDMA Control Register	0x0000_0000

5.9.8 SPI0 Control Register Description

SPI0 Control and Status Register (SPI0_CTL)

Register	Offset	R/W	Description	Reset Value
SPI0_CTL	SPI0_BA+0x00	R/W	Control and Status Register	0x0000_0004

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

Bits	Description	
[31:28]	Reserved	Reserved
[28]	PDMASSEN	**Enable DMA Automatic SS function** When enabled, interface will automatically generate a SS signal for an entire PDMA access transaction.
[27]	TXFULL	**Transmit FIFO Full Status** 0 = The transmit data FIFO is not full. 1 = The transmit data FIFO is full.
[26]	TXEMPTY	**Transmit FIFO Empty Status** 0 = The transmit data FIFO is not empty. 1 = The transmit data FIFO is empty.
[25]	RXFULL	**Receive FIFO Full Status** 0 = The receive data FIFO is not full. 1 = The receive data FIFO is full.
[24]	RXEMPTY	**Receive FIFO Empty Status** 0 = The receive data FIFO is not empty. 1 = The receive data FIFO is empty.

[23]	**VARCLKEN**	**Variable Clock Enable (Master Only)** 0 = The serial clock output frequency is fixed and determined only by the value of DIVIDER0. 1 = SCLK output frequency is variable. The output frequency is determined by the value of SPI_VARCLK, DIVIDER0, and DIVIDER1. Note that when enabled, the setting of DWIDTH must be programmed as 0x10 (16 bits mode)
[21]	**FIFOEN**	**FIFO Mode** 0 = No FIFO present on transmit and receive buffer. 1 = Enable FIFO on transmit and receive buffer.
[20]	**REORDER**	**Byte Endian Reorder Function** This function changes the order of bytes sent/received to be least significant physical byte first.
[19]	**BYTEITV**	**Insert Sleep Interval Between Bytes** This function is only valid for 32bit transfers (DWIDTH aaa 0). If set then a pause of (SUSPITV+2) SCLK cycles is inserted between each byte transmitted.
[18]	**SLAVE**	**Master Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**UNITIEN**	**Interrupt Enable** 0 = Disable SPI Interrupt. 1 = Enable SPI Interrupt to CPU.
[16]	**UNITIF**	**Interrupt Flag** 0 = Indicates the transfer is not finished yet. 1 = Indicates that the transfer is complete. Interrupt is generated to CPU if enabled. NOTE: This bit is cleared by writing 1 to itself.
[15:12]	**SUSPITV**	**Suspend Interval (Master Only)** These four bits provide configurable suspend interval between two successive transmit/receive transactions in a transfer. The suspend interval is from the last falling clock edge of the current transaction to the first rising clock edge of the successive transaction if CLKPOL aaa 0. If CLKPOL aaa 1, the interval is from the rising clock edge to the falling clock edge. The default value is 0x0. When TXCNT aaa 00b, setting this field has no effect on transfer except as determined by REORDER[0] setting. The suspend interval is determined according to the following equation: (SUSPITV[3:0] + 2) * period of SCLK

[11]	**CLKPOL**	**Clock Polarity** 0 = SCLK idle low. 1 = SCLK idle high.
[10]	**LSB**	**LSB First** 0 = The MSB is transmitted/received first (which bit in SPI_TX0/1 and SPI_RX0/1 register that is depends on the DWIDTH field). 1 = The LSB is sent first on the line (bit 0 of SPI_TX0/1), and the first bit received from the line will be put in the LSB position in the Rx register (bit 0 of SPI_RX0/1).
[9:8]	**TXNUM**	**Transmit/Receive Word Numbers** This field specifies how many transmit/receive word numbers should be executed in one transfer. 00 = Only one transmit/receive word will be executed in one transfer. 01 = Two successive transmit/receive word will be executed in one transfer. 10 = Reserved. 11 = Reserved.
[7:3]	**DWIDTH**	**Transmit Bit Length** This field specifies how many bits are transmitted in one transmit/receive. Up to 32 bits can be transmitted. DWIDTH aaa 0x01 --- 1 bit DWIDTH aaa 0x02 --- 2 bits ---- DWIDTH aaa 0x1f --- 31 bits DWIDTH aaa 0x00 --- 32 bits
[2]	**TXNEG**	**Transmit At Negative Edge** 0 = The transmitted data output signal is changed at the rising edge of SCLK. 1 = The transmitted data output signal is changed at the falling edge of SCLK.
[1]	**RXNET**	**Receive At Negative Edge** 0 = The received data input signal is latched at the rising edge of SCLK. 1 = The received data input signal is latched at the falling edge of SCLK.

[0]	**GOBUSY**	**Go and Busy Status** 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit starts the transfer. This bit remains set during the transfer and is automatically cleared after transfer finished. NOTE: All registers should be set before writing 1 to this BUSY bit. When a transfer is in progress, writing to any register of the SPI master/slave core has no effect.

SPI0 Divider Register (SPI0_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SPI0_CLKDIV	SPI0_BA+0x04	R/W	Clock Divider Register (Master Only)	0x0000_0000

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

Bits	Description	
[31:16]	DIVIDER1	**Clock Divider 2 Register** (master only) The value in this field is the 2nd frequency divider of the system clock, PCLK, to generate the serial clock on the output SCLK. The desired frequency is obtained according to the following equation: Fsclk = Fpclk / ((DIVIDER1+1) * 2)
[15:0]	DIVIDER0	**Clock Divider Register** (master only) The value in this field is the frequency division of the system clock, PCLK, to generate the serial clock on the output SCLK. The desired frequency is obtained according to the following equation: Fsclk = Fpclk / ((DIVIDER0+1) * 2) In slave mode, the period of SPI clock driven by a master shall satisfy Fsclk <= (Fpclk / 5) In other words, the maximum frequency of SCLK clock is one fifth of the SPI peripheral clock.

SPI Slave Select Register (SPI0_SSCTL)

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

31	30	29	28	27	26	25	24
Reserved							
23	22	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		LVTRGSTS	LVTRGEN	AUTOSS	SSACTPOL	SSR[1:0]	

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

[2]	**SSACTPOL**	**Slave Select Active Level** It defines the active level of device/slave select signal (SPISSx0/1). 0 = The slave select signal SPISSx0/1 is active at low-level/falling-edge. 1 = The slave select signal SPISSx0/1 is active at high-level/rising-edge.
[1:0]	**SS**	**Slave Select Register (Master only)** If AUTOSS bit is cleared, writing 1 to any bit location of this field sets the proper SPISSx0/1 line to an active state and writing 0 sets the line back to inactive state. If AUTOSS bit is set, writing 1 to any bit location of this field will select appropriate SPISSx0/1 line to be automatically driven to active state for the duration of the transmit/receive, and will be driven to inactive state for the rest of the time. (The active level of SPISSx0/1 is specified in SSACTPOL). **Note:** SPISSx0 is always defined as device/slave select input signal in slave mode.

SPI Data Receive Register (SPI0_RXn)

Register	Offset	R/W	Description	Reset Value
SPI0_RX0	SPI0_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPI0_RX1	SPI0_BA+0x14	R	Data Receive Register 1	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**	**Data Receive Register** The Data Receive Registers hold the value of received data of the last executed transfer. Valid bits depend on the transmit bit length field in the SPI_CNTRL register. For example, if Tx_BIT_LEN is set to 0x08 and Tx_NUM is set to 0x0, bit Rx0[7:0] holds the received data. **NOTE:** The Data Receive Registers are read only registers.

SPI Data Transmit Register (SPI0_TXn)

Register	Offset	R/W	Description	Reset Value
SPI0_TX0	SPI0_BA+0x20	W	Data Transmit Register 0	0x0000_0000
SPI0_TX1	SPI0_BA+0x24	W	Data Transmit Register 1	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**	**Data Transmit Register** The Data Transmit Registers hold the data to be transmitted in the next transfer. Valid bits depend on the transmit bit length field in the CNTRL register. For example, if Tx_BIT_LEN is set to 0x08 and the Tx_NUM is set to 0x0, the bit Tx0[7:0] will be transmitted in next transfer. If Tx_BIT_LEN is set to 0x00 and Tx_NUM is set to 0x1, the core will perform two 32-bit transmit/receive successive using the same setting (the order is Tx0[31:0], Tx1[31:0]).

SPI Variable Clock Pattern Flag Register (SPI_VARCLK)

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

Bits	Description	
[31:0]	**VARCLK**	**Variable Clock Pattern** The value in this field is the frequency pattern of the SPI clock. If the bit field of VARCLK is '0', the output frequency of SCLK is given by the value of DIVIDER. If the bit field of VARCLK is '1', the output frequency of SCLK is given by the value of DIVIDER2. Refer to register DIVIDER. Refer to Variable Serial Clock Frequency paragraph for detailed description. Note: Used for CLKP = 0 only, 16 bit transmission.

DMA Control Register (SPI_PDMACTL)

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

7	6	5	4	3	2	1	0
Reserved						RXPDMAEN	TXPDMAEN

Bits	Description	
[1]	**RXPDMAEN**	**Receive DMA Start** Set this bit to 1 will start the receive DMA process. SPI module will issue request to DMA module automatically.
[0]	**TXPDMAEN**	**Transmit DMA Start** Set this bit to 1 will start the transmit DMA process. SPI module will issue request to DMA module automatically. If using DMA mode to transfer data, remember not to set GO_BUSY bit of SPI_CNTRL register. The DMA controller inside SPI module will set it automatically whenever necessary.

5.10 Timer Controller

5.10.1 General Timer Controller

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

Besides the general timer function, Timer1 and Timer2 provide another function. Timer1 can be used to generate IR carrier output. Timer2 provide touch capture function by detect external capture pins.

A simple fixed frequency timer TimerF is provided. Its frequency can be one of the following:

32kHz crystal / 32,
32kHz crystal / (4x32),
49MHz RC oscillator / 65536,
49MHz RC oscillator / (4x65536).

5.10.2 Features

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

- AMBA APB interface compatible
- Each channel with an 8-bit pre-scale counter, a 16-bit up-counter and an interrupt request signal.
- Independent clock source for each channel.
- Time out period = (Period of timer clock input) * (Prescale + 1) * (TCMP)
- Maximum counting cycle time = (1/49.152M) * (2^8) * (2^16), if TMRx_CLK=49.152MHz.
- Internal 16-bit up-counter is readable on the fly.

5.10.3 Timer Controller Block Diagram

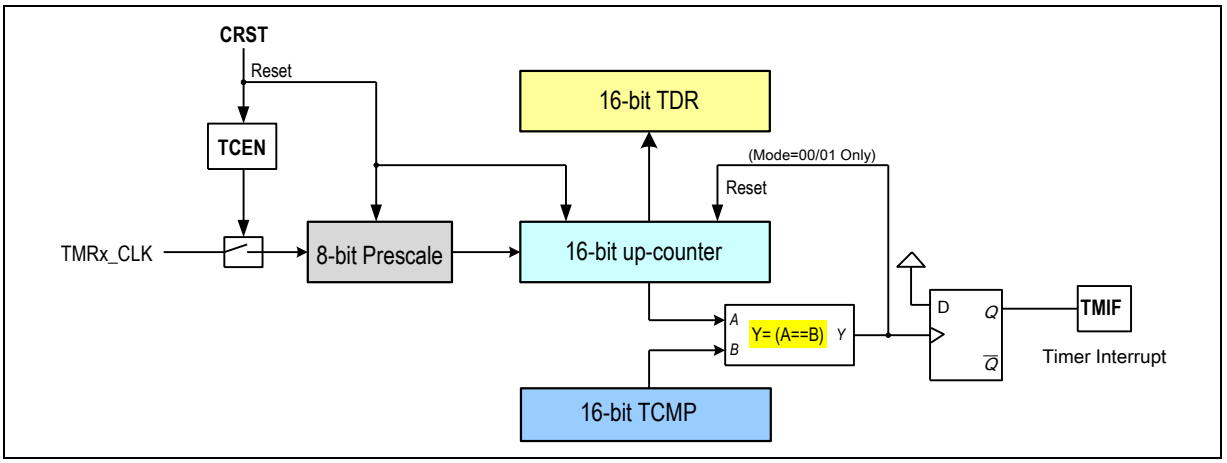
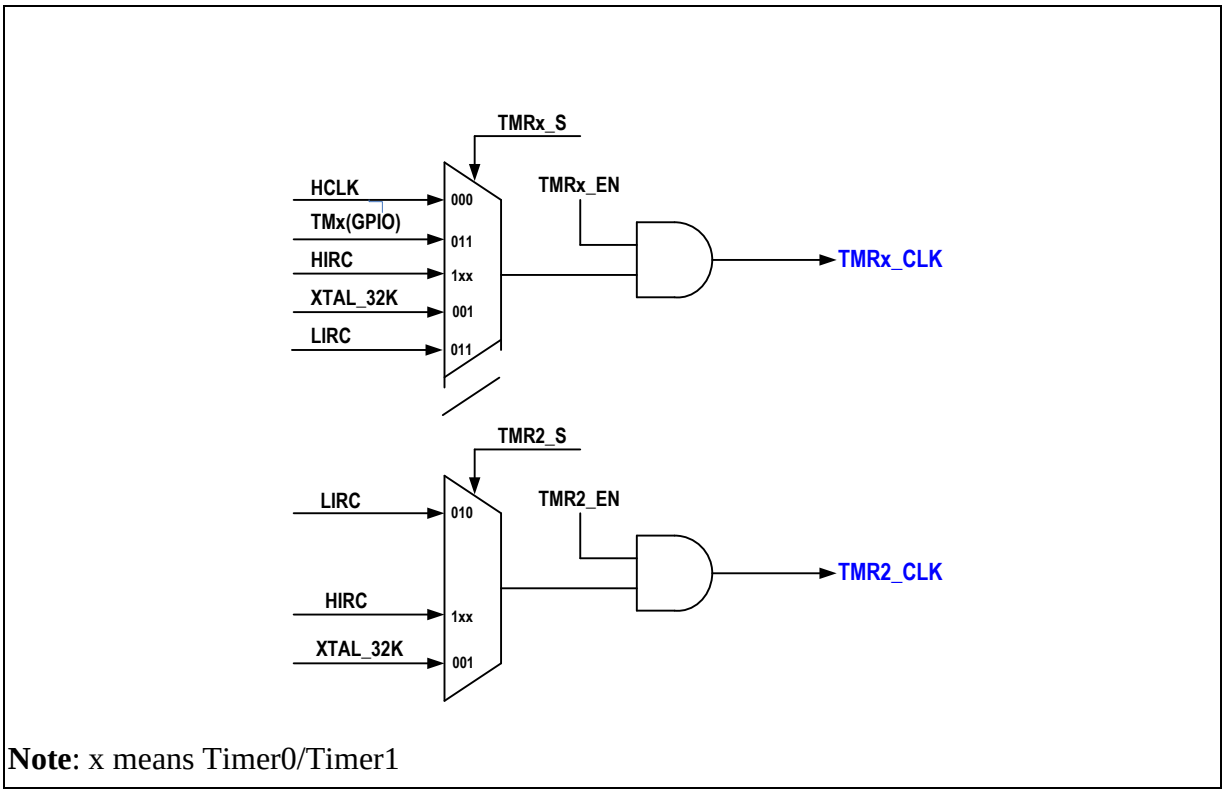


Figure 5-25 Timer0/1/2 Block Diagram



Note: x means Timer0/Timer1

Figure 5-26 Clock Source of Timer0/1/2

5.10.4 Timer Controller Register Map

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

Register	Offset	R/W	Description	Reset Value
TMR Base Address: TMR_BA=0x4001_0000				
TIMER0_CTL	TMR_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TIMER0_CMP	TMR_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TIMER0_INTSTS	TMR_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER0_CNT	TMR_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TIMER1_CTL	TMR_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TIMER1_CMP	TMR_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000
TIMER1_INTSTS	TMR_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER1_CNT	TMR_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TIMERF_INTSTS	TMR_BA+0x30	R/W	TimerF Interrupt Status Register	0x0000_0000
IR_CTL	TMR_BA+0x34	R/W	IR Carrier Output Control Register	0x0000_0000
TIMER2_CTL	TMR_BA+0x40	R/W	Timer2 Control and Status Register	0x0000_0005
TIMER2_CMP	TMR_BA+0x44	R/W	Timer2 Compare Register	0x0000_0000
TIMER2_INTSTS	TMR_BA+0x48	R/W	Timer2 Interrupt Status Register	0x0000_0000
TIMER2_CNT	TMR_BA+0x4C	R	Timer2 Data Register	0x0000_0000

5.10.5 Timer Controller Register Description

Timer Control Register (TIMERx_CTL)

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

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

Bits	Description	
[31]	**Reserved.**	Reserved.
[30]	**CNTEN**	**Counter Enable Bit** 0 = Stop/Suspend counting. 1 = Start counting. **Note:** This bit is auto-cleared by hardware in one-shot mode (OPMODE = 00b) when the associated timer interrupt is generated (INTEN = 1).
[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 associated count is equal to TIMERx_CMP.

[28:27]	**OPMODE**	**Timer Operating Mode** 00 = The Timer is operating in the one-shot mode. The associated interrupt signal is generated once (if INTEN is 1) and CNTEN is automatically cleared by hardware. 01 = The Timer is operating in the periodic mode. The associated interrupt signal is generated periodically (if INTEN is 1). 10 = Reserved. 11 = The Timer is operating in continuous counting mode. The associated interrupt signal is generated when TIMERx_CNT = TIMERx_CMP (if INTEN is 1); however, the 16-bit up-counter counts continuously without reset.
[26]	**RSTCNT**	**Counter Reset Bit** Set this bit will reset the Timer counter, pre-scale and also force CNTEN to 0. 0 = No effect. 1 = Reset Timer's pre-scale counter, internal 16-bit up-counter and CNTEN bit.
[25]	**ACTSTS**	**Timer Active Status Bit (Read Only)** This bit indicates the counter status of Timer. 0 = Timer is not active. 1 = Timer is active.
[24:8]	**Reserved**	Reserved.
[7:0]	**PSC**	**Timer Clock Prescaler** Clock input is divided by (PSC+1) before it is fed to the counter. If PSC = 0, then there is no scaling.

Timer Compare Register (TIMERx_CMP)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPDAT**	**Timer Comparison Value** CMPDAT is a 16-bit comparison register. When the 16-bit up-counter is enabled and its value is equal to CMPDAT value, a Timer Interrupt is requested if the timer interrupt is enabled with TIMERx_CTL.INTEN = 1. **Note 1:** Never set CMPDAT to 0x000 or 0x001. Timer will not function correctly. **Note 2:** No matter CNTEN is 0 or 1, whenever software writes a new value into this register, TIMER will restart counting by using this new value and abort previous count.

Timer Interrupt Status Register (TIMERx_INTSTS)

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

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**TIF**	**Timer Interrupt Flag (Read Only)** This bit indicates the interrupt status of Timer. TIF bit is set by hardware when the 16-bit counter matches the timer comparison value (CMPDAT). It is cleared by writing 1 to itself 0 = No effect. 1 = CNT (TIMERx_CNT[15:0]) value matches the CMPDAT (TIMERx_CMP[15:0]) value.

Timer Data Register (TIMERx_CNT)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNT**	**Timer Data Register** User can read this register for the current up-counter value while TIMERx_CTL.CNTEN is set to 1,

TimerF Interrupt Status Register (TIMERF_INTSTS)

Register	Offset	R/W	Description	Reset Value
TIMERF_INTSTS	TMR_BA+0x30	R/W	TimerF 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						**TFIE**	**TFIF**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TFIE**	**TimerF Interrupt Enable** 0 = Disable TimerF Interrupt. 1 = Enable TimerF Interrupt.
[0]	**TFIF**	**TimerF Interrupt Flag** This bit indicates the interrupt status of TimerF. TFIF bit is set by hardware when TimerF time out. It is cleared by writing 1 to this bit. 0 = It indicates that TimerF does not time out yet. 1 = It indicates that TimerF time out. The interrupt flag is set if TimerF interrupt was enabled.

IR Carrier Output Control Register (IR_CTL)

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

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

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

Bits	Descriptions	
[31:2]	**Reserved**	Reserved
[1]	**IRCEN**	**IR carrier output enable** 0 = Disable IR carrier output, 1 = Enable IR carrier output. Timer1 time out will toggle the output state on IROUT pin.
[0]	**NONCS**	**Non-carrier state** 0 = IROUT keeps low when IRCEN is 0, 1 = IROUT keeps high when IRCEN is 0.

5.11 Watchdog Timer

The purpose of Watchdog Timer (WDT) is to perform a system reset if system runs into an unknown state. This prevents system from hanging for an indefinite period of time. The watchdog timer includes a 19-bit free running counter with programmable time-out intervals.

Setting WDT_CTL[7](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 WDT_CTL[3](IF) will be set immediately to request a WDT interrupt if the watchdog timer interrupt enable bit WDT_CTL[6](INTEN) is set, in the meantime, a specified delay time follows the time-out event. User must set WDT_CTL[0](RSTCNT) (Watchdog timer reset) high to reset the 19-bit WDT counter to prevent Watchdog timer reset before the delay time expires. WDT_CTL[0](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 WDT_CTL[10:8](TOUTSEL). If the WDT counter has not been cleared after the specific delay time expires, the watchdog timer will set Watchdog Timer Reset Flag WDT_CTL[2](RSTF) high and reset CPU. This reset will last 64 WDT clocks then CPU restarts executing program from reset vector (0x0000 0000). WDT_CTL[2](RSTF) will not be cleared by Watchdog reset. User may poll WDT_CTL[2](RSTF) by software to recognize the reset source.

Table 5-4 Watchdog Timeout Interval Selection

TOUTSEL	WDT Interrupt Timeout	WDT Reset Timeout	WDT Reset Timeout Interval (WDT_CLK=24 MHz)	WDT Reset Timeout Interval (WDT_CLK=32 KHz)
000	2^4 WDT_CLK	(2^4 + 1024) WDT_CLK	43.33 μs	32.5 ms
001	2^6 WDT_CLK	(2^6 + 1024) WDT_CLK	45.33 μs	34 ms
010	2^8 WDT_CLK	(2^8 + 1024) WDT_CLK	53.33 μs	40 ms
011	2^{10} WDT_CLK	(2^{10} + 1024) WDT_CLK	85.33 μs	64 ms
100	2^{12} WDT_CLK	(2^{12} + 1024) WDT_CLK	213.33 μs	160 ms
101	2^{14} WDT_CLK	(2^{14} + 1024) WDT_CLK	0.72 ms	544 ms
110	2^{16} WDT_CLK	(2^{16} + 1024) WDT_CLK	2.78 ms	2080 ms
111	2^{18} WDT_CLK	(2^{18} + 1024) WDT_CLK	10.97 ms	8224 ms

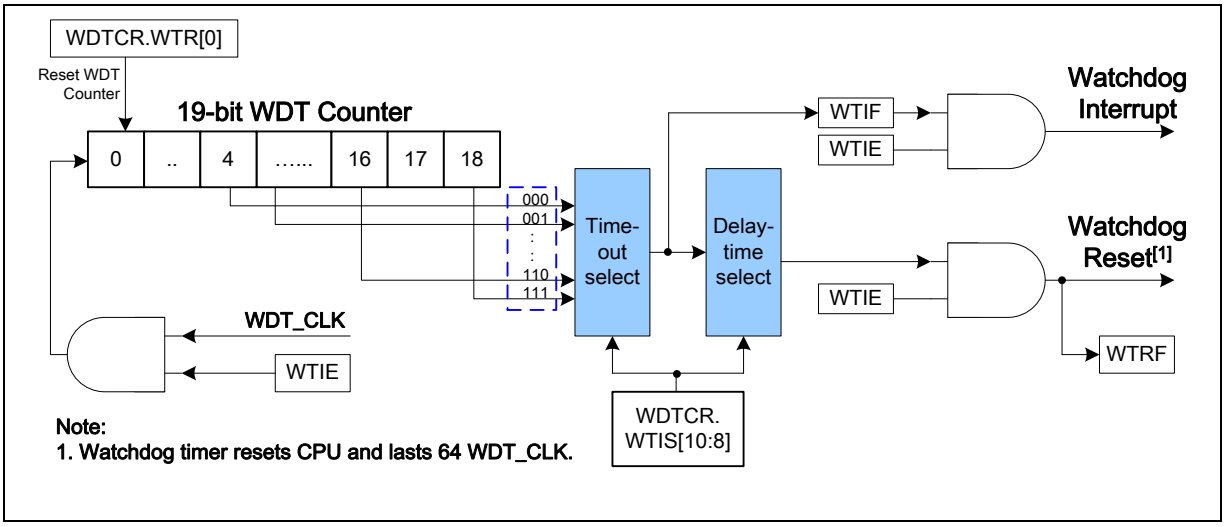


Figure 5-27 Watchdog Timer Block Diagram

5.11.1 Watchdog Timer Control Register Map

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

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

5.11.2 Watchdog Timer Control Register Description

Watchdog Timer Control Register (WDT_CTL)

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

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**TOUTSEL**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WDTEN	**INTEN**	**WKF**	**WKEN**	**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 Watchdog timer is not reset. The WDT interrupt timeout is given by: 000 = 2^4 * WDT_CLK. 001 = 2^6 * WDT_CLK. 010 = 2^8 * WDT_CLK. 011 = 2^{10} * WDT_CLK. 100 = 2^{12} * WDT_CLK. 101 = 2^{14} * WDT_CLK. 110 = 2^{16} * WDT_CLK. 111 = 2^{18} * WDT_CLK. WDT reset timeout = (WDT interrupt timeout +1024) * WDT_CLK. Where WDT_CLK is the period of the Watchdog Timer clock source.
[7]	**WDTEN**	**Watchdog Timer Enable** 0 = Disable the WDT(Watchdog timer) (This action will reset the internal counter). 1 = Enable the WDT(Watchdog timer).
[6]	**INTEN**	**Watchdog Time-Out Interrupt Enable** 0 = Disable the WDT time-out interrupt. 1 = Enable the WDT time-out interrupt.
[5]	**WKF**	**WDT Time-Out Wake-Up Flag** If WDT causes CPU wake up from sleep or power-down mode, this bit will be set to high. 0 = WDT does not cause CPU wake-up. 1 = CPU wakes up from sleep or power-down mode by WDT time-out interrupt. **Note:** This bit is cleared by writing 1 to it.
[4]	**WKEN**	**WDT Time-Out Wake-Up Function Control** If this bit is set to 1, while WDT time-out interrupt flag IF (WDT_CTL[3]) is generated to 1 and interrupt enable bit INTEN (WDT_CTL[6]) is enabled, the WDT time-out interrupt signal will generate a wake-up trigger event to chip. 0 = Enable the Wakeup function that WDT timeout can wake up CPU from power-down mode. 1 = Disable WDT Wakeup CPU function.

[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. **Note:** This function cannot work with XTL32-based clock source.
[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 cycles

5.12 Audio Class D Speaker Driver (DPWM)

5.12.1 Functional Description

The I91032 includes a differential Class D (PWM) speaker driver capable of delivering (TBD)W into an 8Ω load at 4.5V supply voltage. The driver works by up-sampling and modulating a PCM input to differentially drive the SPK+ and SPK- pins. The speaker driver operates from its own independent supply VCCSPK and VSSSPK. This supply should be well decoupled as peak currents from speaker driver are large.

5.12.2 Features

- Differential Bridge-Tied-Load structure to directly drive 8Ω Speaker.
- Power delivery up to 4.5W @5V into 8Ω.
- Configurable input sample rate.
- 16 Sample FIFO for audio output.
- PDMA data channel for streaming of PCM audio data.

5.12.3 Block Diagram

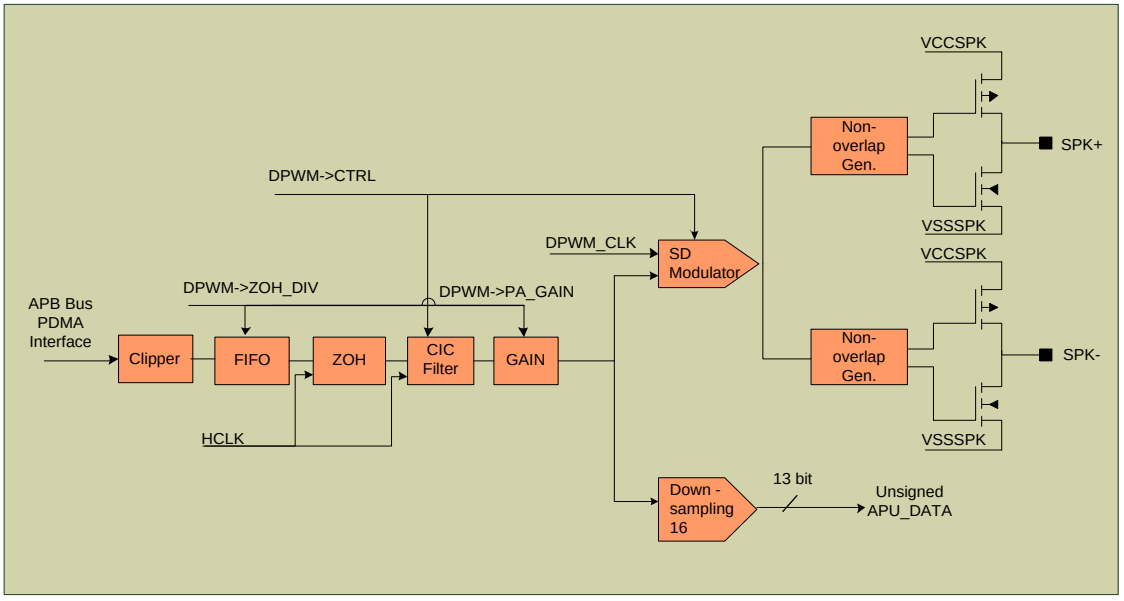


Figure 5-28 DPWM Block Diagram

5.12.4 Operation

The DPWM block receives audio data by writing 16bit PCM audio to the FIFO. FIFO is accessed through PDMA for ease of streaming. The audio stream is sampled by a zero-order hold and fed to an up-sampling Cascaded Integrator Comb (CIC) filter with an up-sampling ratio of 64. The signal is then modulated and sent to the driver stage through a non-overlap circuit. Master clock rate of the Delta-Sigma modulator is controlled by DPWM_CLK. This

clock is generated by the internal oscillator (OSC48M) and operates at the frequency of OSC48M. Ultimate SNR (Signal-to-Noise Ratio) is determined by the time resolution of the master clock.

5.12.4.1 Determining DPWM Sample Rate

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

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

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

Table 5-5 DPWM Sample Rates for Various HCLK

HCLK (MHz)	ZOH_DIV	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

5.12.4.2 Configuring Speaker Driver

To operate the speaker driver the following configuration is recommended:

- Enable DPWM clock source (APBCLK.DPWM_EN).
- Reset DPWM IP block. (IPRSTC1.DPWMRST)
- Select sample rate based on current HCLK frequency.
- Setup PDMA channel to provide data to DPWM.
- Enable PDMA Request.
- Enable Driver.

5.12.4.3 Peripheral DMA Request for DPWM and DAC

Normal use of the DPWM and DAC are with PDMA. In this mode DPWM and DAC request PDMA service whenever there is space in FIFO. PDMA channel will copy data from a streaming buffer to the DPWM/DAC and alert the CPU when buffer is empty. In this way an entire buffer of data can be sent to DPWM /DAC without any CPU intervention.

5.12.4.4 Determining DAC Sample Rate

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

$$F_s = HCLK \div ZOH\_DIV \div (DAC\_DIV + 1)$$

Where HCLK is the master CPU clock rate and ZOH_DIV and DAC_DIV are the divider control register

5.12.5 DPWM and DAC 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 and DAC Control Register	0x0000_0000
DPWM_STS	DPWM_BA+0x04	R	DPWM and DAC FIFO Status Register	0x0000_0002
DPWM_DMACTL	DPWM_BA+0x08	R/W	DPWM and DAC PDMA Control Register	0x0000_0000
DPWM_DATA	DPWM_BA+0x0C	W	DPWM and DAC FIFO Input Register	0x0000_0000
DPWM_ZOHDIV	DPWM_BA+0x10	R/W	DPWM and DAC Zero Order Hold Division Register	0x0000_FF30

5.12.6 APU Control Register Description

DPWM and DAC Control Register (DPWM_CTL)

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

23	22	21	20	19	18	17	16
Reserved		DAC_INSEL	DACBUF_PD	DACBUFBYPASS	DAC_PD	DAC_EN_10BIT	DAC_EN
15	14	13	12	11	10	9	8
Reserved	ZCIE	Reserved	RXTH[3:0]				RXTHIE
7	6	5	4	3	2	1	0
CLIPIE	DPWM_EN	DITHEREN		DEADTIME	MODUFRQ		

Bits	Description	
[21]	**DAC_INSEL**	**DAC input data selection** 0 --- DATA from CIC and GAIN output 1 --- DATA from FIFO output
[20]	**DACBUF_PD**	**DAC BUFFER POWER DOWN** 0 --- power on DAC BUFFER 1 --- power off DAC BUFFER
[19]	**DACBUFBYPASS**	**DAC BUFFER BYPASS** 0 --- disable BYPASS DAC BUFFER 1 --- enable BYPASS DAC BUFFER
[18]	**DAC_PD**	**DAC POWER DOWN** 0 --- power on DAC13B 1 --- power down DAC13B
[17]	**DAC_EN_10BIT**	**DAC ENABLE 10Bit** 0 --- 13 Bit mode 1 --- 10 bit mode, don't use lower 3BITs
[16]	**DAC_EN**	**DAC ENABLE** 0 --- DAC function disable 1 --- DAC function enable

[15]	**Reserved**	**Reserved**
[14]	**ZCIE**	**Zero cross enable** 0: output data doesn't cross zero point 1: output data cross zero point
[13]	**Reserved**	Reserved
[12:9]	**RXTH[3:0]**	**DPWM FIFO threshold** If the valid data count of the DPWM FIFO buffer is less than or equal to RXTH setting, the RXTHF bit will set to 1, else the RXTHF bit will be cleared to 0.
[8]	**RXTHIE**	**DPWM FIFO threshold interrupt** 1: DPWM FIFO threshold interrupt Enabled. 0: DPWM FIFO threshold interrupt Disabled.
[7]	**CLIPIE**	**Read data in DATA[31:0] clipped within 0x0000-7fff ~ 0xffff-8000** Read operation of this register will get clipped data in DATA[31:0] register but clipped with the range : 0x00 ~ 7fff ~ 0x ffff ~8000. The content of DATA[31:0] will not be change.
[6]	**DPWMEN**	**DPWM Enable.** 1: Enable DPWM, SPK pins are enabled and driven, data is taken from FIFO. 0: Disable DPWM, SPK pins are tri-state, CIC filter is reset, FIFO pointers are reset (FIFO data is not reset). **Note** : This field will be effective only when DAC_EN field in this register is set as "0".
[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. **Freq** / DPWM_CLK Division / Frequency for DPWM_CLK = 49.152MHz **0**: 228 / 431,158 **1**: 156 / 630,154 **2**: 76 / 1,293,474 **3**: 52 / 1,890,462 **4**: 780 / 126,031 **5**: 524 / 187,603 **6**: 396 / 248,242 **7**: 268 / 366,806

DPWM and DAC FIFO Status Register (DPWM_STS)

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

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

Bits	Description	
[31:3]	**-**	**Reservd**
[6:3]	**FIFO_POINTER**	**DPWM FIFO Pointer (Read Only)** The FULL bit and FIFO_POINTER[3:0] indicates the field that the valid data count within the DPWM FIFO buffer. The Maximum value shown in FIFO_POINTER is 15. When the using level of DPWM FIFO Buffer equal to 16, The FULL bit is set to 1.

[2]	**RXTHF**	**DPWM FIFO threshold Interrupt Status (Read Only)** 0 = The valid data count within the DPWM FIFO buffer is larger than the setting value of RXTH. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of RXTH.
[1]	**EMPTY**	FIFO Empty 1= FIFO is empty 0= FIFO is not empty
[0]	**FULL**	**FIFO Full** 1 = FIFO is full. 0 = FIFO is not full.

DPWM and DAC PDMA Control Register (DPWM_DMACTL)

Register	Offset	R/W	Description	Reset Value
DPWM_DMACTL	DPWM_BA+0x08	R/W	DPWM and DAC PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved
[0]	**DMAEN**	**Enable DPWM and DAC DMA Interface.** 1= Enable PDMA. Block will request data from PDMA controller whenever FIFO is not empty. 0= Disable PDMA. No requests will be made to PDMA controller.

DPWM and DAC FIFO Input (DPWM_DATA)

Register	Offset	R/W	Description	Reset Value
DPWM_DATA	DPWM_BA+0x0C	W	DPWM and DAC FIFO Input 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**
INDATA[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INDATA[7:0]							

Bits	Description	
[31:16]	**Reserved**	**Reserved**
[15:0]	**INDATA**	**DPWM and DAC FIFO Audio Data Input** A write to this register pushes data onto the DPWM and DAC FIFO and increments the write pointer. This is the address that PDMA writes audio data to.

DPWM and DAC Zero Order Hold Division Register (DPWM_ZOHDIV)

Register	Offset	R/W	Description	Reset Value
DPWM_ZOHDIV	DPWM_BA+0x10	R/W	DPWM and DAC Zero Order Hold Division Register	0x0000_FF30

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

Bits	Description	
[21:16]	**DAC_DIV**	**DAC clock divider** DAC data sample rate is set by this divider Fs = HCLK / ZOH_DIV / (DAC_DIV+1) Note : by DAC_DIV == 0x0, Sample rate is decided by ZOH_DIV
[15:8]	**GAIN**	**(GAIN[7:0]+1)/256, GAIN≠0**
[7:0]	**ZOH_DIV**	**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 / ZOH_DIV / 64 Valid range is 1,..,255. Default is 48, which gives a sample rate of 16kHz for a 49.152MHz (default) HCLK.

5.13 13-bit DAC

5.13.1 Functional Description

The 13 bit DAC output is an alternative to the DPWM output. The resolution of the DAC can be reduced to 10 bit.

The maximum sample rate of the DAC is 5MHz.

The DAC has an output buffer which is capable of driving resistive loads with a minimum impedance of 5kOhm rail to rail. This output buffer can be bypassed if necessary. The low pass -3dB corner frequency of this buffer is above 100kHz.

Figure 5-31 shows the block diagram of the DAC and Output Buffer.

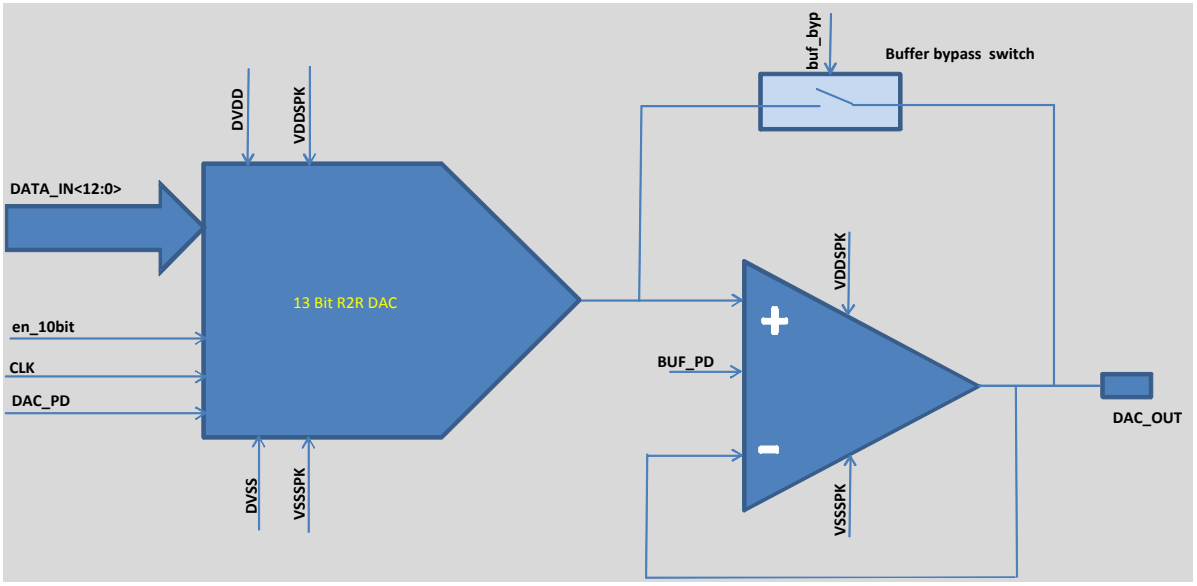


Figure 5-29 Block Diagram of DAC and Output Buffer.

5.14 10-bit Analog-to-Digital Converter (ADC) and Pre-amplifier

I91032 contains one 10-bit successive approximation analog-to-digital converters (ADC), with 16 times oversampling and H/W decimation filter, SNR can achieve close to 12 bits resolution. ADC can be programmed operation independently. ADC has 8-channel inputs. (Ch-4 is differential input from AIN4/AIN5 followed by programmable gain control amplifer, Ch-5 is reserved) The A/D converter supports three operation modes: single, single-cycle scan and continuous scan mode. Each of A/D converters can be started by software.

Note: The PCLK is equivalent to HCLK thereafter.

5.14.1 Features

- There are max 8 time sharing ADC channel input. The ch0-ch7 is related with external input. Ch-4 is differential input from AIN4/AIN5 followed by programmable gain control amplifer,ch-5 is reserved . For other channel input show in 5.14.3.
- Analog input voltage range: 0~5.5V
- SNR close to 12-bits resolution (based on 16 times over_sampling, and only for channel-4, the differential input from AIN4/AIN5) and 10-bits accuracy is guaranteed
- For every AD Conversion take 12 ADC clock minimum. Other clock more than 12 is idle clock.
- Maximum ADC clock frequency is 2.4MHz, minimum is 12.5kHz
- Up to 200K (i.e. 2.4MHz/12 = 200KHz) SPS conversion rate
- Three operating modes
  - Single mode: Single channel A/D conversion
  - Single-cycle scan mode: Conversion on all enabled channels once
  - Continuous scan mode: Repetitive conversion on enabled channels
- An A/D conversion can be started by
  - Software write SWTRG bit
- Conversion results are held in data registers for assign channel with valid and overrun indicators
- Conversion result can be compared with specified value and user can select whether to generate an interrupt when conversion result is less than/great or equal to the compare register setting.

5.14.2 ADC Block Diagram

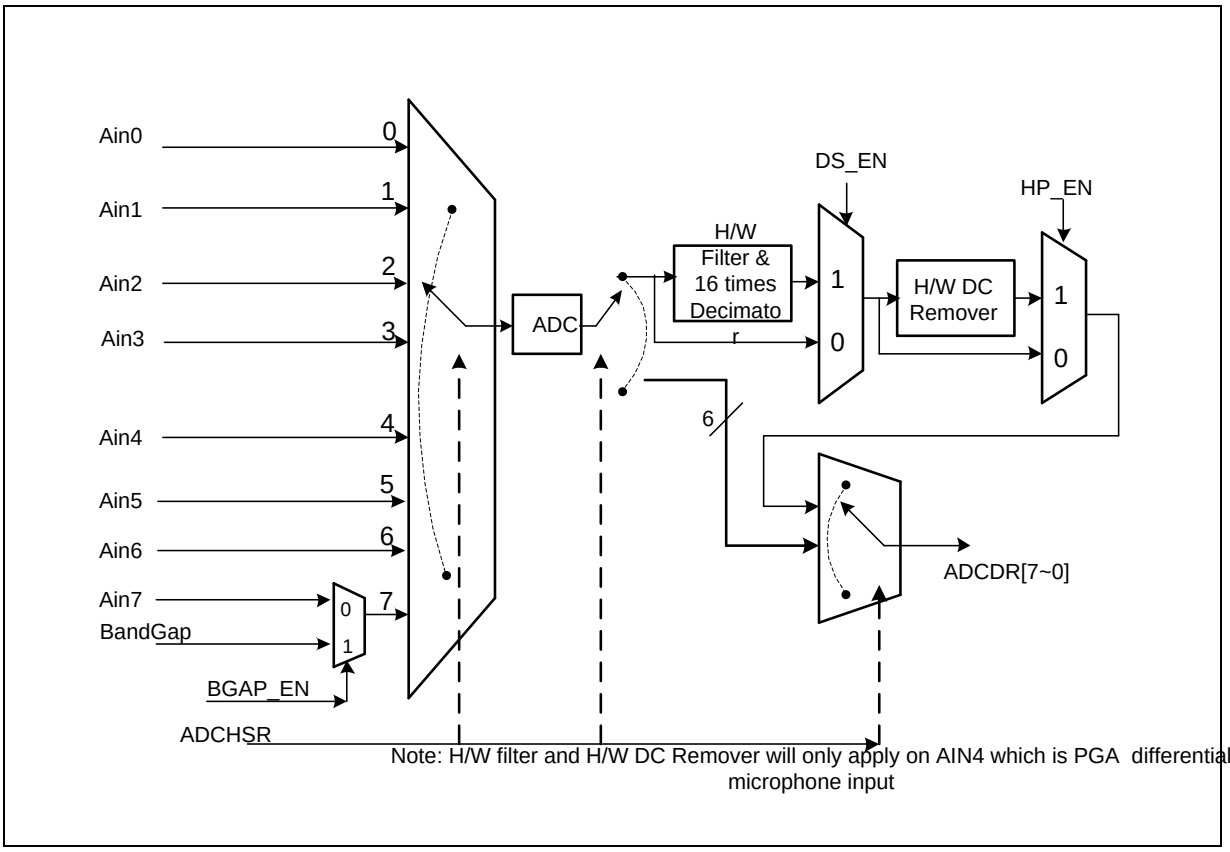


Figure 5-30 ADC Controller Block Diagram

5.14.3 ADC Inputs

SAR INPUT	Signal
Ain0	Reserved
Ain1	GPA1
Ain2	GPA2
Ain3	GPA3
Ain4	MICPGA
Ain5	MICN or GPA14
Ain6	Reserved
Ain7_0	MICBIAS
Ain7_1	Reserved

5.14.4 ADC H/W Filter and Decimator

To improve 10 bits ADC SNR further, signal can be oversampling 2/4/8/16 times by programming registers related with ADC and filter/decimated to desired sampling rate by H/W.

For example, if desired sampling rate in application is Fs, however user intends to want to have better 12 dB SNR improvement than desired sampling rate Fs. To achieve the above goal, programmer can set ADC sampling rate as 16xFs. This is because there is close to 6dB improvement for every 4 times oversampling of ADC. The first step is to enable DS_EN (setting this bit will also force ADC operating in continuous scan mode). Then setting DS_1CH to specify one or two channel to do ADC conversion, if DS_1CH setting as "0" (two channel), both channel will be in same sampling rate and same decimation rate. Finally, to set decimation (down sample) rate by setting DS_RATE as"11" can have 16 times decimation. After completing the above settings, the application program can get desired sampling rate Fs while the ADC operates at 16xFs.

Another optional DC remover H/W can be used to filter out DC component from signal.

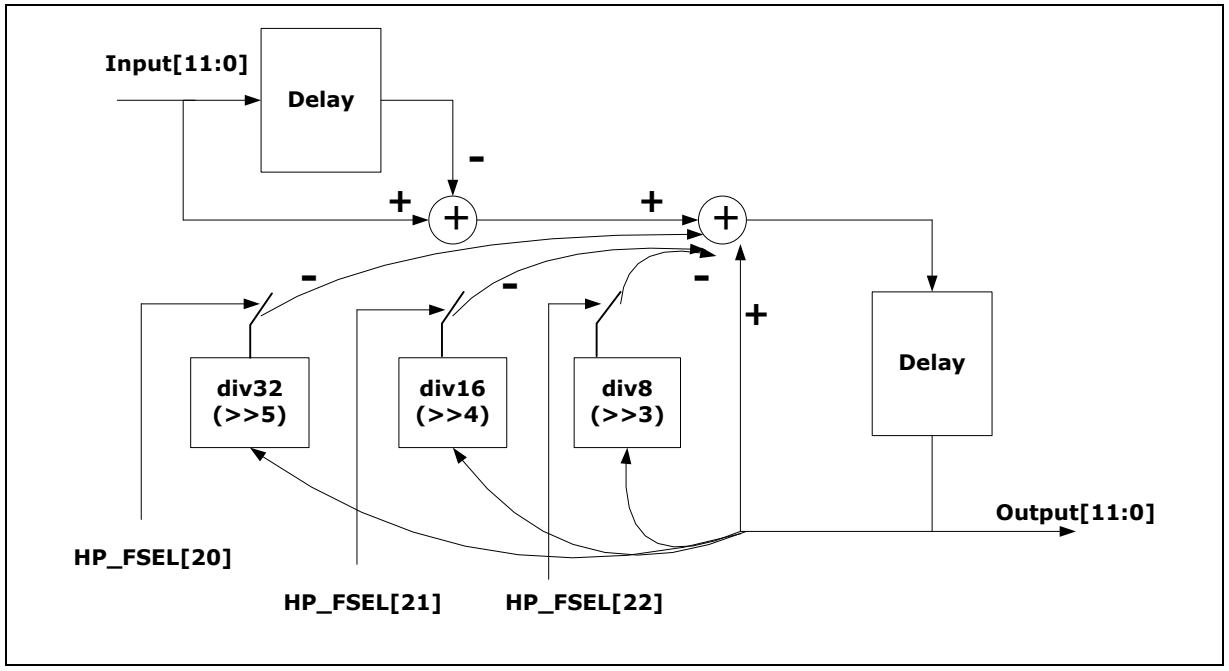


Figure 5-31 H/W DC Remover Architecture

5.14.5 ADC Operation Procedure

The A/D converter operates by successive approximation with 12-bit resolution. The ADCs have three operation modes: single mode, single-cycle scan mode and continuous mode and.

When changing the operating mode or analog input channel enable, in order to prevent incorrect operation, software must clear SWTRG bit to 0 in ADC_CTL register. The A/D converter discards current conversion immediately and enters idle state while SWTRG bit is cleared

5.14.5.1 ADC Clock Generator

The maximum sampling rate is up to 500 kHz (TBD) and the conversion time is less 2μS (TBD). It needs at least 12 ADC clocks to complete an A/D conversion. The ADC clock source can be from HCLK or HIRC oscillator. The selected clock source is divided by (ADCDIV+1) to produce the clock to A/D converter which should be less than or equal to 6MHz (500k Hz * 12) and higher than 12.5kHz.

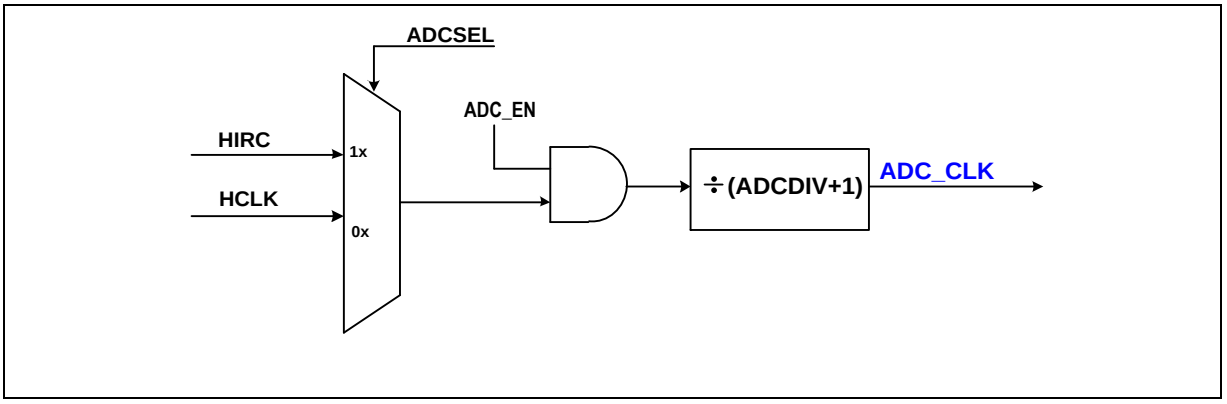


Figure 5-32 ADC Clock Source

5.14.5.2 Single Mode

In single mode, A/D conversion is to be performed only once on the specified single channel. The operations are as follows:

1. The channel defined in CHSEQ0 is the channel to be converted.
2. A/D conversion is started when the SWTRG bit in ADC_CTL is set to 1 by software.
3. When A/D conversion is finished, the result is transferred to the A/D data register ADC_DATn corresponding to the channel.
4. On completion of conversion, the ADIF bit in ADC_STATUS is set to 1. If the ADCIE bit is 1, an ADINT interrupt request is generated.
5. The SWTRG bit remains “1” during A/D conversion. When A/D conversion ends, the SWTRG bit is automatically cleared to 0 and the A/D converter enters the idle state.
6. When the SWTRG bit is cleared to 0 during A/D conversion, A/D conversion stops and the A/D converter enters the idle state.

5.14.5.3 Continuous Scan Mode

In continuous scan mode, and DS_EN field of register ADC_CTL is "0", A/D conversion is to be performed sequentially on the specified channels that are defined by CHSEQx bits in ADC_CHSEQ register (8 channels maximum). The operations are as follows:

1. When the SWTRG bit in ADC_CTL is set to 1 by software, A/D conversion starts on the channel selected by CHSEQ0.
2. When A/D conversion for each channel selected by channel sequence register (ADC_CHSEQ) is completed, the converted result is sequentially transferred to the A/D data register corresponding to channel sequence.
3. When the conversion for all the selected channels is completed, the ADIF bit in ADC_STATUS is set to 1. If the ADCIE bit is 1, an ADINT interrupt is requested after A/D conversion ends. Conversion of the channel defined in CHSEQ0 starts again.
4. Steps 2 to 3 are repeated as long as the SWTRG bit remains to 1. When the SWTRG bit is cleared to 0, A/D conversion stops and the A/D converter enters the idle state.

An example timing diagram for continuous scan mode on 3 channels (CHSEQ0=4, CHSEQ1=3, CHSEQ2=4, CHSEQ3=2 and CHSEQ4=F) is shown as below:

(This example is only appropriate for ADC)

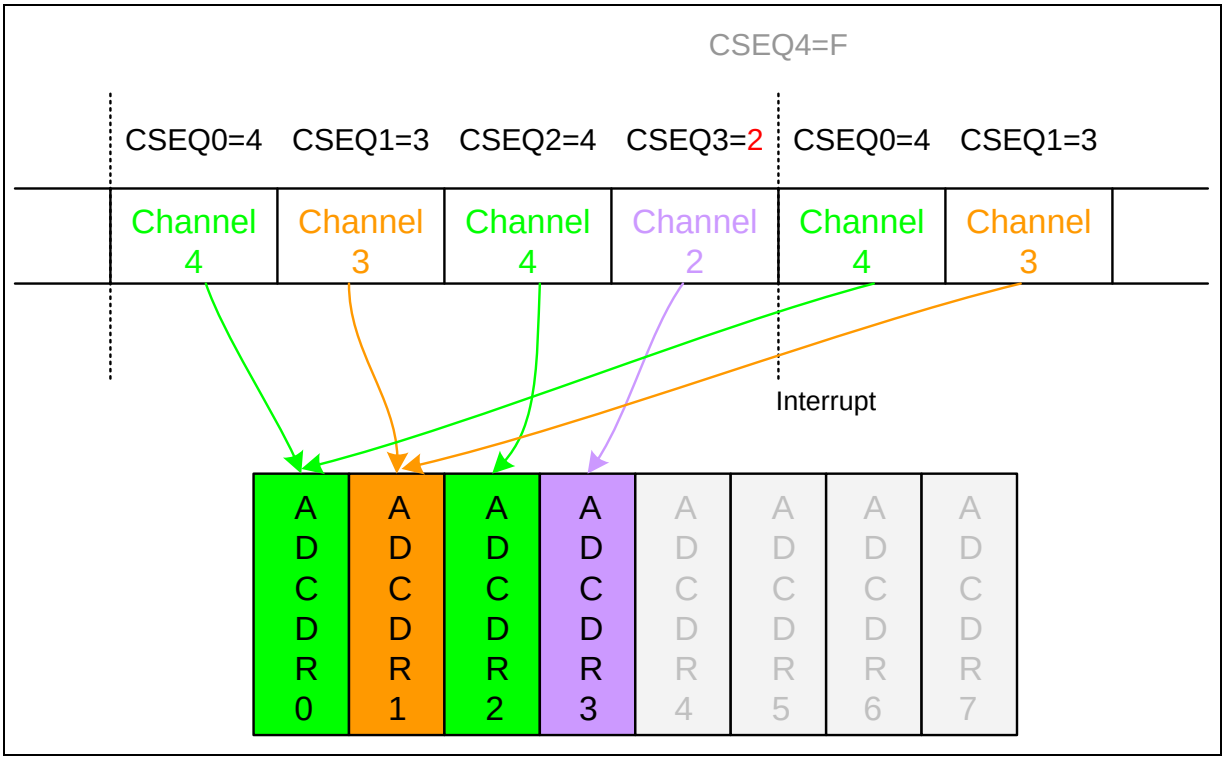


Figure 5-33 Continuous Scan on Selected Channels and **DS_EN** == "0"

When **DS_EN** field in **ADC_CTL** register was set as "1", the ADC conversion will be forces as continuous scan mode. When **DS_1CH** field of **ADC_CTL** register is "1", A/D conversion is to be performed repeatedly on the channels defined by CHSEQ0 bits in ADC_CHSEQ

register (8 channels maximum). When **DS_1CH** field of **ADC_CTL** register is "0", A/D conversion is to be performed toggled and repeatedly on the channels defined by **CHSEQ0** bits and **CHSEQ1** in ADC_CHSEQ register (8 channels maximum).

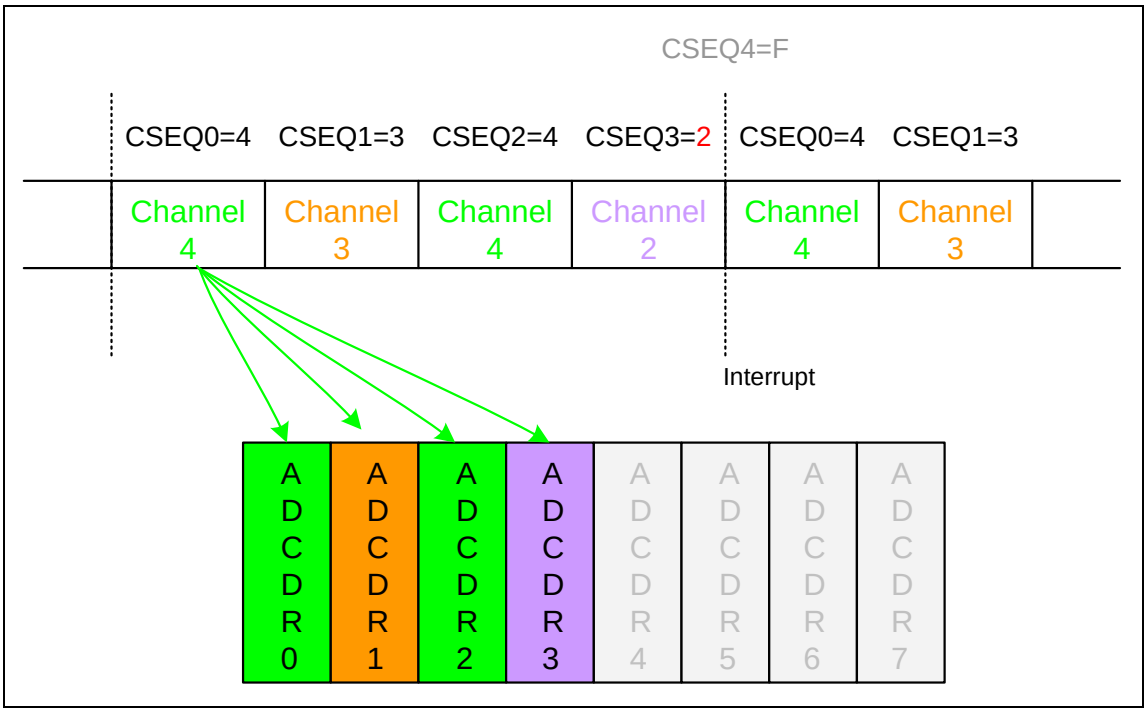


Figure 5-34 Continuous Scan on Selected Channels
DS_EN == "1", **DS_1CH** == "1"

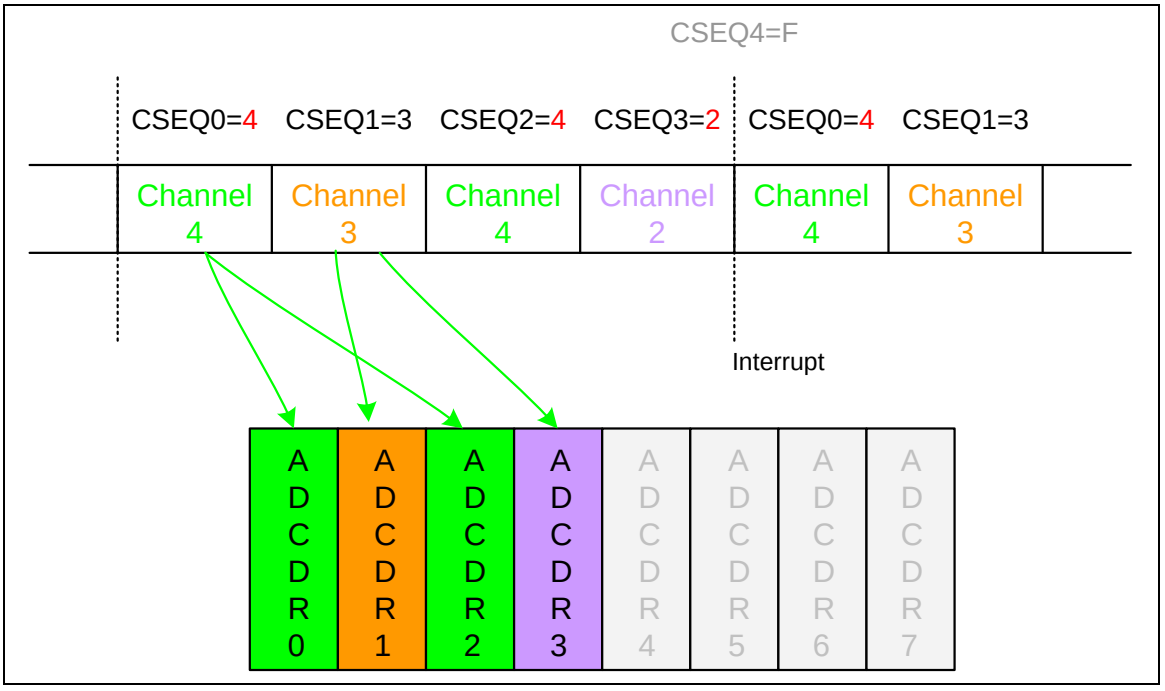


Figure 5-35 Continuous Scan on Selected Channels
DS_EN == "1", **DS_1CH** == "0"

5.14.5.4 Single-Cycle Scan Mode

In single-cycle scan mode, A/D conversion is to be performed once on the specified channels that are defined by CHSEQx bits in ADC_CHSEQ register (8 channels maximum for ADC). Operations are as follows:

1. When the SWTRG bit in ADC_CTL is set to 1 by a software, A/D conversion starts on the channel selected by CHSEQ0.
2. When A/D conversion for channel selected by channel sequence registers (CHSEQx) is completed, the result is sequentially transferred to the A/D data register corresponding to channel sequence.
3. When the conversion for all the selected channels is completed, the ADIF bit in ADC_STATUS is set to 1. If the ADCIE bit is 1, an ADINT interrupt is requested after A/D conversion ends.
4. After A/D conversion ends, the SWTRG bit is automatically cleared to 0 and the A/D converter enters the idle state. When the SWTRG bit is cleared to 0 during A/D conversion, A/D conversion stops and the A/D converter enters the idle state.

An example timing diagram for single-cycle scan on 4 channels (CHSEQ0=2, CHSEQ1=4, CHSEQ2=3, CHSEQ3=4 and CHSEQ4=F) is shown as below:

(This example is only appropriate for ADC)

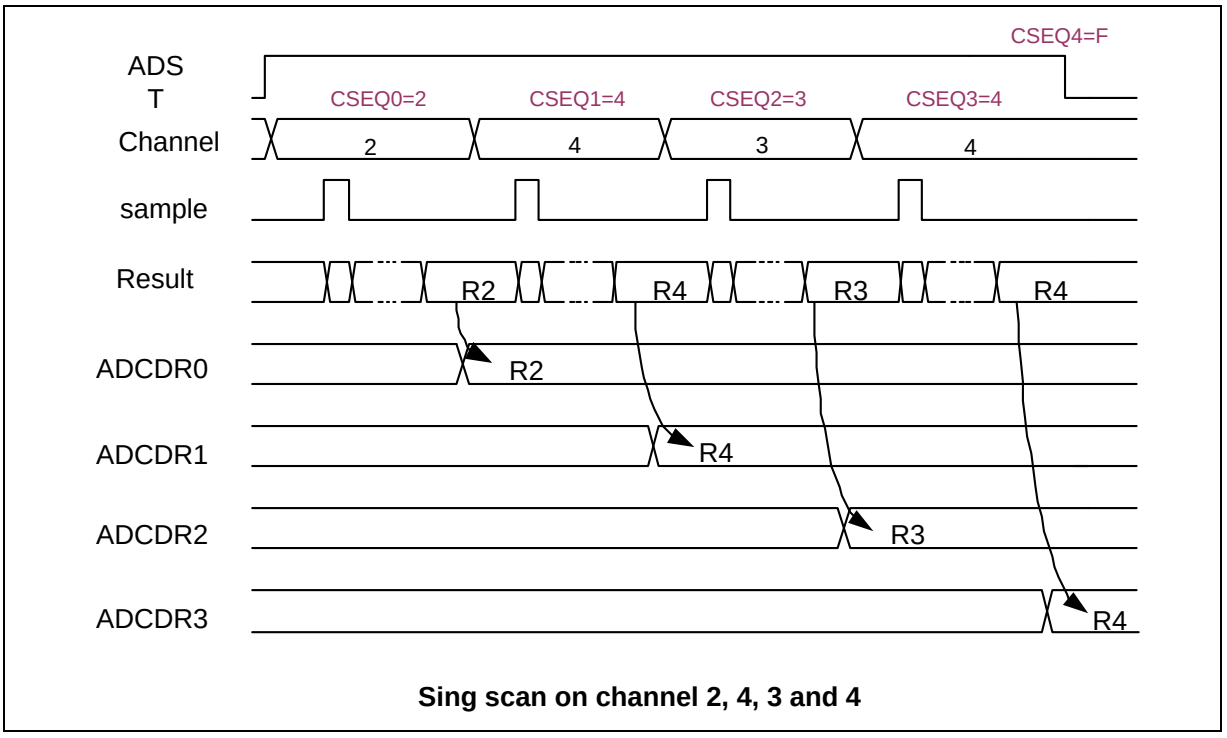


Figure 5-36 Single-Cycle Scan on selected Channels

5.14.5.5 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit for each module, the A/D samples the analog input when A/D conversion start delay time (Td) has passed after SWTRG bit in ADC_CTL is set to 1, then start conversion. Due to ADC clock is generated by PCLK divided by (N+1), the maximum delay time from APB write to A/D start sampling analog input time is 2N PCLKs. The start delay time is shown as below:

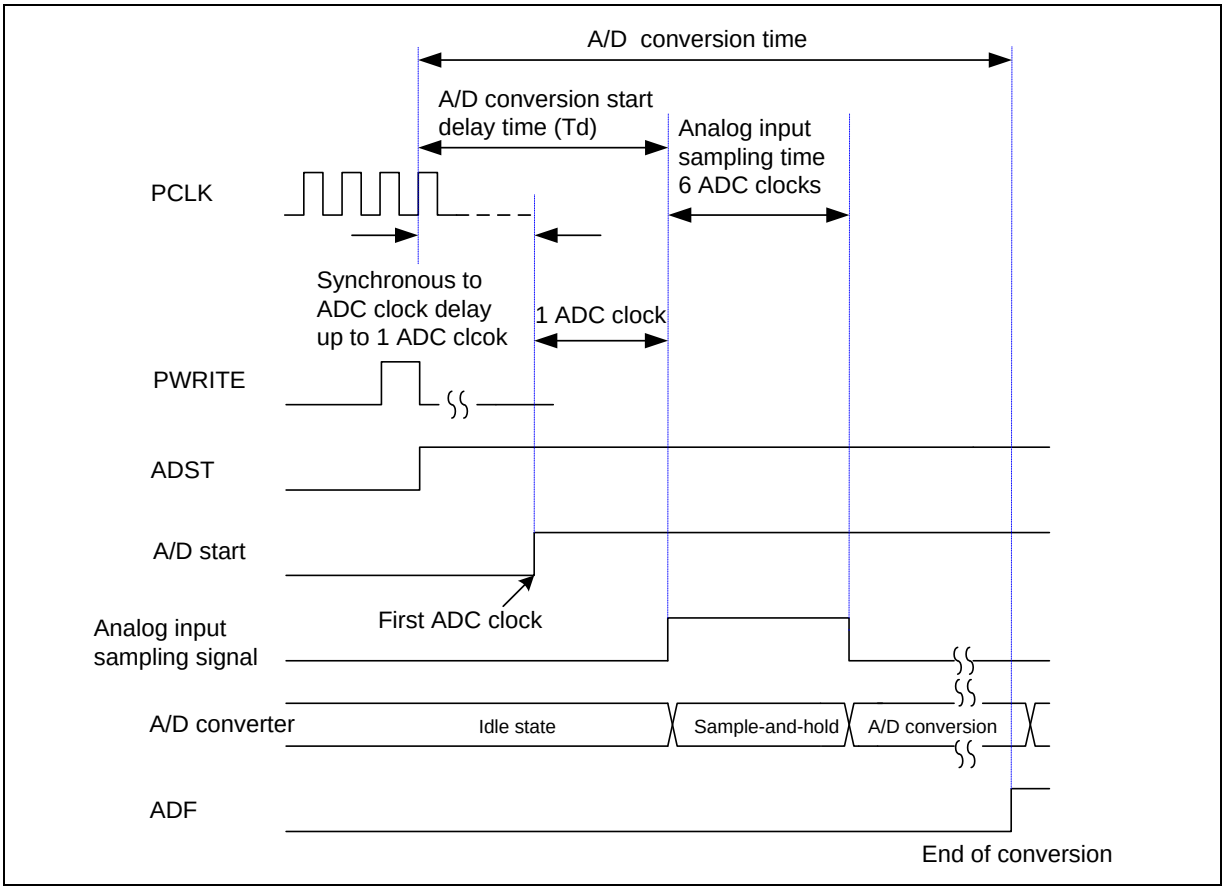


Figure 5-37 Conversion Start Delay Timing Diagram

5.14.5.6 Conversion Result Monitor

I91032 ADC controller provides two compare registers ADC_CMP0/1 to monitor specified channel conversion result from A/D conversion module (see figure below). Software can select which channel to be monitored by set CMPCH and CMPCOND bit is used to check conversion result is less than or greater than (equal to) specified value in CMPDAT. When the compared result meets the setting, compare match counter will increase 1. When counter value reaches the setting of CMPMCNT then ADCMPF bit will be set to 1. If ADCMPIE bit is set, then an ADINT interrupt request is generated. The block diagram is shown as below:

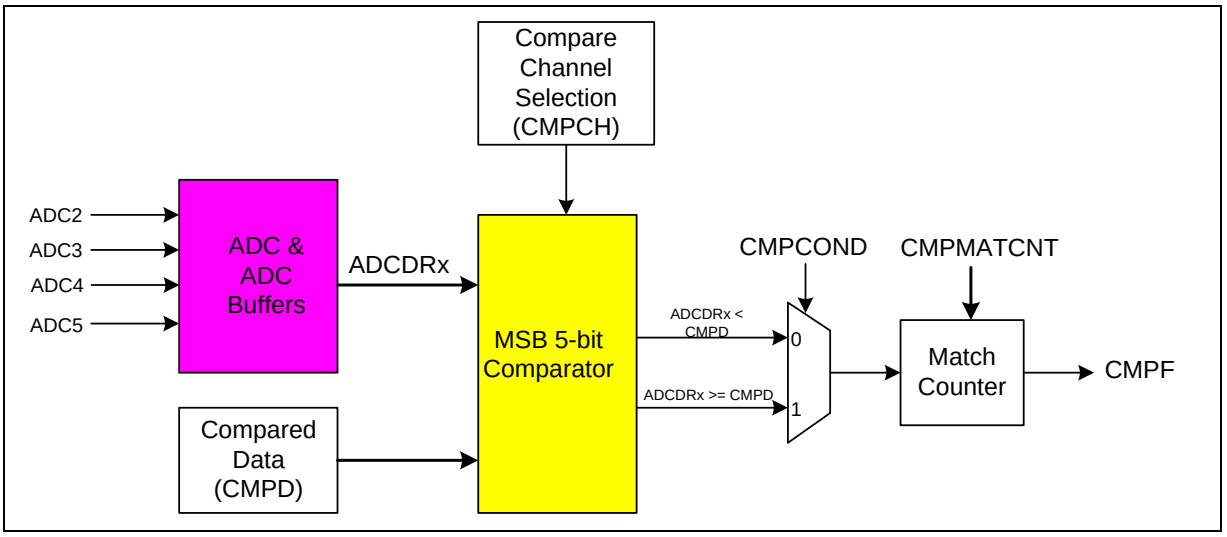


Figure 5-38 A/D Conversion Result Comparison

5.14.5.7 Interrupt Sources

The A/D converter generates a conversion end ADIF in ADC_STATUS register upon the end of A/D conversion. If ADCIE bit in ADC_CTL is set then conversion end interrupt request ADINT is generated. If ADCMPIE bit is enabled, when A/D conversion result meets setting in ADC_CMPn register, monitor interrupt is generated, ADINT will be set also. CPU can clear ADCMPF and ADIF to stop interrupt request.

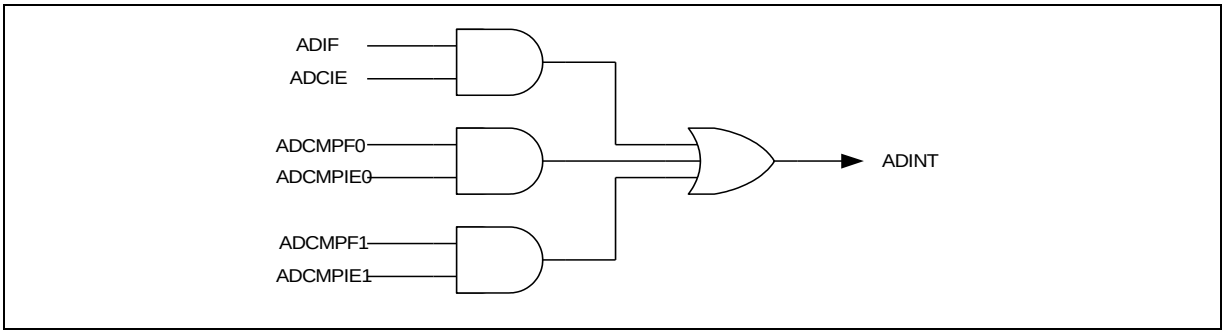


Figure 5-39 A/D Controller Interrupt

5.14.6 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 capacitor should be placed on this pin and returned to analog ground (VSSA) as shown in Figure 5.39.

Before using the ADC, PGC analog blocks, the VMID reference needs to be enabled. A low impedance option allows fast charging of the external noise de-coupling capacitor, while a higher impedance options provides lower power consumption. A pulldown option allows the reference to be discharged when off.

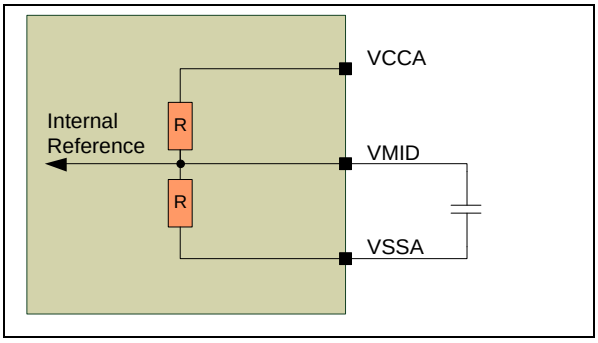


Figure 5-40 VMID Reference Generation

5.14.7 Microphone Bias Generator

The I91032 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 MICBSEL[1:0] to select bias voltages from 50% to 90% of the VCCA supply voltage (see description below). The user should consider the microphone manufacturers specification in deciding on the optimum MICBIAS voltage to use. Generally, a microphone will require a current of 0.1mA to a maximum 0.5mA and a voltage of 1V to 3V across it.
Referring to the application diagram of Figure 5.42, external resistor R1 and R2 values are selected to limit the current to maximum 1mA that can be provided by MICBIAS. MICBIAS output voltage should be such that the following condition is met:

VMICBIAS > Vs + (R1 + R2) x IMIC

Where Vs is the desired voltage across the microphone from specitication and IMIC is the current through the microphone (0.1-0.5mA)

From Figure 5.41, MIC_IN1 and MIC_IN2 are AC coupled to the MIC+ and MIC- respectively for differential inputs. In single-ended operation, MIC_IN1 should go to MIC-. C1 and C2 are AC coupling capacitors. In single-ended application R2, can be removed and R1 increased to cover R2. For improved performance, it is recommended to keep R2 to provide additional rejection from ground noise.

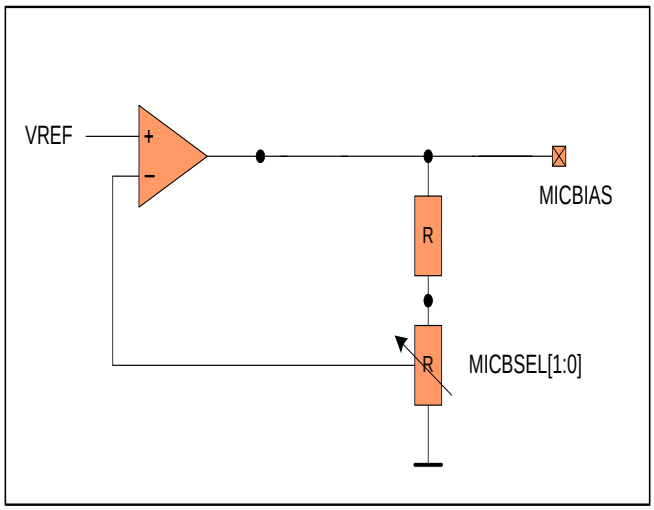


Figure 5-41 MICBIAS Block Diagram

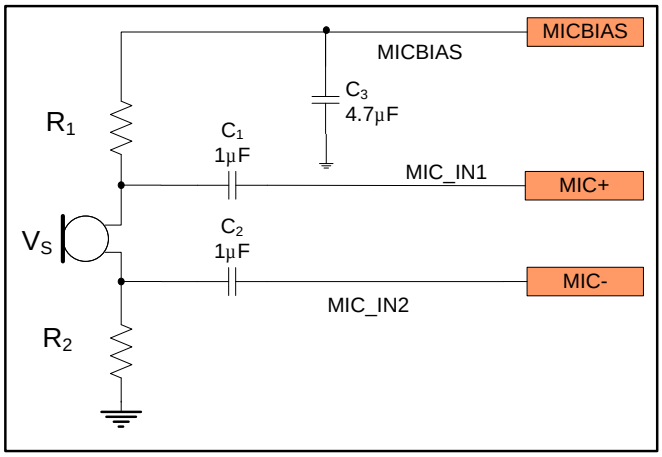


Figure 5-42 MICBIAS Application Diagram

5.14.8 ADC Pre-amplifier Block Diagram

For voice recording application, the I91032 comes equipped with a microphone bias output, and programmable gain amplifier control (PGA). This block is controlled by bits in the ADC_PGATL register.The MICN and MICP pins can connect to differential input of PGA which drives the AIN4 input of ADC.The PGA provides a gain from -18dB to +45dB,in increments of 1 dB steps using a 6-bit control, PGA_SEL[5:0]. The gain is monotonically increasing with 0x00 for lowest gain (-18dB) and 0x3f for the maximum gain (45dB). The signal path is enabled by powering up the gain elements (EN_PGA,). Input to the PGA can be either differential or single-ended on the MICN input

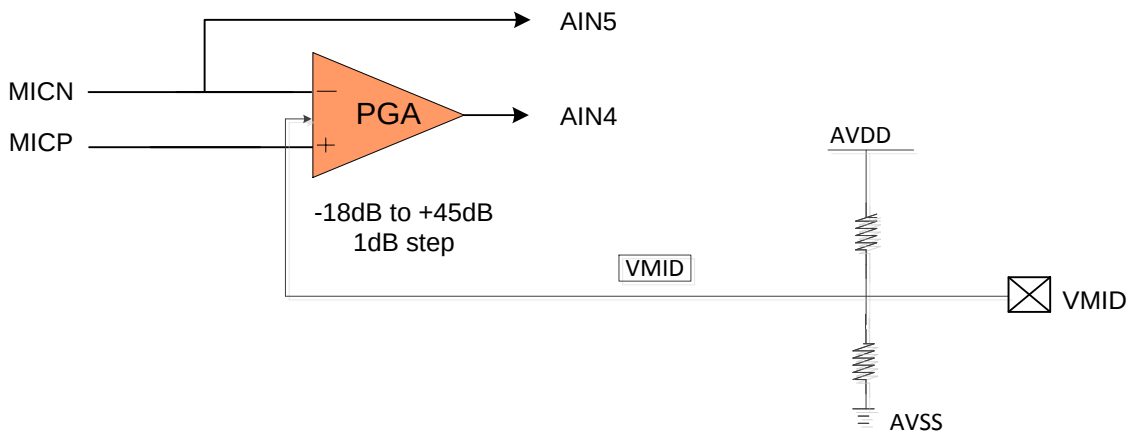


Figure 5-43 Pre-amplifier

5.14.9 ADC, Pre-amplifier and Microphone Bias Register Map

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

Register	Offset	R/W	Description	Reset Value
ADC Base Address: ADC_BA = 0x400E_0000				
ADC_DAT0	ADC_BA+0x00	R	A/D Data Register for the channel defined in CHSEQ0	0x0000_0000
ADC_DAT1	ADC_BA+0x04	R	A/D Data Register for the channel defined in CHSEQ1	0x0000_0000
ADC_DAT2	ADC_BA+0x08	R	A/D Data Register for the channel defined in CHSEQ2	0x0000_0000
ADC_DAT3	ADC_BA+0x0C	R	A/D Data Register for the channel defined in CHSEQ3	0x0000_0000
ADC_DAT4	ADC_BA+0x10	R	A/D Data Register for the channel defined in CHSEQ4	0x0000_0000
ADC_DAT5	ADC_BA+0x14	R	A/D Data Register for the channel defined in CHSEQ5	0x0000_0000
ADC_DAT6	ADC_BA+0x18	R	A/D Data Register for the channel defined in CHSEQ6	0x0000_0000
ADC_DAT7	ADC_BA+0x1C	R	A/D Data Register for the channel defined in CHSEQ7	0x0000_0000
ADC_CTL	ADC_BA+0x20	R/W	A/D Control Register	0x0000_0000
ADC_CHSEQ	ADC_BA+0x24	R/W	A/D Channel Sequence Register	0xFFFF_FFFF
ADC_CMP0	ADC_BA+0x28	R/W	A/D Compare Register 0	0x0000_0000
ADC_CMP1	ADC_BA+0x2C	R/W	A/D Compare Register 1	0x0000_0000
ADC_STATUS	ADC_BA+0x30	R/W	A/D Status Register	0x0000_0070
ADC_PDMA	ADC_BA+0x34	R/W	ADC PDMA Control Reigster	0x0000_0000
ADC_PGATL	ADC_BA+0x3C	R/W	ADC Pre-amplifier Gain Control Register	0x0000_0900
ADC_VMID	ADC_BA+0x40	R/W	ADC VMID Control Register	0x0000_0006
ADC_HWPARA	ADC_BA+0x44	R/W	ADC H/W Parameter Control Register	0x0000_0B00

A/D Data Registers (ADC_DATn)

Register	Offset	R/W	Description	Reset Value
ADC_DAT0	ADC_BA+0x00	R	A/D Data Register for the channel defined in CHSEQ0	0x0000_0000
ADC_DAT1	ADC_BA+0x04	R	A/D Data Register for the channel defined in CHSEQ1	0x0000_0000
ADC_DAT2	ADC_BA+0x08	R	A/D Data Register for the channel defined in CHSEQ2	0x0000_0000
ADC_DAT3	ADC_BA+0x0C	R	A/D Data Register for the channel defined in CHSEQ3	0x0000_0000
ADC_DAT4	ADC_BA+0x10	R	A/D Data Register for the channel defined in CHSEQ4	0x0000_0000
ADC_DAT5	ADC_BA+0x14	R	A/D Data Register for the channel defined in CHSEQ5	0x0000_0000
ADC_DAT6	ADC_BA+0x18	R	A/D Data Register for the channel defined in CHSEQ6	0x0000_0000
ADC_DAT7	ADC_BA+0x1C	R	A/D Data Register for the channel defined in CHSEQ7	0x0000_0000

The A/D converted results are saved in these ADC_DATn registers sequentially. The channel scan sequence is defined in ADC_HSR register.

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

Bits	Description	
[31:18]	**Reserved**	Reserved.

[17]	**VALID**	**Valid Flag** 0 = Data in RESULT are not valid. 1 = Data in RESULT are valid. This bit is set to 1 when corresponding channel analog input conversion is completed and cleared by hardware after ADC_DAT register is read.
[16]	**OV**	**Over Run Flag** 0 = Data in RESULT are recent conversion result. 1 = Data in RESULT are overwritten. If converted data in RESULT[11:0] have not been read before new conversion result is loaded to this register, OV is set to 1. It is cleared by hardware after ADC_DAT register is read.
[15:12]	**EXTS**	**Extension Bits Of RESULT for Different Data Format** If ADCFM is “0”, EXTS all are read as “0”. If ADCFM is “1”, EXTS all are read as bit RESULT[11].
[11:0]	**RESULT**	**A/D Conversion Result** This field contains the 12-bit conversion result. Its data format is defined by ADCFM bit.

A/D Control Register (ADC_CTL)

Register	Offset	R/W	Description	Reset Value
ADC_CTL	ADC_BA+0x20	R/W	A/D Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HP_EN	**HP_FSEL**			**DS_EN**	**Reserved**	**DS_RATE**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**ADCFM**	**SWTRG**	**Reserved**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**PDMA_EN**	**OPMODE**		**ADCIE**	**ADCEN**

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**HP_EN**	**High-pass Filter Enable** 0 = High-pass filter is disabled 1 = High-pass filter is enabled **(must in continuous scan mode)**
[22:20]	**HP_FSEL**	**High-pass Filter Frequency Selection:** 000 = Do not remove DC part 001 = DC part is suppressed by -40dB, -3dB at 0.005 x Sampling Rate 010 = DC part is suppressed by -40dB, -3dB at 0.010 x Sampling Rate 011 = DC part is suppressed by -40dB, -3dB at 0.014 x Sampling Rate 100 = DC part is suppressed by -40dB, -3dB at 0.019 x Sampling Rate 101 = DC part is suppressed by -40dB, -3dB at 0.023 x Sampling Rate 110 = DC part is suppressed by -40dB, -3dB at 0.027 x Sampling Rate 111 = DC part is suppressed by -40dB, -3dB at 0.032 x Sampling Rate
[19]	**DS_EN**	**Down Sample Function Enable** 0 = Down sample function is disabled 1 = Down sample function is enabled. When this field is set, ADC will be forced to **continuous scan mode**, no matter what is specified in field OPMODE **(**ADC_CTL[3:2]**).**
[18]	**Reserved**	Reserved

[17:16]	**DS_RATE**	**Down Sample Rate** 00 = Down sample X2 01 = Down sample X4 10 = Down sample X8 11 = Down sample X16
[15:13]	**Reserved**	Reserved
[12]	**ADCFM**	**Data Format Of ADC Conversion Result** 0 = Unsigned 1 = 2'Complemet
[11]	**SWTRG**	**A/D Conversion Start** 0 = Conversion is stopped and A/D converter enters idle state. 1 = Start conversion. **Note**: SWTRG bit can be reset to 0 by software, or can be cleared to 0 by hardware automatically at the end of single mode and single-cycle scan mode on specified channel. In continuous scan mode, A/D conversion is continuously performed sequentially until software writes 0 to this bit or chip resets.
[10:5]	**Reserved**	Reserved.
[4]	**PDMAEN**	**PDMA Transfer Enable Bit** When A/D conversion is completed, the converted data is loaded into ADC_DATn (n: 0 ~ 7) register, user can enable this bit to generate a PDMA data transfer request. 0 = PDMA data transfer Disabled. 1 = PDMA data transfer Enabled.
[3:2]	**OPMODE**	**A/D Converter Operation Mode** 00 = Single conversion 01 = Reserved 10 = Single-cycle scan 11 = Continuous scan **Note 1**: This field will be effective only when DS_EN field in this register is set as "0". When DS_EN is set as "1", ADC conversion will be forced to "continuous scan mode" **Note 2**: When changing the operation mode, software should disable SWTRG bit firstly.
[1]	**ADCIE**	**A/D Interrupt Enable** 0 = Disable A/D interrupt function 1 = Enable A/D interrupt function A/D conversion end interrupt request is generated if ADCIE bit is set to 1.

[0]	**ADCEN**	**A/D Converter Enable** 0 = Disable 1 = Enable Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit power consumption.

A/D Channel Sequence Register (ADC_CHSEQ)

Register	Offset	R/W	Description	Reset Value
ADC_CHSEQ	ADC_BA+0x24	R/W	A/D Channel Sequence Register	0xFFFF_FFFF

31	30	29	28	27	26	25	24
CHSEQ7				**CHSEQ6**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CHSEQ5				**CHSEQ4**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CHSEQ3				**CHSEQ2**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHSEQ1				**CHSEQ0**			

Bits	Description	
[31:28]	**CHSEQ7**	**Select Channel N As The 8th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[27:24]	**CHSEQ6**	**Select Channel N As The 7th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[23:20]	**CHSEQ5**	**Select Channel N As The 6th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[19:16]	**CHSEQ4**	**Select Channel N As The 5th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[15:12]	**CHSEQ3**	**Select Channel N As The 4th Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[11:8]	**CHSEQ2**	**Select Channel N As The 3rd Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.
[7:4]	**CHSEQ1**	**Select Channel N As The 2nd Conversion In Scan Sequence** The definition of channel selection is the same as CHSEQ0.

[3:0]	**CHSEQ0**	**Select Channel N As The 1st Conversion In Scan Sequence** If CHSEQ0[3]=0, one of the following channel is selected according to CHSEQ0[2:0]. If CHSEQ0[3] =1, this is dedicated to select pre-amplifier output as ADC input. For microphone application, the audio signal from microphone circuit is connected to the differential input pair of pre-amplifier, so programmer must set CHSEQ0[3]=1 in order to process the audio signal. For CHSEQ0[0] =0, Only CHSEQ0[2:1].={1,0} is meaningful. The output of pre-amplifier with differential input pair is selected as ADC input. For CHSEQ0[0].=1, CHSEQ0 = 1xx1: No channel is selected, scan sequence is stopped.

CHSEQ0	Selected channel to ADC input
0000	Band Gap/ Channel 0
0001	Channel 1
0010	Channel 2
0011	Channel 3
0100	Reserved
0101	GND
0110	Channel 6
0111	Channel 7

CHSEQ0	Selected channel to ADC input
1000	Reserved
1010	Reserved
1100	Pre-amplifier output
1110	None

A/D Compare Register 0/1 (ADC_CMPn)

Register	Offset	R/W	Description	Reset Value
ADC_CMP0	ADC_BA+0x28	R/W	A/D Compare Register 0	0x0000_0000
ADC_CMP1	ADC_BA+0x2C	R/W	A/D Compare Register 1	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:23]	**CMPDAT**	**Compare Data** This field possessing the 5 MSB of 12-bit compare data, and 7 LSB are treated as "0", is used to compare with conversion result of specified channel. Software can use it to monitor the external analog input pin voltage transition in scan mode without imposing a load on software. The data format should be consistent with the setting of ADCFM bit.
[22:12]	**Reserved**	Reserved.
[11:8]	**CMPMCNT**	**Compare Match Count** When the specified A/D channel analog conversion result matches the comparing condition, the internal match counter will increase 1. When the internal counter achieves the setting, (CMPMCNT+1) hardware will set the ADCMPF bit.
[7:6]	**Reserved**	Reserved.

[5:3]	**CMPCH**	**Compare Channel Selection** 000 = Channel 0 conversion result is selected to be compared. 001 = Channel 1 conversion result is selected to be compared. 010 = Channel 2 conversion result is selected to be compared. 011 = Channel 3 conversion result is selected to be compared. 100 = Reserved. 101 =The conversion result of pre-amplifier output is selected to be compared. 110 = Channel 6 conversion result is selected to be compared. 111 = Channel 7 conversion result is selected to be compared.
[2]	**CMPCOND**	**Compare Condition** 0 = ADCMPFx bit is set if conversion result is less than CMPDAT. 1 = ADCMPFx bit is set if conversion result is greater or equal to CMPDAT,
[1]	**ADCMPIE**	**Compare Interrupt Enable** 0 = Disable 1 = Enable When converted data in RESULT is less (or greater) than the compare data CMPDAT, ADCMPF bit is asserted. If ADCMPIE is set to 1, a compare interrupt request is generated.
[0]	**ADCMPEN**	**Compare Enable** 0 = Disable compare. 1 = Enable compare. Set this bit to 1 to enable the comparison CMPDAT with specified channel conversion result when converted data is loaded into ADC_DAT register.

A/D Status Register (ADC_STATUS)

Register	Offset	R/W	Description	Reset Value
ADC_STATUS	ADC_BA+0x30	R/W	A/D Status Register	0x0000_0070

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
OV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VALID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CHANNEL**			**BUSY**	**ADCMPF1**	**ADCMPF0**	**ADIF**

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**OV**	**Over Run Flag** It is a mirror to OV bit in ADC_DATn.
[15:8]	**VALID**	**Data Valid Flag** It is a mirror of VALID bit in ADC_DATn.
[7]	**Reserved**	Reserved.
[6:4]	**CHANNEL**	**Current Conversion Channel** This filed reflects current conversion channel when BUSY=1. When BUSY=0, it shows the next channel will be converted. It is read only.
[3]	**BUSY**	**BUSY/IDLE** 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion. This bit is mirror of SWTRG bit in ADC_CTL. It is read only.
[2]	**ADCMPF1**	**Compare Flag** When the selected channel A/D conversion result meets setting conditions in ADC_CMP1, then this bit is set to 1. And it is cleared by write 1. 0 = Converted result RESULT in ADC_DAT does not meet ADC_CMP1 setting. 1 = Converted result RESULT in ADC_DAT meets ADC_CMP1 setting,

[1]	**ADCMPF0**	**Compare Flag** When the selected channel A/D conversion result meets setting conditions in ADC_CMP0, then this bit is set to 1. And it is cleared by write 1. 0 = Converted result RESULT in ADC_DAT does not meet ADC_CMP0 setting. 1 = Converted result RESULT in ADC_DAT meets ADC_CMP0 setting,
[0]	**ADIF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. ADIF is set to 1 under the following two conditions: 1. When A/D conversion ends in single mode, 2. When A/D conversion ends on all channels specified by channel sequence register in scan mode. And it is cleared when 1 is written.

ADC PDMA Result Register (ADC PDMA)

Register	Offset	R/W	Description	Reset Value
ADC_PDMA	ADC_BA+0x34	R/W	ADC PDMA result Register	0x0000_0070

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PDMA_RESULT**	**ADC PDMA transfer data** If DW_EN is “0” and HPF_EN is “0”, transfer SAR output to SRAM If DW_EN is “1” and HPF_EN is “0”, transfer DW output to SRAM If HPF_EN is “1”, transfer HPF output to SRAM

Pre-amplifier Gain Control Register (ADC_PGATL)

Register	Offset	R/W	Description	Reset Value
ADC_PGATL	ADC_BA+0x3C	R/W	ADC Pre-amplifier Gain Control Register	0x0000_0900

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**PGA_SEL**	PGA Gain. -18dB to +45dB, 1dB per step. 0x0000 = -18dB 0x3f=45dB
[23:19]	**Reserved**	Reserved
[18:17]	**MICB_VSEL**	Select MIC BIAS level. 0: 0%,1:65%,2:70%,3:50% of VCCA
[16]	**MICB_EN**	1: Enable MIC_BIAS
[15:14]	**Reserved**	Reserved
[13:12]	**IBGEN_TRIM**	Set to 0
[11]	**PD_IBEN**	1: Power down analog bias generation
[10]	**EN_PGA**	1: Enable PGA 0: Disable PGA
[9]	**SAR_VREF**	0: VREF=VDDA 1: VREF=MIC_BIAS
[8:4]	**Reserved**	Reserved
[3]	**ZERO_CROSS**	1: Gain update only on zero crossing. 0: immediate
[2]	**Reserved**	Reserved
[1]	**Reserved**	Reserved
[0]	**Reserved**	Reserved

ADC VMID Control Register (ADC_VMID)

Register	Offset	R/W	Description	Reset Value
ADC_VMID	ADC_BA+0x40	R/W	ADC VMID 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					**PDHIRES**	**PDLOWRES**	**PULLDWN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDHIRES**	1: Disconnect high resistance VMID reference. 0: Enable high resistance VMID reference.
[1]	**PDLOWRES**	1: Disconnect low resistance VMID reference. 0: Enable low resistance VMID reference.
[0]	**PULLDWN**	1: Pull down VMID reference to 0V.

ADC H/W Parameter Control Register (ADC_HWPARA)

Register	Offset	R/W	Description	Reset Value
ADC_HWPARA	ADC_BA+0x44	R/W	ADC H/W Parameter Control Register	0x0000_1400

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[14:8]	**CONV_N**	**Specify ADC conversion clock number** ADC Conversion clock number = (CONV_N + 1). CONV_N has to be equal to or great than 11. To update this field, programmer can only revise bit [14:8] and keep other bits the same as before. Note: CONV_N valid range is from 11~127
[7:6]	**Reserved**	Reserved.
[5:0]	**SHCLK_N**	**Specify the high level of ADC start signal.** ADC start signal high level duration time = ADC_CLK x (SHCLK_N + 1). **Note**: Suggested and default value is 0.

5.15 PDMA Controller

5.15.1 Overview

The I91032 incorporates a Peripheral Direct Memory Access (PDMA) controller that transfers data between SRAM and APB devices. The PDMA has two channels of DMA PDMA (CH0~CH1). 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, ADC and DPWM.

PDMA operation is controlled for each channel by configuring a source and destination address and specifying a number of bytes to transfer. Source and destination addresses can be fixed, automatically increment or wrap around a circular buffer. When PDMA operation is complete, controller can be configured to provide CPU with an interrupt.

5.15.2 Features

- Provides access to SPI, ADC and DPWM peripherals.
- AMBA AHB master/slave interface, transfers can occur concurrently with CPU access to flash memory.
- PDMA source and destination addressing modes allow fixed, incrementing, and wrap-around addressing.

5.15.3 Block Diagram

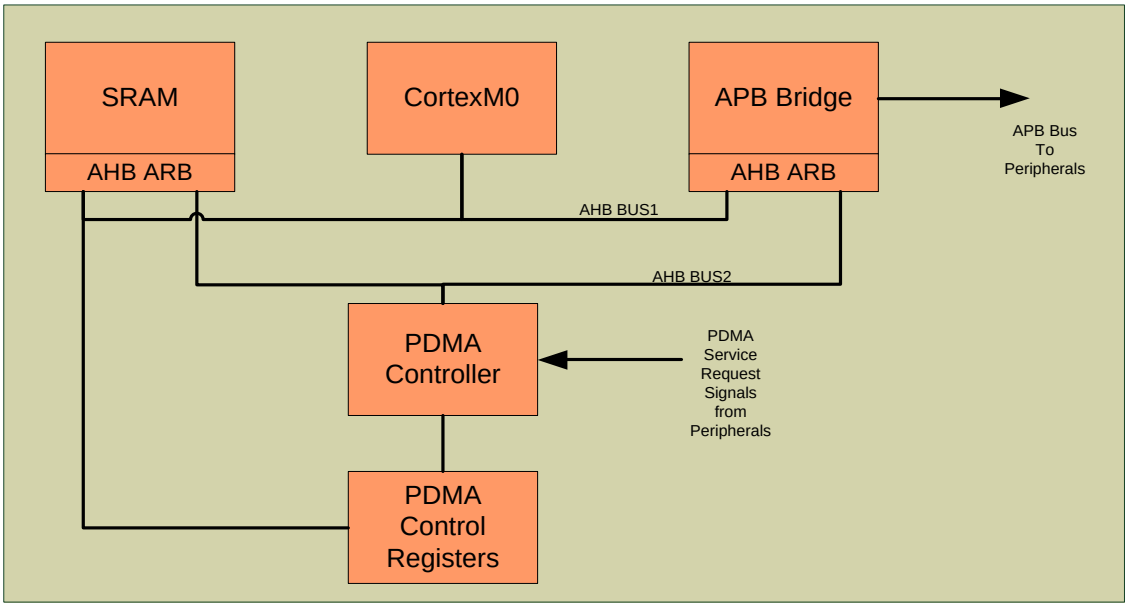


Figure 5-44 PDMA Controller Block Diagram

5.15.4 Function Description

The PDMA controller has four channels of DMA, each channel can be configured to one of the following transfer types: Peripheral-to-SRAM SRAM-to-Peripheral or SRAM-to-SRAM. The SRAM and the AHB-APB bus bridge each have an AHB bus arbiter that allows AHB bus access to occur either from the CPU or the PDMA controller. The PDMA controller requests bus transfers over the AHB bus from one address into a single word buffer within the PDMA controller then writes this buffer to another address over the AHB bus. Peripherals with PDMA capability generate control signals to the PDMA block requesting service when they need data (Rx request) or have data to transfer (Tx request). The PDMA control registers reside in address space on the AHB bus.

Transfer completion can be determined by polling of status registers or by generation of PDMA interrupt to CPU. A transfer is set up as a specified number of bytes from a source address to a destination address. Both source and destination address can be configured as a fixed address, an incrementing address or a wrap-around buffer address.

The general procedure to operate a DMA channel is as follows:

- Enable PDMA channel *n* clock by setting PDMA->HCLK*n*_EN
- Enable PDMA channel *n* by setting PDMA->channel[*n*].CSR.PDMACEN
- Set source address in PDMA->channel[*n*].SAR
- Set destination address in PDMA->channel[*n*].DAR
- Set the transfer count in PDMA->channel[*n*].BCR
- Set transfer mode and address increment mode in PDMA->channel[*n*].CSR
- Route peripheral PDMA request signal to channel *n* in service selection register.
- Trigger transfer PDMA->channel[*n*].CSR.TRIG_EN

If the source or destination address is not in wraparound mode, the PDMA will continue the transfer until PDMA->channel[*n*].CBCR decrements to zero (CBCR is initialized to BCR, in wraparound mode, CBCR will reload and continue until PDMACEN is disabled). If an error occurs during the PDMA operation, the channel stops until software clears the error condition and sets the PDMA->channel[*n*].CSR.SW_RST bit to reset the PDMA channel. After reset the PDMACEN and TRIG_EN bits would need to be set to start a new operation.

5.15.5 PDMA Controller Register Map

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

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMA_CHx_BA = 0x5000_9000 + (0x100 * x)** **x = 0, 1** **PDMA_GCR_BA = 0x5000_9F00**				
PDMA_CSRx	PDMA_CHx_BA + 0x00	R/W	PDMA Channel x Control Register	0x0000_0000
PDMA_SARx	PDMA_CHx_BA + 0x04	R/W	PDMA Channel x Source Address Register	0x0000_0000
PDMA_DARx	PDMA_CHx_BA + 0x08	R/W	PDMA Channel x Destination Address Register	0x0000_0000
PDMA_BCRx	PDMA_CHx_BA + 0x0C	R/W	PDMA Channel x Transfer Byte Count Register	0x0000_0000
PDMA_POINTx	PDMA_CHx_BA + 0x10	R	PDMA Channel x Internal buffer pointer Register	0xXXXX_0000
PDMA_CSARx	PDMA_CHx_BA + 0x14	R	PDMA Channel x Current Source Address Register	0x0000_0000
PDMA_CDARx	PDMA_CHx_BA + 0x18	R	PDMA Channel x Current Destination Address Register	0x0000_0000
PDMA_CBCRx	PDMA_CHx_BA + 0x1C	R	PDMA Channel x Current Transfer Byte Count Register	0x0000_0000
PDMA_IERx	PDMA_CHx_BA + 0x20	R/W	PDMA Channel x Interrupt Enable Register	0x0000_0001
PDMA_ISRx	PDMA_CHx_BA + 0x24	R/W	PDMA Channel x Interrupt Status Register	0x0X0X_0000
PDMA_GCRCSR	PDMA_GCR_BA + 0x00	R/W	PDMA Global Control Register	0x0000_0000
PDMA_PDSSR	PDMA_GCR_BA + 0x04	R/W	PDMA Service Selection Control Register	0x0000_0000
PDMA_GCRISR	PDMA_GCR_BA + 0x0C	R	PDMA Global Interrupt Status Register	0x0000_0000

5.15.6 PDMA Controller Register Description

PDMA Channel x Controller Register (PSMA_CSRx)

Register	Offset	R/W	Description	Reset Value
PDMA_CSRx	PDMA_CHx_BA + 0x00	R/W	PDMA Channel x Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TRIG_EN	**Reserved**		**APB_TWS**		**Reserved**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
WRA_INT_SEL				**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAD_SEL		**SAD_SEL**		**MODE_SEL**		**SW_RST**	**PDMACEN**

Bits	Description	
[23]	**TRIG_EN**	**Trigger Enable – Start a PDMA operation.** 0 = Write: no effect. Read: Idle/Finished. 1 = Enable PDMA data read or write transfer. **Note:** When PDMA transfer completed, this bit will be cleared automatically. If a bus error occurs, all PDMA transfer will be stopped. Software must reset PDMA channel, and then trigger again.
[20:19]	**APB_TWS**	**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 **MODE_SEL** is IP to Memory mode (APB-to-Memory) or Memory to IP mode (Memory-to-APB).

[15:12]	**WRA_INT_SEL**	**Wrap Interrupt Select** x1xx: If this bit is set, and wraparound mode is in operation a Wrap Interrupt can be generated when half each PDMA transfer is complete. For example if BCR=32 then an interrupt could be generated when 16 bytes were sent. xxx1: If this bit is set, and wraparound mode is in operation a Wrap Interrupt can be generated when each PDMA transfer is wrapped. For example if BCR=32 then an interrupt could be generated when 32 bytes were sent and PDMA wraps around. x1x1: Both half and w interrupts generated.
[7:6]	**DAD_SEL**	**Destination Address Select** This parameter determines the behavior of the current destination address register with each PDMA transfer. It can either be fixed, incremented or wrapped. 00 = Transfer Destination Address is incremented. 01 = Reserved. 10 = Transfer Destination Address is fixed (Used when data transferred from multiple addresses to a single destination such as peripheral FIFO input). 11 = Transfer Destination Address is wrapped. When CBCR (Current Byte Count) equals zero, the CDAR (Current Destination Address) and CBCR registers will be reloaded from the DAR (Destination Address) and BCR (Byte Count) registers automatically and PDMA will start another transfer. Cycle continues until software sets PDMA_EN=0. When PDMA_EN is disabled, the PDMA will complete the active transfer but the remaining data in the SBUF will not be transferred to the destination address.
[5:4]	**SAD_SEL**	**Source Address Select** This parameter determines the behavior of the current source address register with each PDMA transfer. It can either be fixed, incremented or wrapped. 00 = Transfer Source address is incremented. 01 = Reserved. 10 = Transfer Source address is fixed 11 = Transfer Source address is wrapped. When CBCR (Current Byte Count) equals zero, the CSAR (Current Source Address) and CBCR registers will be reloaded from the SAR (Source Address) and BCR (Byte Count) registers automatically and PDMA will start another transfer. Cycle continues until software sets 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]	**MODE_SEL**	**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]	**SW_RST**	**Software Engine Reset** 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset the internal state machine and pointers. The contents of the control register will not be cleared. This bit will auto clear after a few clock cycles.
[0]	**PDMACEN**	**PDMA Channel Enable** Setting this bit to 1 enables PDMA's operation. If this bit is cleared, PDMA will ignore all PDMA request and force Bus Master into IDLE state. **Note:** SW_RST will clear this bit.

PDMA Channel x Source Address Register (PDMA_SARx)

Register	Offset	R/W	Description	Reset Value
PDMA_SARx	PDMA_CHx_BA + 0x04	R/W	PDMA Channel x Source Address Register	0x0000_0000

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

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

PDMA Channel x Destination Address Register (PDMA_DARx)

Register	Offset	R/W	Description	Reset Value
PDMA_DARx	PDMA_CHx_BA + 0x08	R/W	PDMA Channel x Destination Address Register	0x0000_0000

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

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

PDMA Channel x Transfer Byte Count Register (PDMA_BCRx)

Register	Offset	R/W	Description	Reset Value
PDMA_BCRx	PDMA_CHx_BA + 0x0C	R/W	PDMA Channel x Transfer Byte Count Register	0x0000_0000

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

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

PDMA Channel x Internal Buffer Pointer Register (PDMA_POINTx)

Register	Offset	R/W	Description	Reset Value
PDMA_POINTx	PDMA_CHx_BA + 0x10	R	PDMA Channel x Internal buffer pointer Register	0xXXXX_0000

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

Bits	Description	
[31:4]	Reserved	Reserved
[3:0]	**POINT**	**PDMA Internal Buffer Pointer Register (Read Only)** A PDMA transaction consists of two stages, a read from the source address and a write to the destination address. Internally this data is buffered in a 32bit register. If transaction width between the read and write transactions are different, this register tracks which byte/half-word of the internal buffer is being processed by the current transaction.

PDMA Channel x Current Source Address Register (PDMA_CSARx)

Register	Offset	R/W	Description	Reset Value
PDMA_CSARx	PDMA_CHx_BA + 0x14	R	PDMA Channel x Current Source Address Register	0x0000_0000

Bits	Description	
[31:0]	**CSAR**	**PDMA Current Source Address Register (Read Only)** This register returns the source address from which the PDMA transfer is occurring. This register is loaded from SAR when PDMA is triggered or when a wraparound occurs.

PDMA Channel x Current Destination Address Register (PDMA_CDARx)

Register	Offset	R/W	Description	Reset Value
PDMA_CDARx	PDMA_CHx_BA + 0x18	R	PDMA Channel x Current Destination Address Register	0x0000_0000

Bits	Description	
[31:0]	**CDAR**	**PDMA Current Destination Address Register (Read Only)** This register returns the destination address to which the PDMA transfer is occurring. This register is loaded from DAR when PDMA is triggered or when a wraparound occurs.

PDMA Channel x Current Transfer Byte Count Register (PDMA_CBCRx)

Register	Offset	R/W	Description	Reset Value
PDMA_CBCRx	PDMA_CHx_BA + 0x1C	R	PDMA Channel x Current Transfer Byte Count Register	0x0000_0000

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

PDMA Channel x Interrupt Enable Control Register (PDMA_IERx)

Register	Offset	R/W	Description	Reset Value
PDMA_IERx	PDMA_CHx_BA + 0x20	R/W	PDMA Channel x Interrupt Enable Register	0x0000_0001

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

Bits	Description	
[31:3]	Reserved	Reserved
[2]	**WAR_IE**	**Wraparound Interrupt Enable** If enabled, and channel source or destination address is in wraparound mode, the PDMA controller will generate a WRAP interrupt to the CPU according to the setting of CSR.WRA_INT_SEL. 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]	**BLKD_IE**	**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]	**TABORT_IE**	**PDMA Read/Write Target Abort Interrupt Enable** If enabled, the PDMA controller will generate and interrupt to the CPU whenever a PDMA transaction is aborted due to an error. If a transfer is aborted, PDMA channel must be reset to resume DMA operation. 0 = Disable PDMA transfer target abort interrupt generation. 1 = Enable PDMA transfer target abort interrupt generation.

PDMA Channel x Interrupt Status Register (PDMA_ISRx)

Register	Offset	R/W	Description	Reset Value
PDMA_ISRx	PDMA_CHx_BA + 0x24	R/W	PDMA Channel x Interrupt Status Register	0x0X0X_0000

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

Bits	Description	
[31]	**INTR**	**Interrupt Pin Status (Read Only)** This bit is the Interrupt pin status of PDMA channel.
[30:12]	Reserved	Reserved
[11:8]	**WAR_IF**	**Wrap around transfer byte count interrupt flag.** These flags are set whenever the conditions for a wraparound interrupt (complete or half complete) are met. They are cleared by writing one to the bits. 0001 = Current transfer finished flag (CBCR==0). 0100 = Current transfer half complete flag (CBCR==BCR/2).

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

PDMA Global Control Register (PDMA_GCRCSR)

Register	Offset	R/W	Description	Reset Value
PDMA_GCRCSR	PDMA_GCR_BA + 0x00	R/W	PDMA Global Control Register	0x0000_0000

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

Bits	Description	
[31:10]	Reserved	Reserved

[9:8]	**HCLK_EN**	**PDMA Controller Channel Clock Enable Control** To enable clock for channel *n* HCLK_EN[*n*] must be set. HCLK_EN[*n*]=1: Enable Channel *n* clock HCLK_EN[*n*]=0: Disable Channel *n* clock
[7:1]	Reserved	Reserved
[0]	**PDMA_RST**	**PDMA Software Reset** 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset the internal state machine and pointers. The contents of control register will 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_PDSSR)

Register	Offset	R/W	Description	Reset Value
PDMA_PDSSR	PDMA_GCR_BA + 0x04	R/W	PDMA Service Selection Control Register	0x0000_0000

PDMA peripherals have transmit and/or receive request signals to control dataflow during PDMA transfers. These signals must be connected to the PDMA channel assigned by software for use with that peripheral. For instance if PDMA Channel 1 is to be used to transfer data from memory to DPWM peripheral, then DPWM_TXSEL should be set to 1. This will route the DPWM transmit request signal to PDMA channel 1, whenever DPWM has space in FIFO it will request transmission of data from PDMA. When not used the selection should be set to 0xFF.

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**DPWM_TXSEL**		**Reserved**		**ADC_RXSEL**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**SPIM_TXSEL**		**Reserved**		**SPIM_RXSEL**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**SPI0_TXSEL**		**Reserved**		**SPI0_RXSEL**	

Bits	Description	
[31:22]	**Reserved**	Reserved
[21:20]	**DPWM_TXSEL**	**PDMA DPWM Transmit Selection** This field defines which PDMA channel is connected to DPWM peripheral transmit (PDMA destination) request. 00 = No channel select. 01 = Select channel 0. 10 = Select channel 1. 11 = Reserved.
[19:18]	**Reserved**	Reserved
[17:16]	**ADC_RXSEL**	**PDMA ADC Receive Selection** This field defines which PDMA channel is connected to ADC peripheral receive (PDMA source) request. 00 = No channel select. 01 = Select channel 0. 10 = Select channel 1. 11 = Reserved.
[15:14]	**Reserved**	Reserved
[13:12]	**SPIM_TXSEL**	**PDMA SPIM Transmit Selection** This field defines which PDMA channel is connected to SPIM peripheral transmit (PDMA destination) request. 00 = No channel select. 01 = Select channel 0. 10 = Select channel 1. 11 = Reserved.
[11:10]	**Reserved**	Reserved
[9:8]	**SPIM_RXSEL**	**PDMA SPIM Receive Selection** This field defines which PDMA channel is connected to SPIM peripheral receive (PDMA source) request. 00 = No channel select. 01 = Select channel 0. 10 = Select channel 1. 11 = Reserved.
[7:6]	**Reserved**	Reserved

[5:4]	**SPI0_TXSEL**	**PDMA SPI0 Transmit Selection** This field defines which PDMA channel is connected to SPI0 peripheral transmit (PDMA destination) request. 00 = No channel select. 01 = Select channel 0. 10 = Select channel 1. 11 = Reserved.
[3:2]	**Reserved**	Reserved
[1:0]	**SPI0_RXSEL**	**PDMA SPI0 Receive Selection** This field defines which PDMA channel is connected to SPI0 peripheral receive (PDMA source) request. 00 = No channel select. 01 = Select channel 0. 10 = Select channel 1. 11 = Reserved.

PDMA Global Interrupt Status Register (PDMA_GCRISR)

Register	Offset	R/W	Description	Reset Value
PDMA_GCRISR	PDMA_GCR_BA + 0x0C	R	PDMA Global Interrupt Status Register	0x0000_0000

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

Bits	Description	
[1:0]	**GCRISR**	**Interrupt Pin Status (Read Only)** GCRISR[*n*] is the interrupt status of PDMA channel *n*.

5.16 SPI Memory Interface Controller (SPIM)

5.16.1 Overview

The SPI Memory Interface Controller performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data received from CPU. This controller can drive up to 2 external peripherals and act as a SPI master. It can generate an interrupt signal when data transfer is finished and can be cleared by writing 1 to the interrupt flag. The active level of device/slave select signal can be chosen to low active or high active, which depends on the peripheral. Writing a divisor into the SPIM_CTL1 register can program the frequency of serial clock output to the peripheral. This controller contains four 32-bit transmit/receive buffers, and can provide 1 to 4 burst mode operation. The number of bits in each transaction can be 8, 16, 24, or 32; data can be transmitted/received up to four successive transactions in one transfer.

5.16.2 Features

- Supports SPI master mode
- 8-, 16-, 24-, and 32-bit length of transaction
- Supports standard (1-bit), dual (2-bit), and quad (4-bit) I/O transfer mode
- Provides burst mode operation, which can transmit/receive data up to four successive transactions in one transfer
- Two slave/device select lines.
- Fully static synchronous design with one clock domain

5.16.3 Block Diagram

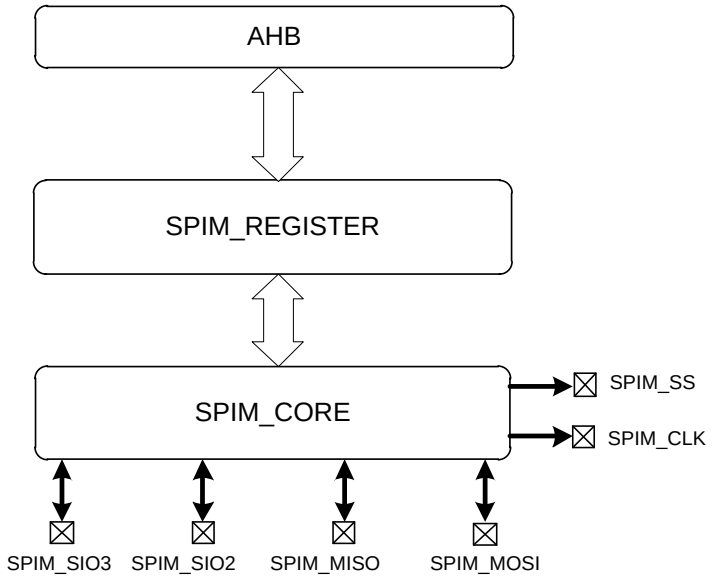


Figure 5-45 SPIM Block Diagram

5.16.3.1 SPIM Timing Diagram

The timing diagram of SPI transaction is shown as follows:

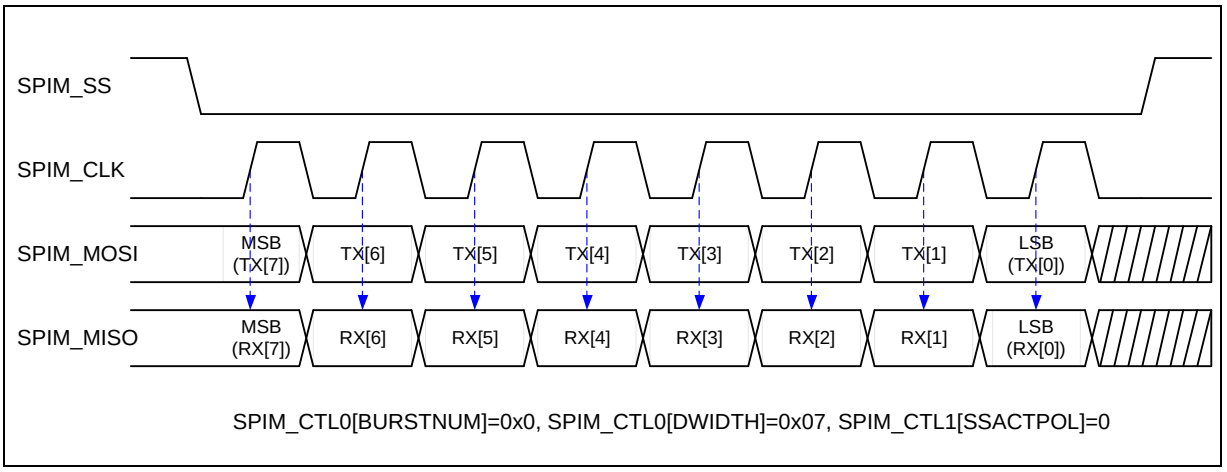


Figure 5-46 SPIM Timing Diagram

5.16.3.2 SPIM Programming Example

To access a device with the following requirements:

- Data is transferred with the MSB first.
- Only one byte is transmitted/received in a transfer.
- Chip select signal is active low.

You should do the following actions basically (refer to the specification of the device for the detailed steps):

1. Write a divisor into DIVIDER (SPIM_CTL1[31:16]) to determine the frequency of serial clock.
2. Set SSACTPOL (SPIM_CTL1[5]) to 0 activate the device you want to access.
3. When transmitting (writing) data to device:
   - QDIODIR (SPIM_CTL0[15]) = 1
   - BURSTNUM (SPIM_CTL0[14:13]) = 0x0
   - DWIDTH (SPIM_CTL0[12:8]) = 0x07
   - Write the data you want to transmit into SPIM_TX0[7:0]
4. When receiving (reading) data from device:
   - QDIODIR (SPIM_CTL0[15]) = 0
   - BURSTNUM (SPIM_CTL0[14:13]) = 0x0
   - DWIDTH (SPIM_CTL0[12:8]) = 0x07
5. Set SPIMEN (SPIM_CTL1[0]) to 1 to start the transfer.
6. Wait for interrupt or poll the SPIMEN (SPIM_CTL1[0]) until it turns to 0.

7. When receiving (reading) data from device:
   - Read out the received data from the SPIM_RX0[7:0] register.

5.16.4 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPIM Base Address: SPIM_BA = 0x5000_7000				
SPIM_CTL0	SPIM_BA+0x00	R/W	Control and Status Register 0	0x00C0_0002
SPIM_CTL1	SPIM_BA+0x04	R/W	Control Register 1	0x0000_0010
SPIM_RX0	SPIM_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPIM_RX1	SPIM_BA+0x14	R	Data Receive Register 1	0x0000_0000
SPIM_RX2	SPIM_BA+0x18	R	Data Receive Register 2	0x0000_0000
SPIM_RX3	SPIM_BA+0x1C	R	Data Receive Register 3	0x0000_0000
SPIM_TX0	SPIM_BA+0x20	R/W	Data Transmit Register 0	0x0000_0000
SPIM_TX1	SPIM_BA+0x24	R/W	Data Transmit Register 1	0x0000_0000
SPIM_TX2	SPIM_BA+0x28	R/W	Data Transmit Register 2	0x0000_0000
SPIM_TX3	SPIM_BA+0x2C	R/W	Data Transmit Register 3	0x0000_0000

Note: If software wants to write to any register of the peripheral, the SPIMEN bit of the SPIM_CTL1 register should be the status low.

5.16.5 Register Description

Control and Status Register 0 (SPIM_CTL0)

Register	Offset	R/W	Description	Reset Value
SPIM_CTL0	SPIM_BA+0x00	R/W	Control and Status Register 0	0x00C0_0002

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**BITMODE**		**SUSPITV**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**

QDIODIR	BURSTNUM		DWIDTH				
7	6	5	4	3	2	1	0
IF	IEN	Reserved				TXDMAEN	RXDMAEN

Bits	Description	
[31:22]	**Reserved**	**Reserved.**
[21:20]	**BITMODE**	**SPI Interface Bit Mode** 00 = Standard mode. 01 = Dual mode. 10 = Quad mode. 11 = Reserved. **Note.** SPIM_MOSI is Data 0 pin for Quad Mode. SPIM_MISO is Data 1 pin for Quad Mode.
[19:16]	**SUSPITV**	**Suspend Interval** These four bits provide the configuration of suspend interval between two successive transmit/receive transactions in a transfer. The default value is 0x00. When BURSTNUM = 00, setting this field has no effect on transfer. The desired interval is obtained according to the following equation (from the last falling edge of current sclk to the first rising edge of next sclk): (SUSPITV+2)*period of HCLK 0x0 = 2 HCLK clock cycles. 0x1 = 3 HCLK clock cycles. …… 0xE = 16 HCLK clock cycles. 0xF = 17 HCLK clock cycles.
[15]	**QDIODIR**	**SPI Interface Direction Select For Quad/Dual Mode** 0 = Interface signals are input. 1 = Interface signals are output.
[14:13]	**BURSTNUM**	**Transmit/Receive Burst Number** This field specifies how many transmit/receive transactions should be executed continuously in one transfer. 00 = Only one transmit/receive transaction will be executed in one transfer. 01 = Two successive transmit/receive transactions will be executed in one transfer. 10 = Three successive transmit/receive transactions will be executed in one transfer. 11 = Four successive transmit/receive transactions will be executed in one transfer.
[12:8]	**DWIDTH**	**Transmit/Receive Bit Length** This field specifies how many bits are transmitted/received in one

		transmit/receive transaction. 0x7 = 8 bits. 0xF = 16 bits. 0x17 = 24 bits. 0x1F = 32 bits. Others = Incorrect transfer result. **Note:** Only 8-, 16-, 24-, and 32-bit are allowed. Other bit length will result in incorrect transfer.
[7]	**IF**	**Interrupt Flag** Write Operation: 0 = No effect. 1 = Write 1 to clear. Read Operation: 0 = The transfer has not finished yet. 1 = The transfer has done.
[6]	**IEN**	**Interrupt Enable Control** 0 = SPIM Interrupt Disabled. 1 = SPIM Interrupt Enabled.
[5:2]	**Reserved**	**Reserved.**
[1]	**TXDMAEN**	**TX DMA Enable Control Bit** If set TXDMAEN to high, SPI interface will transfer the data to slave automatically. 0 = DMA Disabled. 1 = DMA Enable. **Note:** Only support master mode. **Note2:** Before setting RXDMAEN, user must set PDMA register correctly first.
[0]	**RXDMAEN**	**RX DMA Enable Control Bit** If set RXDMAEN to high, SPI interface will receive the data from slave automatically. 0 = DMA Disabled. 1 = DMA Enable. **Note:** Only support master mode. **Note2:** Before setting RXDMAEN, user must set PDMA register correctly first.

Control Register 1 (SPIM_CTL1)

Register	Offset	R/W	Description	Reset Value
SPIM_CTL1	SPIM_BA+0x04	R/W	Control Register 1	0x0000_0010

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

Bits	Description	
[31:16]	**DIVIDER**	**Clock Divider Register** The value in this field is the frequency divider of the system clock to generate the serial clock on the output SPIM_CLK pin. The desired frequency is obtained according to the following equation: F_SPIM_CLK = F_SYS_CLK / DIVIDER / 2. $f_{SPIM\_CLK} = \frac{f_{SYS\_CLK}}{(DIVIDER)*2}$ **Note:** When set DIVIDER to zero, the frequency of SPIM_CLK will be equal to the frequency of SYS_CLK.
[15:12]	**Reserved**	Reserved.
[14:12]	**DLYSEL**	**RX Sample Clock Source Delay Chain Select** 000 = Not Delay. 001 = Select sample clock through 2 Delay Cell. 010 = Select sample clock through 4 Delay Cell. 011 = Select sample clock through 6 Delay Cell. … 111 = Select sample clock through 14 Delay Cell.
[11:6]	**Reserved**	Reserved.
[5]	**SSACTPOL**	**Slave Select Active Level** It defines the active level of device/slave select signal (SPIM_SS). 0 = The SPIM_SS slave select signal is Active Low. 1 = The SPIM_SS slave select signal is Active High.

[4]	**SS**	**Slave Select Active Enable Control** 0 = SPIM_SS is in active level. 1 = SPIM_SS is in inactive level. **Note:** This interface can only drive one device/slave at a given time. Therefore, the slave selects of the selected device must be set to its active level before starting any read or write transfer.
[3:1]	**Reserved**	Reserved.
[0]	**SPIMEN**	**Go and Busy Status** Write Operation: 0 = No effect. 1 = Start the transfer. This bit remains set during the transfer and is automatically cleared after transfer finished. Read Operation: 0 = The transfer has done. 1 = The transfer has not finished yet. **Note:** All registers should be set before writing 1 to the SPIMEN bit. When a transfer is in progress, you should not write to any register of this peripheral.

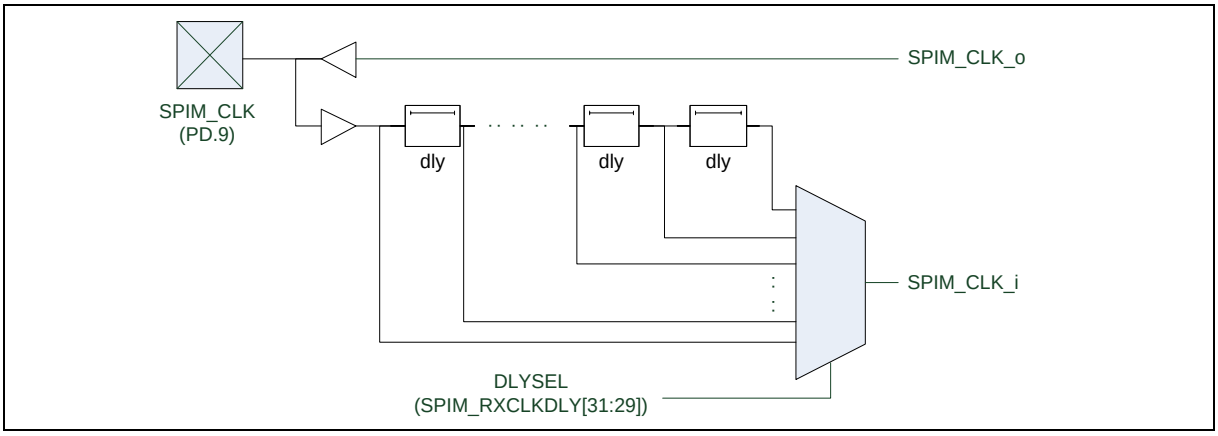


Figure 5-47 SPIM_CLK Delay Block Diagram

Because the RX data (from SPI Flash) may have some delay, using DLYSEL (SPIM_RXCLKDLY[31:29]) can adjust the sampling clock (SPIM_CLK_i) to latch the correct data.

Data Receive Register 0~3 (SPIM_RX0~3)

Register	Offset	R/W	Description	Reset Value
SPIM_RX0	SPIM_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPIM_RX1	SPIM_BA+0x14	R	Data Receive Register 1	0x0000_0000
SPIM_RX2	SPIM_BA+0x18	R	Data Receive Register 2	0x0000_0000
SPIM_RX3	SPIM_BA+0x1C	R	Data Receive Register 3	0x0000_0000

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

Bits	Description	
[31:0]	**RX**	**Data Receive Register** The Data Receive Registers hold the received data of the last executed transfer. Number of valid RX registers is specified in SPIM_CTL0[BURSTNUM]. If BURSTNUM > 0, received data are held in the most significant RX register first. Number of valid-bit is specified in SPIM_CTL0[DWIDTH]. If DWIDTH is 16, 24, or 32, received data are held in the least significant byte of RX register first. In a byte, received data are held in the most significant bit of RX register first. **Example 1:** If SPIM_CTL0[BURSTNUM] = 0x3 and SPIM_CTL1[DWIDTH] = 0x17, received data will be held in the order SPIM_RX3[23:0], SPIM_RX2[23:0], SPIM_RX1[23:0], SPIM_RX0[23:0]. **Example 2:** If SPIM_CTL0[BURSTNUM = 0x0 and SPIM_CTL0[DWIDTH] = 0x17, received data will be held in the order SPIM_RX0[7:0], SPIM_RX0[15:8], SPIM_RX0[23:16]. **Example 3:** If SPIM_CTL0[BURSTNUM = 0x0 and SPIM_CTL0[DWIDTH] = 0x07, received data will be held in the order SPIM_RX0[7], SPIM_RX0[6], …, SPIM_RX0[0].

Data Transmit Register 0~3 (SPIM_TX0~3)

Register	Offset	R/W	Description	Reset Value
SPIM_TX0	SPIM_BA+0x20	R/W	Data Transmit Register 0	0x0000_0000
SPIM_TX1	SPIM_BA+0x24	R/W	Data Transmit Register 1	0x0000_0000
SPIM_TX2	SPIM_BA+0x28	R/W	Data Transmit Register 2	0x0000_0000
SPIM_TX3	SPIM_BA+0x2C	R/W	Data Transmit Register 3	0x0000_0000

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

Bits	Description	
[31:0]	**TX**	**Data Transmit Register** The Data Transmit Registers hold the data to be transmitted in next transfer. Number of valid TX registers is specified in SPIM_CTL0[BURSTNUM]. If BURSTNUM > 0, data are transmitted in the most significant TX register first. Number of valid-bit is specified in SPIM_CTL0[DWIDTH]. If DWIDTH is 16, 24, or 32, data are transmitted in the least significant byte of TX register first. In a byte, data are transmitted in the most significant bit of TX register first. **Example 1:** If SPIM_CTL0[BURSTNUM] = 0x3 and SPIM_CTL1[DWIDTH] = 0x17, data will be transmitted in the order SPIM_TX3[23:0], SPIM_TX2[23:0], SPIM_TX1[23:0], SPIM_TX0[23:0] in next transfer. **Example 2:** If SPIM_CTL0[BURSTNUM] = 0x0 and SPIM_CTL0[DWIDTH] = 0x17, data will be transmitted in the order SPIM_TX0[7:0], SPIM_TX0[15:8], SPIM_TX0[23:16] in next transfer. **Example 3:** If SPIM_CTL0[BURSTNUM] = 0x0 and SPIM_CTL0[DWIDTH] = 0x07, data will be transmitted in the order SPIM_TX0[7], SPIM_TX0[6], …, SPIM_TX0[0] in next transfer.

5.17 UART Interface Controller

The I91032 includes a Universal Asynchronous Receiver/Transmitter (UART). The UART supports high speed operation and flow control functions as well as protocols for Serial Infrared (IrDA) and Local interconnect Network (LIN).

5.17.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART) performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the CPU. The UART controller also supports LIN(Local Interconnect Network) master mode function and IrDA SIR (Serial Infrared)function. The UART channel supports seven types of interrupts including transmitter FIFO empty interrupt(THRE_INT), receiver threshold level interrupt (RDA_INT), line status interrupt (overrun error or parity error or framing error or break interrupt) (RLS_INT), time out interrupt (TOUT_INT), MODEM status interrupt (MODEM_INT), Buffer error interrupt (BUF_ERR_INT) and LIN receiver break field detected interrupt.

The UART has a 4-byte transmit FIFO (TX_FIFO) and a 4-byte receive FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU. The CPU can read the status of the UART at any time during the operation. The reported status information includes the type and condition of the transfer operations being performed by the UART, as well as 4 error conditions (parity error, overrun error, framing error and break interrupt) that can occur while receiving data. The UART includes a programmable baud rate generator that is capable of dividing master clock input by divisors to produce the baud rate clock. The baud rate equation is Baud Rate = UART_CLK / M * [BRD + 2], where M and BRD are defined in Baud Rate Divider Register (UART0->BAUD). Table 5-lists the equations under various conditions.

The UART controller supports auto-flow control function that uses two active-low signals,/CTS (clear-to-send) and /RTS (request-to-send), to control the flow of data transfer between the UART and external devices (e.g. Modem). When auto-flow is enabled, the UART will not receive data until the UART asserts /RTS to external device. When the number of bytes in the Rx FIFO equals the value ofUART0->FCR.RTS_TRIG_LEVEL, the /RTS is de-asserted. The UART sends data out when UART controller detects /CTS is asserted from external device. If /CTS is not detected the UART controller will not send data out.

The UART controller also provides Serial IrDA (SIR, Serial Infrared) function (UART0->FUNSEL.IrDA_EN =1 to enable IrDA function). The SIR specification defines a short-range infrared asynchronous serial transmission mode with one start bit, 8 data bits, and 1 stop bit. The maximum data rate is 115.2 Kbps (half duplex). The IrDA SIR block contains an IrDA SIR Protocol encoder/decoder. The IrDA SIR protocol is half-duplex only. So it cannot transmit and receive data at the same time. The IrDA SIR physical layer specifies a minimum 10ms transfer delay between transmission and reception. This delay must be implemented by software.

The alternate function of UART controller is LIN (Local Interconnect Network) function. The LIN mode is selected by setting the UART0->FUNSEL.LIN_EN bit. In LIN mode, one start bit, 8-bit data format with 1-bit stop bit are generated in accordance with the LIN standard.

Table 5-6 UART Baud Rate Equation

Mode	DIVX_EN	DIVX_ONE	DIVX[3:0]	BRD[15:0]	Baud rate equation
0	0	0	B	A	UART_CLK / [16 * (A+2)]
1	1	0	B	A	UART_CLK / [(B+1) * (A+2)] , B ≥ 8
2	1	1	Don't care	A	UART_CLK / (A+2), A ≥ 3

Table 5-7 UART Baud Rate Setting Table

System clock = 49.152MHz						
Baud rate	Mode0	%err	Mode1	%err	Mode2	%err
921600	x		A=4,B=8	1.2	A=51	-0.6
460800	x		A=10,B=8	1.2	A=104	0.3
230400	x		A=22,B=8 A=7,B=11	1.2 1.2	A=211	-0.2
115200	A=25	1.2	A=37,B=10 A=31,B=12	0.5 0.5	A=425	0.1
57600	A=51	-0.6	A=59,B=13 A=93,B=8	0.1 0.2	A=851	0.0
38400	A=78	0.0	A=126,B=9 A=78,B=15	0.0 0.0	A=1278	0.0
19200	A=158	0.0	A=254,B=9 A=158,B=15	0.0 0.0	A=2558	0.0
9600	A=318	0.0	A=510,B=9 A=318,B=15	0.0 0.0	A=5118	0.0
4800	A=638	0.0	A=1022,B=9 A=638,B=15	0.0 0.0	A=10238	0.0

5.17.2 Features of UART controller

- UART supports 4 byte FIFO for receive and transmit data payloads.
- Auto flow control function (/CTS, /RTS) supported.
- Programmable baud-rate generator.
- Fully programmable serial-interface characteristics:
  - 5-, 6-, 7-, or 8-bit character.
  - Even, odd, or no-parity bit generation and detection.
  - 1-, 1&1/2, or 2-stop bit generation.
  - Baud rate generation.
  - False start bit detection.
- IrDA SIR Function.
- LIN master mode.

5.17.3 Block Diagram

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

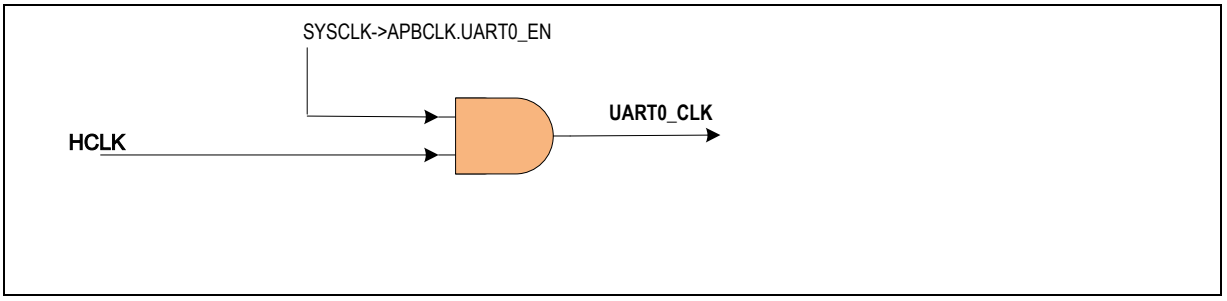


Figure 5-48 UART Clock Control Diagram

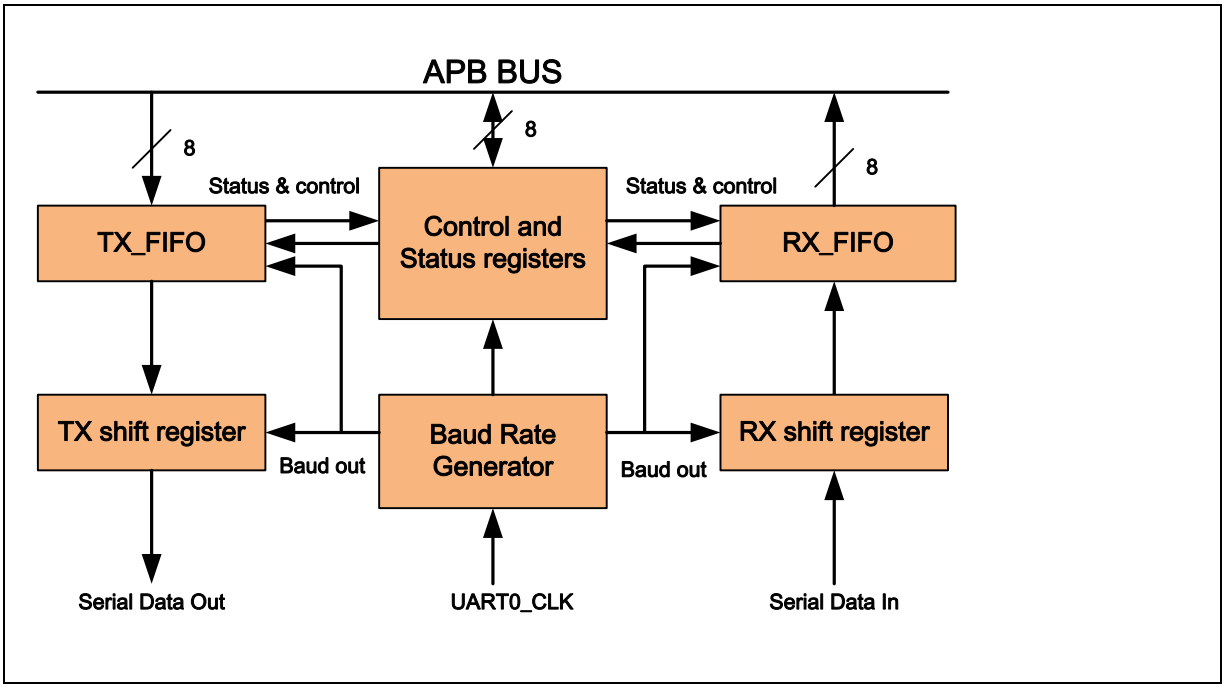


Figure 5-49 UART Block Diagram

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

RX_FIFO
The receiver is buffered with an 8 byte FIFO (plus three error bits per byte) to reduce the number of interrupts presented to the CPU.

TX shift Register
Shifts the transmit data out serially

RX shift Register
Shifts the receive data in serially

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

Baud Rate Generator
Divides the UART0_CLK clock by the divisor to get the desired baud rate clock. Refer to Table 5-for the baud rate equation.

Control and Status Register
This is a register set, including the FIFO control registers (FCR), FIFO status registers (FSR), and line control register (LCR) for transmitter and receiver. The time out control register (TOR) identifies the condition of time out interrupt. This register set also includes the interrupt enable register (IER) and interrupt status register (ISR) 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(THRE_INT), receiver threshold level reaching interrupt (RDA_INT), line status interrupt (overrun error or parity error or framing error or break interrupt) (RLS_INT) , time out interrupt (TOUT_INT), MODEM status interrupt (MODEM_INT) and Buffer error interrupt (BUF_ERR_INT).

Figure 5-50 demonstrates the auto-flow control block diagram.

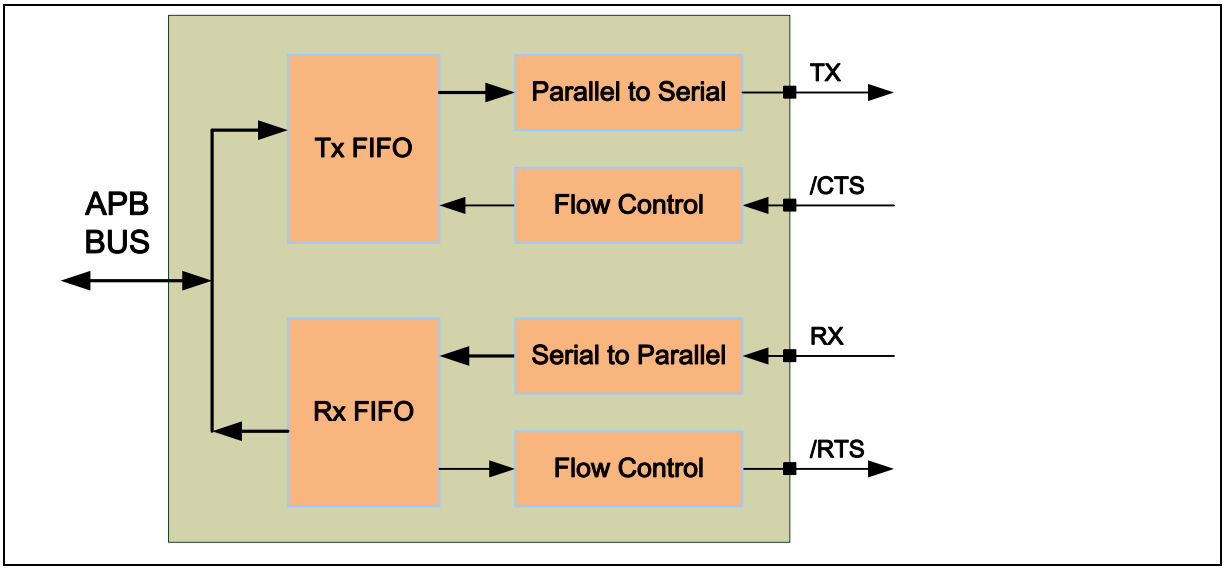


Figure 5-50 Auto Flow Control Block Diagram

5.17.4 IrDA Mode

The UART supports IrDA SIR (Serial Infrared) Transmit Encoder and Receive Decoder. IrDA mode is selected by setting the UART0->FUNSEL.IrDA_ENbit.

When in IrDA mode, the UART0->BAUD.DIVX_EN register must be zero and baud rate is given by:

Baud Rate = UART_CLK / (16 * BRD), where BRD is Baud Rate Divider in the UART0->BAUD.BRD register.

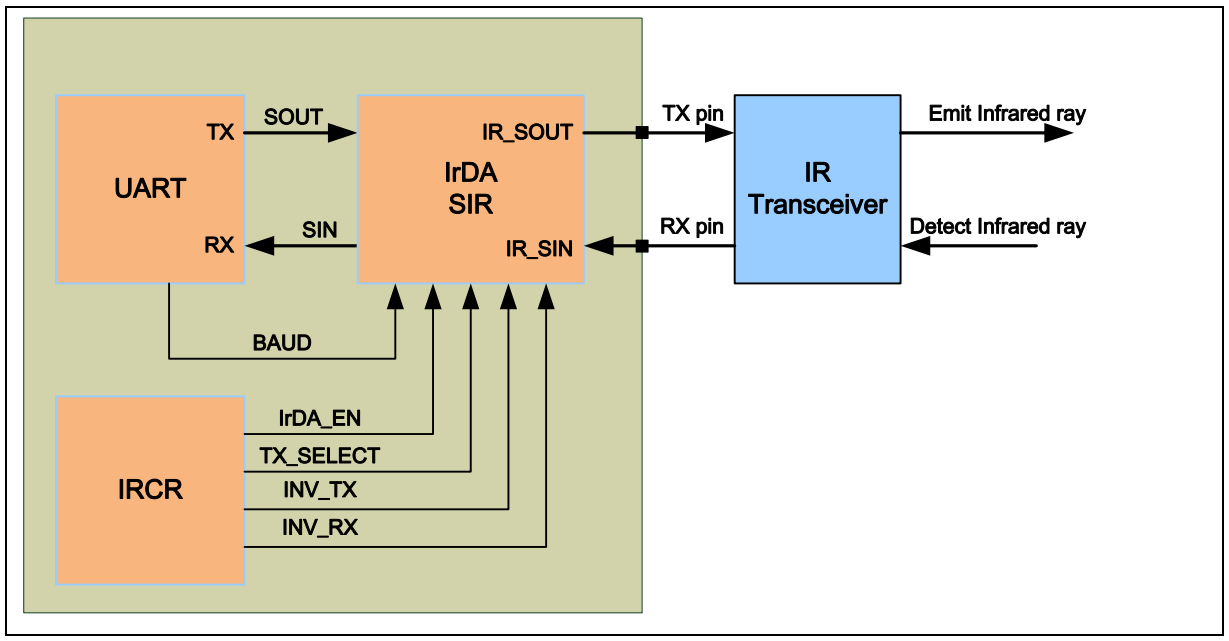


Figure 5-51 IrDA Block Diagram

- **IrDA SIR Transmit Encoder**

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

- **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, IRCR.RX_INV_EN should be set 1 by default). A start bit is detected when the IR_SIN 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. Figure 5-52 shows the IrDA encoder/decoder waveform:

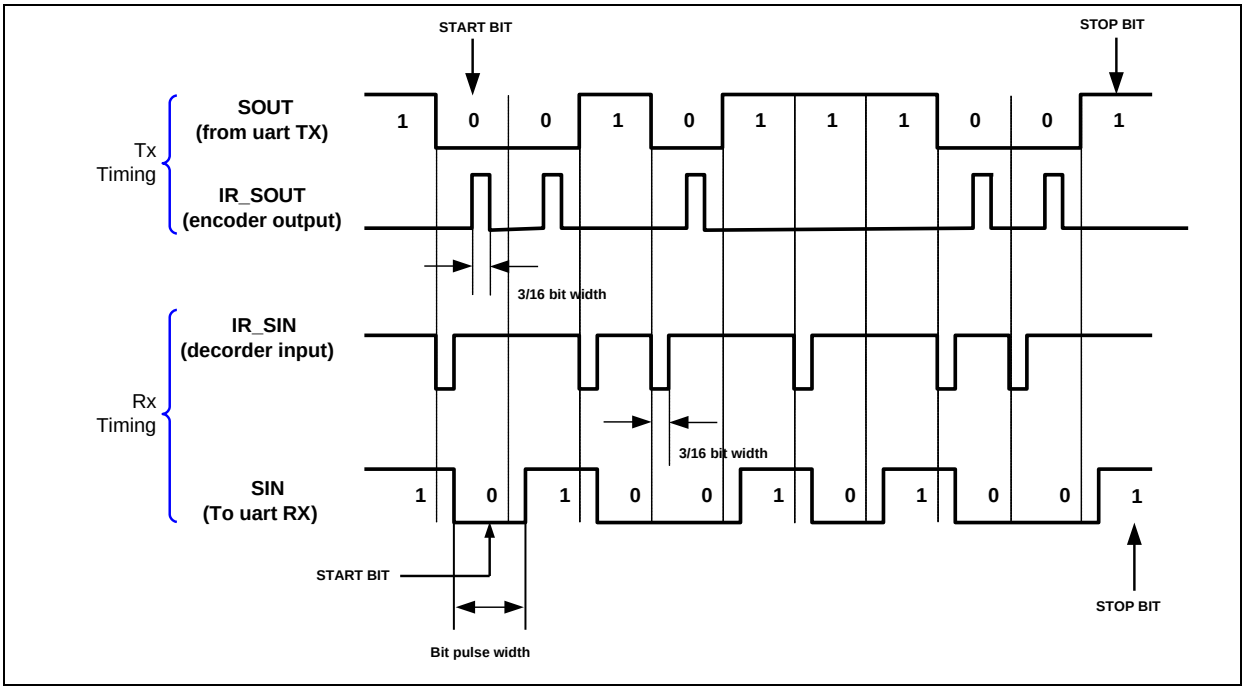


Figure 5-52 IrDA Tx/Rx Timing Diagram

5.17.5 LIN (Local Interconnection Network) mode

The UART supports a Local Interconnection Network (LIN) function. LIN mode is selected by setting the UART0->FUNSEL.LIN_EN bit. In LIN mode, each byte field is initiated by a start bit with value zero (dominant), followed by 8 data bits (LSB is first)and ended by 1 stop bit with value one (recessive) in accordance with the LIN standard (http://www.lin-subbus.org/).

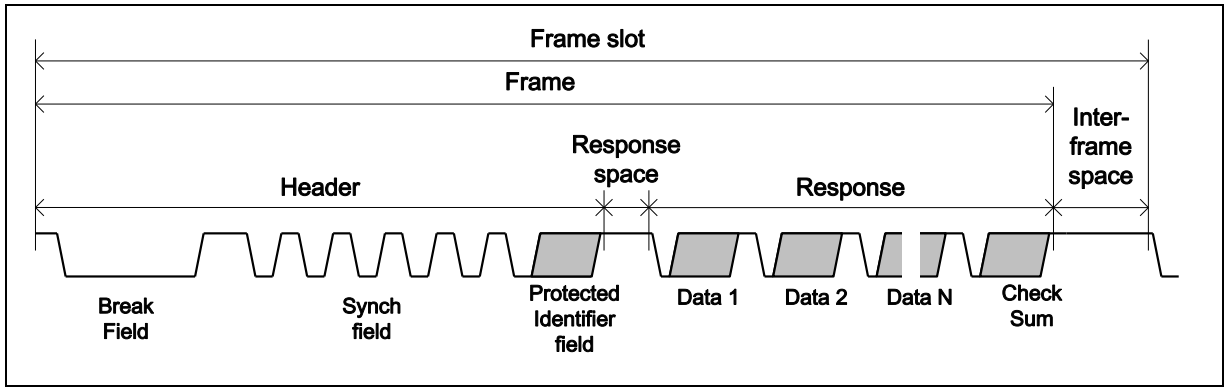


Figure 5-53 Structure of LIN Frame

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

1. Set the UART0->FUNSEL.LIN_EN bit to enable LIN Bus mode.
2. SetUART0->LINCON.LINBCNT to choose break field length. The break field length is LINBCNT+2.
3. Fill 0x55 to UART0->DATA to request synch field transmission.
4. Request Identifier Field transmission by writing the protected identifier value toUART0->DATA
5. Set the UART0->LINCON.LINTX_EN bit to start transmission (When break filed operation is finished, LINTX_EN will be cleared automatically).
6. When the STOP bit of the last byte UART0->DATA has been sent to bus, hardware will set flag UART0->FSR.TE to 1.
7. Fill N bytes data and Checksum to UART0->DATA then repeat step 5 and 6 to transmit the data.

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

1. Set the UART0->FUNSEL.LIN_EN bit to enable LIN Bus mode.
2. Set the UART0->LINCON.LINRX_EN bit register to enable LIN Rx mode.
3. Wait for the flagUART0->ISR.LIN_Rx_Break_IF to indicate Rx received Break field or not.
4. Wait for the flagUART0->ISR.RDA_IF read back the UART0->DATA register.

5.17.6 UART Interface Control Register Map

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

Register	Offset	R/W	Description	Reset Value	Reference
UART0_BA= 0x4006_0000					
DATA	UART0_BA + 0x00	R	UART0 Receive FIFO Register.	Undefined	Table 5-
DATA	UART0_BA + 0x00	W	UART0 Transmit FIFO Register.	Undefined	Table 5-
IER	UART0_BA + 0x04	R/W	UART0 Interrupt Enable Register.	0x0000_0000	Table 5-
FCR	UART0_BA + 0x08	R/W	UART0 FIFO Control Register.	0x0000_0000	Table 5-
LCR	UART0_BA + 0x0C	R/W	UART0 Line Control Register.	0x0000_0000	Table 5-
MCR	UART0_BA + 0x10	R/W	UART0 Modem Control Register.	0x0000_0000	Table 5-
MSR	UART0_BA + 0x14	R/W	UART0 Modem Status Register.	0x0000_0000	Table 5-
FSR	UART0_BA + 0x18	R/W	UART0 FIFO Status Register.	0x1040_4000	Table 5-
ISR	UART0_BA + 0x1C	R/W	UART0 Interrupt Status Register.	0x0000_0002	Table 5-
TOR	UART0_BA + 0x20	R/W	UART0 Time Out Register	0x0000_0000	Table 5-
BAUD	UART0_BA + 0x24	R/W	UART0 Baud Rate Divisor Register	0x0F00_0000	Table 5-
IRCR	UART0_BA + 0x28	R/W	UART0 IrDA Control Register.	0x0000_0040	Table 5-
LINCON	UART0_BA + 0x2C	R/W	UART0 LIN Control Register.	0x0000_0000	Table 5-
FUNSEL	UART0_BA + 0x30	R/W	UART0 Function Select Register.	0x0000_0000	Table 5-

5.17.7 UART Interface Control Register Description

Receive FIFO Data Register (DATA)

Register	Offset	R/W	Description	Reset Value
DATA	UART0_BA + 0x00	R	UART0 Receive 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
8-bit Received Data							

Table 5-8 UART Receive FIFO Data Register (DATA, address 0x4006_0000)

Bits	Descriptions	
[7:0]	8-bit Received Data	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).

Transmit FIFO Data Register (DATA)

Register	Offset	R/W	Description	Reset Value
DATA	UART0_BA + 0x00	W	UART0 Transmit FIFO Data 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
8-bit Transmit Data							

Table 5-9 UART Transmit FIFO Data Register (DATA, address 0x4006_0000)

Bits	Descriptions	
[7:0]	8-bit Transmit Data	Transmit FIFO Data Register By writing to this register, transmit data will be pushed onto the transmit FIFO. The UART will send out an 8-bit data through the Tx pin (LSB first).

Interrupt Enable Register (IER)

Register	Offset	R/W	Description	Reset Value
IER	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
DMA_RX_EN	DMA_TX_EN	AUTO_CTS_EN	AUTO_RTS_EN	TOC_EN	Reserved		LIN_RX_BRK_IEN
7	6	5	4	3	2	1	0
Reserved		BUF_ERR_IEN	RTO_IEN	MS_IEN	RLS_IEN	THRE_IEN	RDA_IEN

Table 5-10 UART Interrupt Enable Register (IER, address 0x4006_0004)

Bits	Descriptions	
[31:16]	Reserved	Reserved
[15]	DMA_RX_EN	Receive DMA Enable If enabled, the UART will request DMA service when data is available in receive FIFO.
[14]	DMA_TX_EN	Transmit DMA Enable If enabled, the UART will request DMA service when space is available in transmit FIFO.
[13]	AUTO_CTS_EN	CTS Auto Flow Control Enable 1 = Enable,0 = Disable CTS auto flow control. When CTS auto-flow is enabled, the UART will send data to external device when CTS input is asserted (UART will not send data to device until CTS is asserted).
[12]	AUTO_RTS_EN	RTS Auto Flow Control Enable 1 = Enable, 0 = Disable RTS auto flow control. When RTS auto-flow is enabled, if the number of bytes in the Rx FIFO equals FCR.RTS_TRIG_LEVEL, the UART will de-assert the RTS signal.
[11]	TOC_EN	Time-Out Counter Enable 1 = Enable, 0 = Disable Time-out counter.

[10:9]	Reserved	Reserved
[8]	LIN_RX_BRK_IEN	LIN RX Break Field Detected Interrupt Enable 0 = Mask off Lin bus Rx break field interrupt. 1 = Enable Lin bus Rx break field interrupt.
[7:6]	Reserved	Reserved
[5]	BUF_ERR_IEN	Buffer Error Interrupt Enable 0 = Mask off BUF_ERR_INT 1 = Enable IBUF_ERR_INT
[4]	RTO_IEN	Receive Time out Interrupt Enable 0 = Mask off TOUT_INT 1 = Enable TOUT_INT
[3]	MS_IEN	Modem Status Interrupt Enable 0 = Mask off MODEM_INT 1 = Enable MODEM_INT
[2]	RLS_IEN	Receive Line Status Interrupt Enable 0 = Mask off RLS_INT 1 = EnableRLS_INT
[1]	THRE_IEN	Transmit FIFO Register Empty Interrupt Enable 0 = Mask off THRE_INT 1 = Enable THRE_INT
[0]	RDA_IEN	Receive Data Available Interrupt Enable. 0 = Mask off RDA_INT 1 = Enable RDA_INT

FIFO Control Register (FCR)

Register	Offset	R/W	Description	Reset Value
FCR	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				RTS_TRIG_LEVEL			
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
RFITL				Reserved	TFR	RFR	Reserved

Table 5-11 UART FIFO Control Register (FCR, address 0x4006_0008)

Bits	Descriptions	
[31:20]	Reserved	Reserved
[19:16]	RTS_TRIG_LEVEL	RTS Trigger Level for Auto-flow Control Sets the FIFO trigger level when auto-flow control will de-assert RTS (request-to-send). RTS_Tri_Lev / Trigger Level (Bytes) 0000 / 1 0001 / 4 0010 / 8
[7:4]	RFITL	Receive FIFO Interrupt (RDA_INT) Trigger Level When the number of bytes in the receive FIFO equals the RFITL then the RDA_IF will be set and, if enabled, an RDA_INT interrupt will generated. RFITL / INTR_RDA Trigger Level (Bytes) 0000 / 1 0001 / 4 0010 / 8
[3]	Reserved	Reserved

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

Line Control Register (LCR)

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

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

Table 5-12 UART Line Control Register (LCR, address 0x4006_000C)

Bits	Descriptions	
[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 WLS[1:0] / Character length 00 / 5 bits 01 / 6 bits 10 / 7 bits 11 / 8 bits

MODEM Control Register (MCR)

Register	Offset	R/W	Description	Reset Value
MCR	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		RTS_ST	Reserved			RTS_ACT	Reserved
7	6	5	4	3	2	1	0
Reserved			LBME	Reserved		RTS_SET	Reserved

Table 5-13 UART Modem Control Register (MCR, address 0x4006_0010)

Bits	Descriptions	
[31:14]	Reserved	Reserved
[13]	RTS_ST	RTS Pin State(read only) This bit is the pin status of RTS.
[12:10]	Reserved	Reserved
[9]	RTS_ACT	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]	LBME	Loopback Mode Enable.
[3:2]	Reserved	Reserved
[1]	RTS_SET	RTS (Request-To-Send) Signal If IER.AUTO_RTS_EN=0, this bit controls whether RTS pin is active or not. 0: Drive RTS inactive (=~RTS_ACT). 1: Drive RTS active (=RTS_ACT).

Modem Status Register (MSR)

Register	Offset	R/W	Description	Reset Value
MSR	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							CTS_ACT
7	6	5	4	3	2	1	0
Reserved			CTS_ST	Reserved			DCTSF

Table 5-14 UART Modem Status Register (MSR, address 0x4006_0014)

Bits	Descriptions	
[31:9]	Reserved	Reserved
[8]	CTS_ACT	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]	CTS_ST	CTS Pin Status (read only) This bit is the pin status of CTS.
[3:1]	Reserved	Reserved
[0]	DCTSF	Detect CTS State Change Flag This bit is set whenever CTS input has state change. It will generate Modem interrupt to CPU when IER.MS_IEN=1 NOTE: This bit is cleared by writing 1 to itself.

FIFO Status Register (FSR)

Register	Offset	R/W	Description	Reset Value
FSR	UART0_BA + 0x18	R/W	UART0 FIFO Status Register.	0x1040_4000

31	30	29	28	27	26	25	24
Reserved			TE	Reserved			TX_OVF_IF
23	22	21	20	19	18	17	16
TX_FULL	TX_EMPTY	TX_POINTER					
15	14	13	12	11	10	9	8
RX_FULL	RX_EMPTY	RX_POINTER					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	Reserved			RX_OVF_IF

Table 5-15 UART FIFO Status Register (FSR, address 0x4006_0018)

Bits	Descriptions	
[31:29]	Reserved	Reserved
[28]	TE	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]	TX_OVF_IF	Tx Overflow Error Interrupt Flag If the Tx FIFO (UART0->DATA) is full, an additional write to UART0->DATA will cause an overflow condition and set this bit to logic 1. It will also generate a BUF_ERR_IF event and interrupt if enabled. NOTE: This bit is cleared by writing 1 to itself.
[23]	TX_FULL	Transmit FIFO Full(Read Only) This bit indicates whether the Tx FIFO is full or not. This bit is set when TxFIFO is full; otherwise it is cleared by hardware. TX_FULL=0 indicates there is room to write more data to Tx FIFO.

[22]	TX_EMPTY	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]	TX_POINTER	Tx FIFO Pointer (Read Only) This field returns the Tx FIFO buffer pointer. When CPU writes a byte into the TxFIFO, TX_POINTER is incremented. When a byte from Tx FIFO is transferred to the Transmit Shift Register, TX_POINTER is decremented.
[15]	RX_FULL	Receive FIFO Full(Read Only) This bit indicates whether the Rx FIFO is full or not. This bit is set when RxFIFO is full; otherwise it is cleared by hardware.
[14]	RX_EMPTY	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]	RX_POINTER	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, RX_POINTER is incremented. When one byte of Rx FIFO is read by CPU, RX_POINTER 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]	RX_OVF_IF	Rx Overflow Error Interrupt Flag If the Rx FIFO (UART0->DATA) is full, and an additional byte is received by the UART, an overflow condition will occur and set this bit to logic 1. It will also generate a BUF_ERR_IF event and interrupt if enabled. NOTE: This bit is cleared by writing 1 to itself.

Interrupt Status Register (ISR)

Register	Offset	R/W	Description	Reset Value
ISR	UART0_BA + 0x1C	R/W	UART0 Interrupt Status Register.	0x0000_0002

31	30	29	28	27	26	25	24
DMA_LIN_Rx_Break_INT	Reserved	DMA_BUF_ERR_INT	DMA_TOUT_INT	DMA_MODEM_INT	DMA_RLS_INT	Reserved	
23	22	21	20	19	18	17	16
DMA_LIN_Rx_Break_IF	Reserved	DMA_BUF_ERR_IF	DMA_TOUT_IF	DMA_MODEM_IF	DMA_RLS_IF	Reserved	
15	14	13	12	11	10	9	8
LIN_Rx_Break_INT	Reserved	BUF_ERR_INT	TOUT_INT	MODEM_INT	RLS_INT	THRE_INT	RDA_INT
7	6	5	4	3	2	1	0
LIN_Rx_Break_IF	Reserved	BUF_ERR_IF	TOUT_IF	MODEM_IF	RLS_IF	THRE_IF	RDA_IF

Table 5-16 UART Interrupt Status Register (ISR, address 0x4006_001C)

Bits	Descriptions	
[31:16]	DMA mode Bits	DMA mode equivalent following interrupt indicators and flags. See Table 5-and normal mode descriptions below. In DMA mode (either DMA transmit or receive requests are active) these bits are generated rather than the normal use bits below.
[15]	LIN_Rx_Break_INT	LIN Bus Rx Break Field Detected Interrupt Indicator to Interrupt Controller Logical AND of IER.LIN_RX_BRK_IEN and LIN_Rx_Break_IF
[14]	Reserved	Reserved
[13]	BUF_ERR_INT	Buffer Error Interrupt Indicator to Interrupt Controller Logical AND of IER.BUF_ERR_IEN and BUF_ERR_IF

[12]	TOUT_INT	Time Out Interrupt Indicator to Interrupt Controller Logical AND of IER.RTO_IEN and TOUT_IF
[11]	MODEM_INT	MODEM Status Interrupt Indicator to Interrupt Logical AND of IER.MS_IEN and MODEM_IF
[10]	RLS_INT	Receive Line Status Interrupt Indicator to Interrupt Controller Logical AND of IER.RLS_IEN and RLS_IF
[9]	THRE_INT	Transmit Holding Register Empty Interrupt Indicator to Interrupt Controller Logical AND of IER.THRE_IEN and THRE_IF
[8]	RDA_INT	Receive Data Available Interrupt Indicator to Interrupt Controller Logical AND of IER.RDA_IEN and RDA_IF
[7]	LIN_Rx_Break_IF	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]	BUF_ERR_IF	Buffer Error Interrupt Flag (Read Only) This bit is set when either the Tx or Rx FIFO overflows (FSR.TX_OVF_IF or FSR.RX_OVF_IF is set). When BUF_ERR_IF is set, the serial transfer may be corrupted. If IER.BUF_ERR_IEN is enabled a CPU interrupt request will be generated. NOTE: This bit is cleared when both FSR.TX_OVF_IF and FSR.RX_OVF_IF are cleared.
[4]	TOUT_IF	Time Out Interrupt Flag (Read Only) This bit is set when the Rx FIFO is not empty and no activity occurs in the Rx FIFO and the time out counter equal to TOIC. If IER.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]	MODEM_IF	MODEM Interrupt Flag (Read Only) This bit is set when the CTS pin has changed state (MSR.DCTSF=1). If IER.MS_IEN is enabled, a CPU interrupt request will be generated. NOTE: This bit is read only and reset when bit MSR.DCTSF is cleared by a write 1.
[2]	RLS_IF	Receive Line Status Interrupt Flag (Read Only). This bit is set when the Rx receive data has a parity, framing or break error (at least one of, FSR.BIF, FSR.FEF and FSR.PEF, is set). If IER.RLS_IEN is enabled, the RLS interrupt will be generated. NOTE: This bit is read only and reset to 0 when all bits of BIF, FEF and PEF are cleared.

[1]	THRE_IF	Transmit Holding Register Empty Interrupt Flag (Read Only). This bit is set when the last data of Tx FIFO is transferred to Transmitter Shift Register. If IER.THRE_IEN is enabled, the THRE interrupt will be generated. NOTE: This bit is read only and it will be cleared when writing data into the Tx FIFO.
[0]	RDA_IF	Receive Data Available Interrupt Flag (Read Only). When the number of bytes in the Rx FIFO equals FCR.RFITL then the RDA_IF will be set. If IER.RDA_IEN is enabled, the RDA interrupt will be generated. NOTE: This bit is read only and it will be cleared when the number of unread bytes of Rx FIFO drops below the threshold level (RFITL).

When the DMA controller is used to transmit or receive data to the UART, an alternate set of flags and interrupt indicators are generated. These are equivalent to the normal mode set above and are summarized in Table 5-.

Table 5-17 UART Interrupt Sources and Flags Table In DMA Mode

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator to Interrupt Controller	Interrupt Flag	Flag Cleared by
LIN RX Break Field Detected interrupt	LIN_RX_BRK_IEN	DMA_LIN_Rx_Break_INT	DMA_LIN_Rx_Break_IF	Write '1' to LIN_Rx_Break_IF
Buffer Error Interrupt BUF_ERR_INT	BUF_ERR_IEN	DMA_BUF_ERR_INT	DMA_BUF_ERR_IF = (TX_OVF_IF or RX_OVF_IF)	Write '1' to TX_OVF_IF/ RX_OVF_IF
Rx Timeout Interrupt TOUT_INT	RTO_IEN	DMA_TOUT_INT	DMA_TOUT_IF	Read data FIFO
Modem Status Interrupt MODEM_INT	MS_IEN	DMA_MODEM_INT	DMA_MODEM_IF = (DCTSF)	Write '1' to DCTSF
Receive Line Status Interrupt RLS_INT	RLS_IEN	DMA_RLS_INT	DMA_RLS_IF = (BIF or FEF or PEF)	Write '1' to BIF/FEF/PEF
Transmit Holding Register Empty Interrupt THRE_INT	THRE_IEN	DMA_THRE_INT	DMA_THRE_IF	Write data FIFO
Receive Data Available Interrupt RDA_INT	RDA_IEN	DMA_RDA_INT	DMA_RDA_IF	Read data FIFO

Table 5-18 UART Interrupt Sources and Flags Table In Software Mode

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator to Interrupt Controller	Interrupt Flag	Flag Cleared by
LIN RX Break Field Detected interrupt	LIN_RX_BRK_IEN	LIN_Rx_Break_INT	LIN_Rx_Break_IF	Write '1' to LIN_Rx_Break_IF
Buffer Error Interrupt BUF_ERR_INT	BUF_ERR_IEN	BUF_ERR_INT	BUF_ERR_IF = (TX_OVF_IF or RX_OVF_IF)	Write '1' to TX_OVF_IF/ RX_OVF_IF
Rx Timeout Interrupt TOUT_INT	RTO_IEN	TOUT_INT	TOUT_IF	Read data FIFO
Modem Status Interrupt MODEM_INT	MS_IEN	MODEM_INT	MODEM_IF = (DCTSF)	Write '1' to DCTSF
Receive Line Status Interrupt RLS_INT	RLS_IEN	RLS_INT	RLS_IF = (BIF or FEF or PEF)	Write '1' to BIF/FEF/PEF
Transmit Holding Register Empty Interrupt THRE_INT	THRE_IEN	THRE_INT	THRE_IF	Write data FIFO
Receive Data Available Interrupt RDA_INT	RDA_IEN	RDA_INT	RDA_IF	Read data FIFO

Time Out Register (TOR)

Register	Offset	R/W	Description	Reset Value
TOR	UART0_BA + 0x20	R/W	UART0 Time Out Register	0x0000_0000

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

Table 5-19 UART Time Out Register (TOR, address 0x4006_0020)

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

Baud Rate Divider Register (BAUD)

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

The baud rate generator takes the UART master clock UART_CLK and divides it to produce the baud rate (bit rate) clock. The divider has two division stages controlled by BRD and DIVX fields. These are configured in three modes depending on the selections of DIVX_EN and DIVX_ONE. These modes and the baud rate equations for them are described in Table 3-22.

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

Table 5-20 UART Baud Rate Divider Register (BAUD, address 0x4006_0024)

Bits	Descriptions	
[31:30]	Reserved	Reserved
[29]	DIVX_EN	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 = DIVX+1, with DIVX ≥8). Refer to Table 5-21 for more information. NOTE: When in IrDA mode, this bit must disabled.

[28]	DIVX_ONE	Divider X equal 1 0:M = DIVX+1, with restriction DIVX ≥8. 1: M = 1, with restriction BRD[15:0]≥ 3. Refer to Table 5-21 for more information.
[27:24]	DIVX	Divider X The baud rate divider M = DIVX+1.
[23:16]	Reserved	Reserved
[15:0]	BRD	Baud Rate Divider. Refer to Table 5-21 for more information.

Table 5-21 Baud Rate Equations.

Mode	DIVX_EN	DIVX_ONE	DIVX[3:0]	BRD[15:0]	Baud rate equation
0	0	0	B	A	UART_CLK / [16 * (A+2)]
1	1	0	B	A	UART_CLK / [(B+1) * (A+2)] , requires B≥ 8
2	1	1	Don't care	A	UART_CLK / (A+2), requires A ≥ 3

IrDA Control Register (IRCR)

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

7	6	5	4	3	2	1	0
Reserved	**RX_INV_EN**	**TX_INV_EN**	**Reserved**		**LOOPBACK**	**TX_SELECT**	**Reserved**

Table 5-22 UART IrDA Control Register (IRCR, address 0x4006_0028)

Bits	Descriptions	
[31:7]	Reserved	Reserved
[6]	**RX_INV_EN**	**Receive Inversion Enable** 1= Invert Rx input signal 0= No inversion
[5]	**TX_INV_EN**	**Transmit inversion enable** 1= Invert Tx output signal 0= No inversion
[4:3]	Reserved	Reserved
[2]	**LOOPBACK**	IrDA Loopback test mode. Loopback Tx to Rx.
[1]	**TX_SELECT**	**Transmit/Receive Selection** 1: Enable IrDA transmitter 0: Enable IrDA receiver
[0]	Reserved	Reserved

UART LIN Network Control Register (LINCON)

Register	Offset	R/W	Description	Reset Value
LINCON	UART0_BA + 0x2C	R/W	UART0 LIN Network Control/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**
LIN_TX_EN	**LIN_RX_EN**	**Reserved**		**LINBCNT**			

Table 5-23 UART LIN Network Control Register (LINCON, address 0x4006_002C)

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

UART Function Select Register (FUNSEL)

Register	Offset	R/W	Description	Reset Value
FUNSEL	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						**IrDA_EN**	**LIN_EN**

Table 5-24 UART Function Select Register (FUNSEL, address 0x4006_0030)

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

5.18 Flash Memory Controller (FMC)

5.18.1 Overview

The I91032 series is available with 64kbytes of on-chip embedded Flash EEPROM for application program and data flash memory. The memory can be updated through procedures for In-Circuit Programming (ICP) through the ARM Serial-Wire Debug (SWD) port or via In-System Programming (ISP) functions under software control. In-System Programming (ISP) functions 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 512 Byte. This boundary is also specified in the CONFIG memory.

5.18.2 Features

- AHB interface compatible
- Running up to 49 MHz with one wait state and 24.5 MHz without wait state for discontinuous address read access
- Mini-cache to reduce flash access and power consumption.
- 64KB application program memory (APROM)
- 4KB in system programming (ISP) boot loader program memory (LDROM)
- Configurable data flash with 512 Bytes sector erase unit
- Programmable data flash start address.
- In System Program (ISP) capability to update on chip Flash EEPROM

5.18.3 Flash Memory Controller Block Diagram

The flash memory controller consist of AHB slave interface, ISP control logic, writer interface and flash macro interface timing control logic. The block diagram of flash memory controller is shown as following:

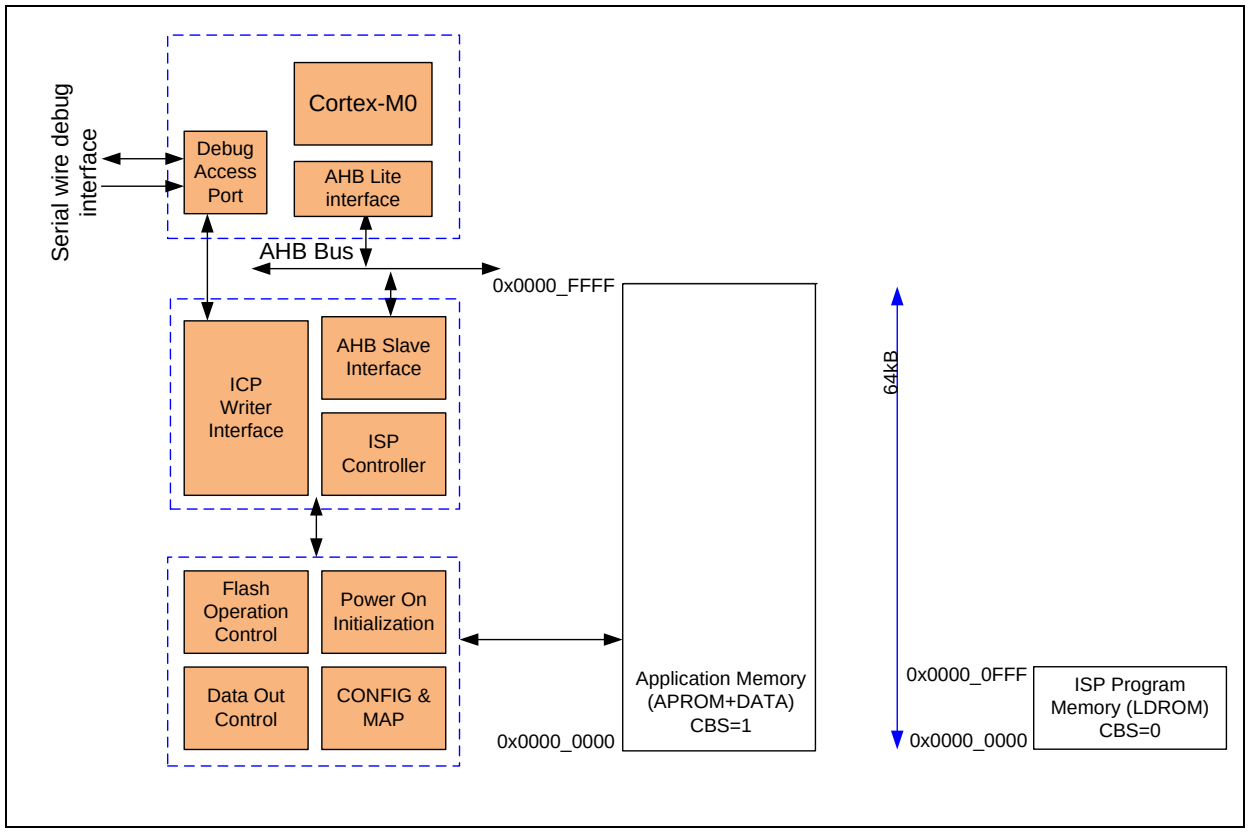


Figure 5-54 Flash Memory Control Block Diagram

5.18.4 Flash Memory Organization

The I91032 flash memory consists of Application Program (APROM) memory (64/32/16kB), 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 (DFBADR) is defined in CONFIG memory and determines the relative size of the APROM and DATAF partitions.

Table 5-25 Memory Address Map

Block Name	Size	Start Address	End Address
APROM	64kB	0x0000_0000	0x0000_FFFF (64KB) OR DFBADR-1 if DFEN!=0
DATAF	User Configurable	DFBADR	0x0000_FFFF (64kB)
LDROM	4 KB	0x0010_0000	0x0010_0FFF
CONFIG	512B	0x0030_0000	0x0030_01FF

The Flash memory organization is shown as below:

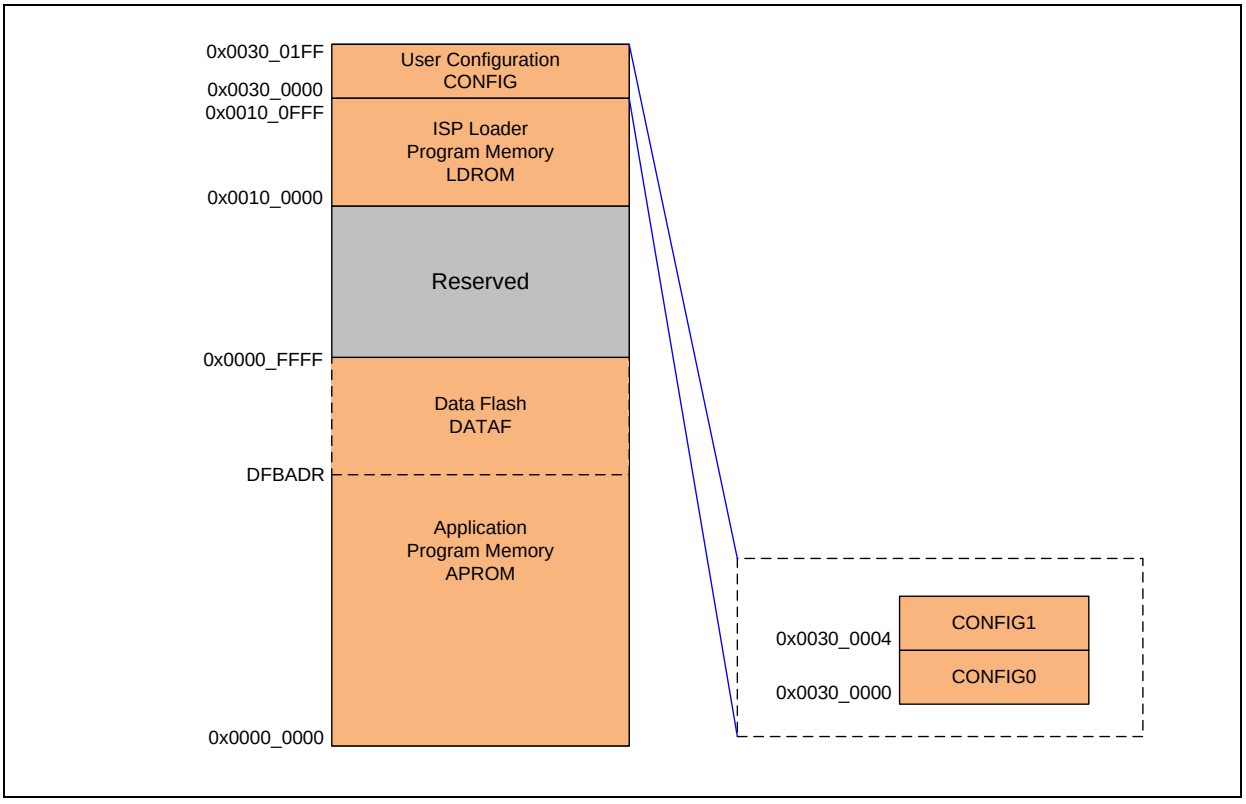


Figure 5-55 Flash Memory Organization

5.18.5 Boot Selection

The I91032 provides an in-system programming (ISP) feature to enable user to update the application program memory when the chip is mounted on a PCB. A dedicated 4kB boot loader program memory is used to store ISP firmware. The user customizes this firmware to implement a protocol specific to their system to download updated application code. This firmware could utilize device peripherals such as UART, SPI or I2C to fetch new application code. The memory area from which the I91032 boots is controlled by the CBS bit in Config0 register.

5.18.6 Data Flash (DATAF)

The I91032 provides Data Flash for user to store data which is read/write thru ISP registers. The erase unit is 512 bytes. When a word will be changed, all 128 words need to be copied to another page or SRAM in advance. The data flash base address is defined by DFBA if DFEN bit in Config0 is enabled. For example for 4k/2k/1k/0kB data flash, the DFBA setting value is listed in the following table.

Table 5-26 Data Flash Table

Data Flash / APROM	4kB (DFEN=0)	2kB (DFEN=0)	1kB (DFEN=0)	0kB (DFEN=1)
64k Flash	DFBA=0x0000_F000	DFBA=0x0000_F800	DFBA=0x0000_FC00	DFEN=1
32k Flash	DFBA=0x0000_7000	DFBA=0x0000_7800	DFBA=0x0000_7C00	DFEN=1
16k Flash	DFBA=0x0000_3000	DFBA=0x0000_3800	DFBA=0x0000_3C00	DFEN=1

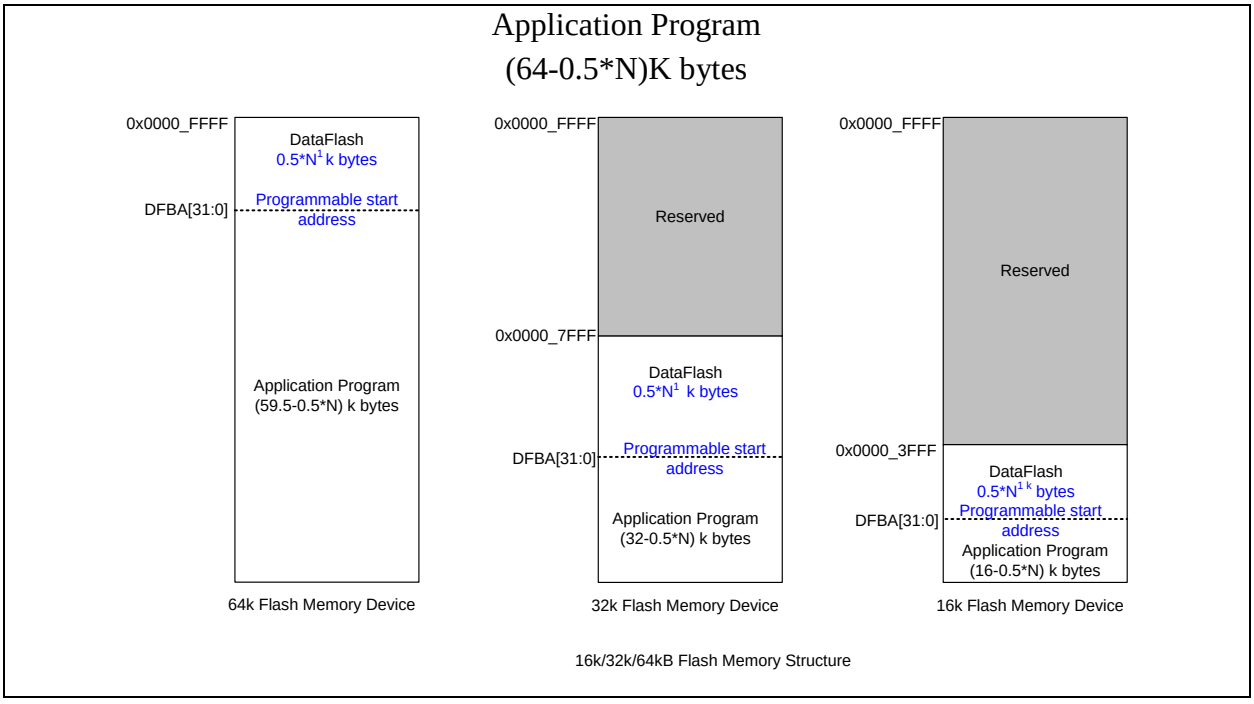


Figure 5-56 Flash Memory Structure

5.18.7 Brown-out Detection

The I91032 includes brown-out detection function for monitoring the voltage on VCCD the main digital supply voltage. If VCCD voltage falls below level setting of both CBOVEXT and CBOV, the BOD event will be triggered when BOD enabled. User can decide to use BOD reset by enable CBORST or just enable BOD interrupt by NVIC when BOD detected. Because BOD reset is issued whenever Digitalvoltage falls below the level setting of CBOV, user must make sure the CBOV setting to avoid BOD reset right after BOD reset enabled. For example, if the Digital is 3.3V and CBOVEXT is 0'b, CBOV could only be 00'b or 01'b. Otherwise, the system will be halted in BOD reset state when BOD reset is enabled and CBOV is 10'b or 11'b.

5.18.8 User Configuration (CONFIG)

Config0 (Address = 0x0030_0000)

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

Config0	Address = 0x0030_0000	
Bits	**Description**	
[31:28]	-	**Reserved**
[27]	**CLVR**	**Config Low Voltage Reset Enable.** 0: LVR reset enabled after configuration 1: LVR Reset disabled after configuration.
[26]	**CBHYS**	**Config BOD Hysteresis** Sets the BOD detect hysteresis after configuration.
[25:22]	**CBOV**	**Config BOD Voltage** Sets the BOD detect voltage after configuration.
[21]	**CBORST**	**Config BOD Reset** 0: BOD reset enabled after configuration

Config0	Address = 0x0030_0000	
Bits	**Description**	
		1: BOD Reset disabled after configuration.
[20]	**CBODEN**	**Config BOD Enable.** 0: BOD function enabled after configuration 1: BOD disabled afer configuration.
[19:8]	-	**Reserved**
[7]	**CBS**	**Chip Boot Selection** 0 = LDROM 1 = APROM.
[6:2]	-	**Reserved**
[1]	**LOCK**	**Security Lock** 0 = Flash data locked. 1 = Flash data unlocked. When flash data is locked, only device ID, unique ID, user configuration can be read by writer and ICP through serial debug interface. Others data is locked as 0xFFFFFFFF. ISP can read data anywhere regardless of LOCK bit value.
[0]	**DFEN**	**Data Flash Enabled** 0 = Data Flash Enabled. 1 = Data Flash Disabled.

Note: The reserved bits of user configuration should be kept as '1'.

Config1 (Address = 0x0030_0004)

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

Config1	Address = 0x0030_0004	
Bits	**Description**	
[31:16]	-	**Reserved**
[15:0]	**DFBA[15:0]**	**Data Flash Base Address** The Data Flash base address is defined by user. Since on chip flash erase unit is 512 bytes, it is mandatory to keep bit 8-0 as 0.

5.18.9 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 I91032 provides an ISP mode allowing a device to be reprogrammed under software control.

ISP is performed without removing the device from the system. Various interfaces enable LDROM firmware to fetch new program code from an external source. A common method to perform ISP would be via a UART controlled by firmware in LDROM. In this scenario, a PC could transfer new APROM code through a serial port. The LDROM firmware receives it and re-programs APROM through ISP commands. An alternative might be to fetch new firmware from an attached SD-Card via the SPI interface.

5.18.10 ISP Procedure

The I91032 will boot from APROM or LDROM from a power-on reset as defined by user configuration bit CBS. If user desires to update application program in APROM, the ISPCON.BS can be set to ’1’ and a software reset issued. This will cause the chip to boot from LDROM. An example flow diagram of the ISP sequence is shown in Figure 5-59.

The ISPCON register is a protected register, user must first follow the unlock sequence (see Register Lock Control Register (SYS_REGLCTL)) to gain access. This procedure is to protect the flash memory from unintentional access.

To enable ISP functionality software must first ensure the ISP clock (AHBCLK.ISP_EN) is present then set the ISPCON.ISPEN bit.

Several error conditions are checked after software writes the ISPTRIG register. If an error condition occurs, ISP operation is not started and the ISP fail flag (ISPCON.ISPFF) will be set instead. The ISPFF flag will remain set until it is cleared by software. Subsequent ISP procedure can be started even if ISPFF is set. It is recommended that software check ISPFF bit and clear it after each ISP operation if set.

When ISPTRIG register is set, the CoretxM0 CPU will wait for ISP operation to finish, during this period; peripherals operate as usual. If any interrupt requests occur, CPU will not

service them until ISP operation finishes. As the ISP functions affect the operation of the flash memory M0 instruction pipeline should be flushed with an ISB (Instruction Synchronization Barrier) instruction after the ISP is triggered.

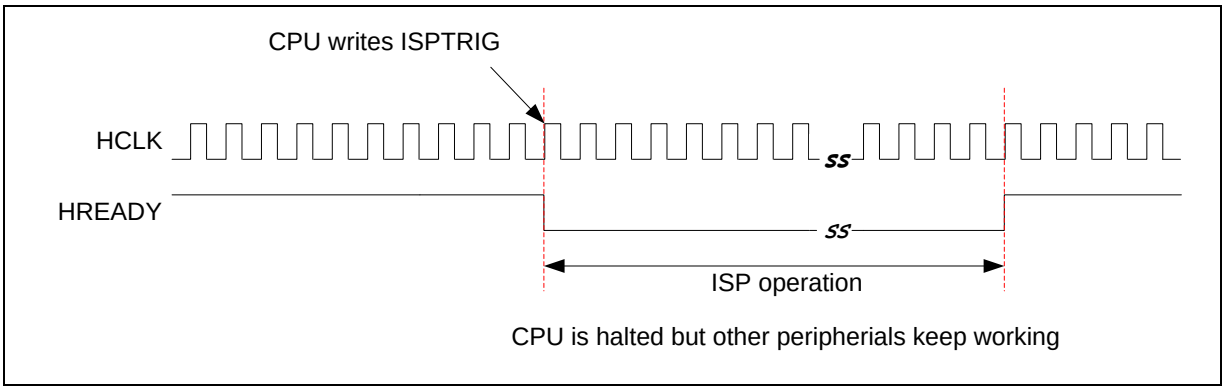


Figure 5-57 ISP Operation Timing

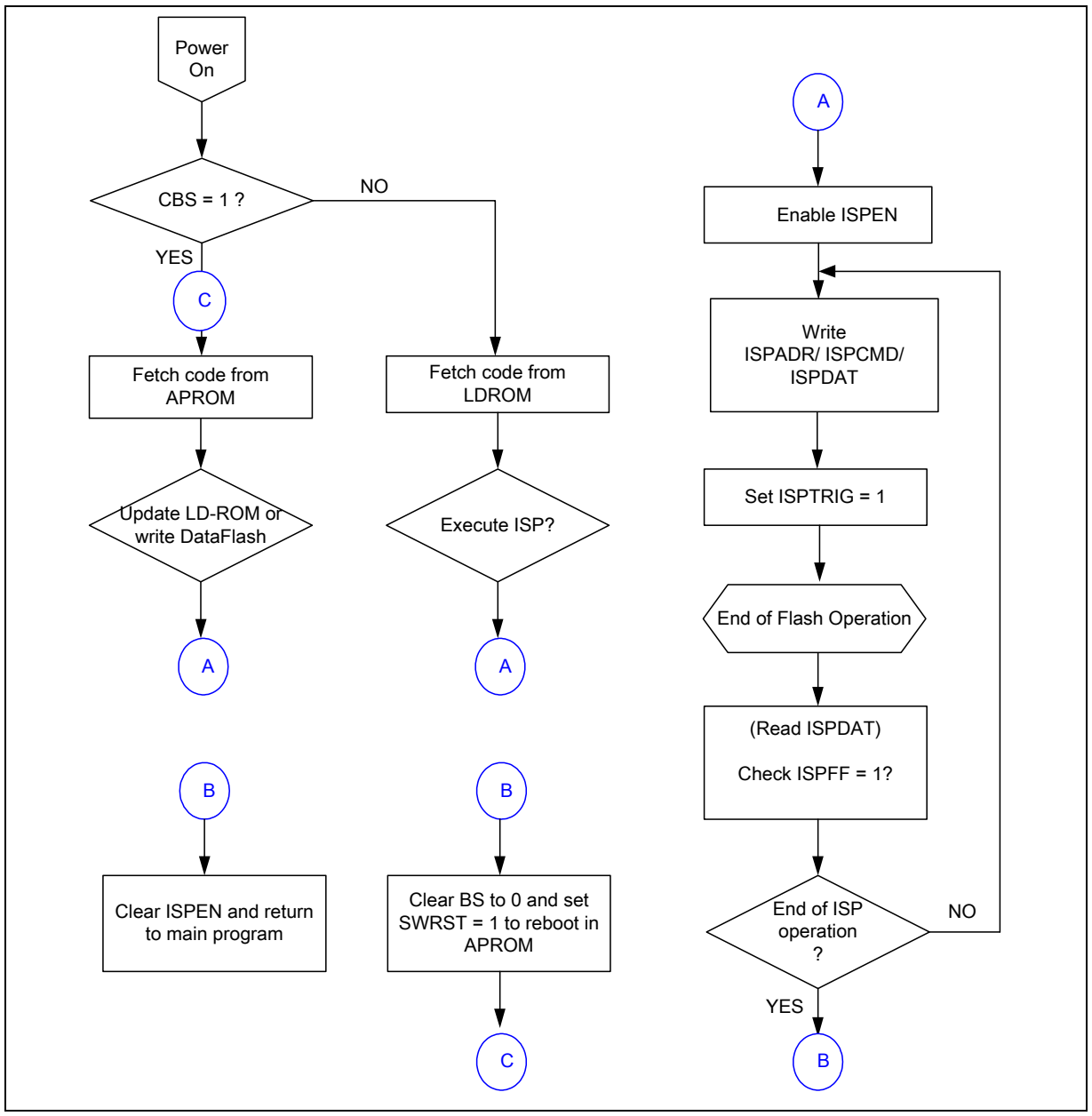


Figure 5-58 Boot Sequence and ISP Procedure

The ISP command set is shown in Table 5-27. Three registers determine the action of a command: ISPCMD is the command register and accepts commands for reading ID registers and read/write/erase of flash memory. The ISPADR is the address register where the flash memory address for access is written. ISPDAT is the data register that input data is written to and return data read from. An ISP command is executed by setting ISPCMD, ISPDAT and ISPADR then writing to the trigger register ISPTRIG.

There is an ISP command to read the device ID register. This register returns a code that reports the memory configuration of theI91032 series part as given in Fig 5-58.

Table 5-27 ISP Command Set

ISP Mode	ISPCMD	ISPADR			ISPDAT
	ISPCMD[5:0]	A21	A20	A[19:0]	D[31:0]
Standby	0x3x	x	x	x	x
Read Company ID	0x0B	x	x	x	Returns 0x0000_00DA
Read Device ID	0x0C	x	x	0x00000	
FLASH Page Erase	0x22	0	A[20]	A[19:0]	x
FLASH Program	0x21	0	A[20]	A[19:0]	Data input
FLASH Read	0x00	0	A[20]	A[19:0]	Data output
CONFIG Page Erase	0x22	1	1	A[19:0]	x
CONFIG Program	0x21	1	1	A[19:0]	Data input
CONFIG Read	0x00	1	1	A[19:0]	Data output

5.18.11 Flash Control Register Map

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

Register	Offset	R/W	Description	Reset Value
FCM Base Address: **FMC_BA = 0x5000_C000**				
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0000_0000
FMC_ISPADR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000
FMC_DFBADR	FMC_BA+0x14	R	Data Flash Base Address Register	0xXXXX_XXXX

5.18.12 Flash Control Register Description

ISP Control Register (FMC_ISPCON)

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

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

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

Bits		Description
[21]	CACHE_DIS	**Cache Disable** When set to 1, caching of flash memory reads is disabled.
[18:16]	WAIT_CFG	**Flash Access Wait State Configuration** 0x11: Zero wait states. HCLK < 24MHz 0x01: Two wait states. 0x02: One wait state. HCLK <= 50MHz 0x00: Four wait state. Before changing WAIT_CFG, ensure HCLK speed is < 25MHz.
[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. 1 = LDROM can be updated when the MCU runs in APROM. 0 = LDROM cannot be updated
[4]	**CFGUEN**	**CONFIG Update Enable** 1 = Enable, 0 = Disable When enabled, ISP functions can access the CONFIG address space and modify device configuration area.
[3:2]	Reserved	Reserved
[1]	**BS**	**Boot Select** 1: LDROM, 0: APROM 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** 1 = Enable ISP function 0 = Disable ISP function

ISP Address Register (FMC_ISPADR)

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

31	30	29	28	27	26	25	24
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]	**ISPADR**	**ISP Address Register** This is the memory address register that a subsequent ISP command will access. ISP operation are carried out on 32bit words only, consequently ISPARD[1:0] must be 00b for correct ISP operation.

ISP Data Register (FMC_ISPDAT)

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

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

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

ISP Command (FMC_ISPCMD)

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

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

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

ISP Trigger Control Register (FMC_ISPTRG)

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

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

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

Bits	Description	
[31:1]	Reserved	Reserved
[0]	**ISPGO**	**ISP Start Trigger** Write 1 to start ISP operation. This will be cleared to 0 by hardware automatically when ISP operation is finished. 1 = ISP is on going 0 = ISP operation is finished After triggering an ISP function M0 instruction pipeline should be flushed with a ISB instruction to guarantee data integrity. This is a protected register, user must first follow the unlock sequence (see Register Lock Control Register (SYS_REGLCTL)) to gain access.

Data Flash Base Address Register (FMC_DFBADR)

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

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

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

6. Revision History

Version	Date	Substantial Changes	Page
A1.0	Dec. 2016	Initial Release	All